MSc-IT

MATHEMATICAL FOUNDATIONS OF COMPUTER SCIENCE

# GAUHATI UNIVERSITY

## Institute of Distance and Open Learning

Semester- I

MSc-IT

Paper: INF 1046

# Mathematical Foundations of Computer Science

www.idolgu.in

M.Sc.-IT-19-I-1046

GAUHATI UNIVERSITY

# Institute of Distance and Open Learning

## First Semester

## M.Sc.-IT

(under CBCS)

**Paper: INF-1046**

# MATHEMATICAL FOUNDATIONS OF COMPUTER SCIENCE

**Contents:**

## Contributors:

**Mr. Tabiruddin Ahmed** (Block I : Units- 1 & 4)
Asstt. Prof., Dept. of BCA
K.C. Das Commerce College, Guwahati, Assam

**Dr. Khurshid Alam Borbora** (Block I: Units- 2 & 3)
Assistant Professor, GUIDOL

**Dr. Hillol Kanti Bhattacharjee** (Block I : Unit- 5)
Asstt. Prof., Dept. of Computer Science
NERIM, Guwahati, Assam

**Mr. Harekrishna Deka** (Block I : Units- 6 & 7)
Asstt. Prof., HCB School of Science and Technology
KKHSOU, Assam

**Dr. Hemen Dutta** (Block I: Unit- 8)
Asstt. Prof., Dept. of Mathematics
Gauhati University, Assam

**Dr. Santanu Acharjee** (Block II : Unit- 1)
Asstt. Prof., Dept. of Mathematics
Gauhati University, Assam

**Ms. Manasi Hazarika** (Block II : Unit- 2)
Asstt. Prof., Dept. of Computer Science and Engineering
Assam Don Bosco University, Guwahati, Assam

**Dr. Sonia Sarmah** (Block II : Unit- 3)
Asstt. Prof., Dept. of Computer Applications
Assam Don Bosco University, Guwahati, Assam

**Dr. Utpal Barman** (Block II : Unit- 4)
Asstt. Prof., Dept. of Computer Science and Engineering
GIMT, Guwahati, Assam

**Dr. Pranab Das** (Block II : Unit- 5)
Asstt. Prof. (Sr.), Dept. of Computer Applications
Assam Don Bosco University, Guwahati, Assam

**Dr. Tapashi Kashyap Das** (Block III : Unit- 1)
Asstt. Prof., HCB School of Science and Technology
KKHSOU, Assam

**Mr. Mridul Suklabaidya** (Block III : Units- 2, 3, 4 & 5)
Teaching Associate, Department of Computer Science
Gauhati University, Assam

## Content Editor:

**Prof. Bipan Hazarika**
Dept. of Mathematics, Gauhati University

## Course Coordination:

**Prof. Dandadhar Sarma** Director, IDOL, Gauhati University
**Prof. Anjana Kakoti Mahanta** Prof., Dept. Computer Science, G.U.

## Cover Page Designing:

Bhaskar Jyoti Goswami IDOL, Gauhati University



**May, 2022**




Published on behalf of Institute of Distance and Open Learning, Gauhati University by the Director, and printed at Gauhati University Press, Guwahati-781014.

# BLOCK I:
# DISCRETE MATHEMATICAL STRUCTURES AND MATHEMATICAL LOGIC

# UNIT 1: CONGRUENCE, PERMUTATION AND COMBINATION WITH REPETITION

***Space for learners:***

**Unit Structure:**

## 1.1 INTRODUCTION

In this unit, you will learn about a useful way of comparing the remainder of two integers, called congruence. You will also learn various properties of congruence with their proof like Addition rule of congruence, Multiplication rule of congruence, Power rule of congruence and Cancellation rule of congruence. You will also learn how Power rule can be used to check the divisibility of certain large numbers. Again, you will learn the concept of Least Residues and Modular Arithmetic with some examples.

In the middle part, you will learn a very familiar concept of Mathematics, Permutation, i.e., how a number of objects can be arranged in a definite order taking some or all at a time. You will learn Factorial Notation with some examples, which is mostly used in Permutation as well as Combination. Again, the concept of Fundamental principle of Counting is explained here with some examples. We can also learn how we arrange n different objects taking r at a time if some objects repeats, i.e., Permutation with repetition with some examples. You will also learn how n objects can be arranged if the objects are distinct objects. Again, you will learn the arrangement of n distinct objects around a fix circle where Clockwise and Anticlockwise orders are different as well as same with some examples. Again, you can see the how certain restrictions can be imposed on Permutation, i.e., Restricted Permutation and some examples of it.

In the latter part, you will learn another very familiar concept of Mathematics, called Combination, i.e., the selection of all or part of a set of objects without regard to the order in which objects are selected with various examples. You will again learn the concept of Restricted Combination, i.e., how Combination can be made if there are certain restriction.

## 1.2 UNIT OBJECTIVES

After going through this unit, you will be able to:

- Understand the concept of Congruence
- Know the Properties of Congruence
- Know the concept of Least Residues

***Space for learners:***

- Know Modular Arithmetic
- Define Permutation
- Understand the Concept of Factorial Notation
- Know the fundamental concept of Counting
- Learn Permutation with repetition
- Learn Permutation of n objects if the objects are distinct
- Learn the concept of Circular Permutation
- Learn the concept of Restricted Permutation
- Learn the concept of Combination
- Learn the concept of Restricted Combination

## 1.3 CONGRUENCE

**Definition1**: Let n be a positive integer. Two integers a and b are congruent modulo n if they each have the same remainder on division by n. If a and b are congruent modulo n, then it is written symbolically as

$$a \equiv b \pmod{n}.$$

For example, 19 and 12 are congruent modulo 7; that is,

$19 \equiv 12 \pmod{7}$

because 19 and 12 each have remainder 5 on division by 7.

Also, −8 and 10 are congruent modulo 6; that is,

$-8 \equiv 10 \pmod{6}$,

because −8 and 10 each have remainder 4 on division by 6.

**Definition 2**: Let n be a fixed integer. Two integers a and b are said to be congruent modulo n if $n \mid a-b$ i.e., if a-b is divisible by n.

For example, 3 and 24 are said to be congruent modulo 7, because (3-24)= -21, which is divisible by 7.

Therefore, $3 \equiv 24 \pmod{7}$.

Again, if a and b are not congruent modulo n, then the difference between a and b is not an integer multiple of n; that is, $a - b$ is not divisible by n.

*Space for learners:*

For example, 4 and 6 are not congruent modulo 5,because 4-6 = -2,which is not divisible by 5.

**Illustrative Example:**

List all integers x in the range $1 < x < 100$ that satisfy $x \equiv 3 \pmod 7$.

**Solution:**

Given,

$$x \equiv 3 \ (\text{mod } 7)$$

$$\text{i.e.} \quad 7 \mid x-3$$

$$\text{i.e.} \quad x-3 = 7k,\ k \in Z$$

$$\text{i.e.} \quad x = 3+7k \ \ldots\ldots\ldots\ldots\ldots\ldots(1)$$

Therefore,

$$1 < 3+7k < 100$$

$$-2 < 7k < 97.$$

From this we obtain the values of k as 0,1,2,3,4,5,6,7,8,9,10,11,12,13.

Now, putting the values of k in equation (1), we get the values of x=3,10,17,24,31,38,45,52,57,66,73,80,87 and 94.

## 1.4 PROPERTIES OF CONGRUENCE

1. $a \equiv a \pmod n$
2. if $a \equiv b \pmod n$ then $b \equiv a \pmod n$
3. if $a \equiv b \pmod n$ and $b \equiv c \pmod n$, then $a \equiv c \pmod n$
4. Addition Law of Congruence

   If $a \equiv b \pmod n$ and $c \equiv d \pmod n$, then $a + c \equiv b + d \pmod n$
5. Multiplication Law of Congruence

   If $a \equiv b \pmod n$ and $c \equiv d \pmod n$, then $ac \equiv bd \pmod n$.
6. For any integers a, b, and c

   (a) If $a \equiv b \pmod n$, then $a + c \equiv b + c \pmod n$.

   (b) If $a \equiv b \pmod n$, then $ca \equiv cb \pmod n$.

**Proof (1)**: For any integer a and any fixed positive integer n,

Space for learners:

We have a-a=0, which is divisible by n

Therefore, $n \mid (a-a)$,since 0 is divisible by any integer.

Therefore $a \equiv a \bmod n$.

For example, $5 \equiv 5 (\bmod 7)$, because 5-5=0 is divisible by 7.

**Proof (2):** Let, $a \equiv b \pmod n$, then, $n|(a - b)$.

Therefore, $n|(-1)(a - b)$

Or, $n|(b - a)$.

Therefore, $b \equiv a \pmod n$.

So, if $a \equiv b \pmod n$ then $b \equiv a \pmod n$.

For example, if $3\equiv 18 \pmod 5$, then $18 \equiv 3 \pmod 5$, because 3-18 = -15 and 18-3=15, both -15 and 15 are divisible by 5.

**Proof (3):** Let $a \equiv b \pmod n$.

Then, $n|(a-b)$……………………………………………….(1)

Again,

Let, $b \equiv c \pmod n$.

Then, $n|(b-c)$……………………………………………….(2)

From equation (1) & (2), we get,

$n|(a - b + b- c)$ [by Linear Combination Theorem]

or $n|(a - c)$.

Thus, $a \equiv c \bmod n$.

Therefore, if $a \equiv b \pmod n$ and $b \equiv c \pmod n$, then $a \equiv c \pmod n$

For example, we consider

$7 \equiv 12 \pmod 5$ [Because 7-12=-5,which is divisible by 5]

and $12 \equiv 22 (\bmod 5)$ [Because 12-22=-10,which is divisible by 5]

Now, clearly we can say that

$7 \equiv 22 \pmod 5$ [Because 7-22=-15,Which is divisible by 5]

**Proof (4):** Let, $a \equiv b \pmod n$.

***Space for learners:***

Then, a-b is divisible by n
…………………………………………………(1)

And, $c \equiv d \pmod{n}$.

Then, $c - d$ are divisible by
n……………………………………………(2)

From equation (1) & (2) we get,

$$(a - b) + (c - d) \text{ is divisible by } n.$$

But, $(a - b) + (c - d) = (a + c) - (b + d)$.

So, $(a + c) - (b + d)$ is divisible by n.

Therefore, $a + c \equiv b + d \pmod{n}$.

So, if $a \equiv b \pmod{n}$ and $c \equiv d \pmod{n}$, then $a + c \equiv b + d \pmod{n}$.

For example,

We know that,

$$19 \equiv 1 \pmod{18} \text{ and } 37 \equiv 1 \pmod{18},$$

so, by the addition rule of congruence,

$$19 + 37 \equiv 1 + 1 \equiv 2 \pmod{18}.$$

Or $56 \equiv 2 \pmod{18}$.

**Proof (5):** Let, $a\equiv b \pmod{n}$.

Then, a-b is divisible by n

Then, (a-b)c is also divisible by
n………………………………………(1)

And $c \equiv d \pmod{n}$.

Then, c-d is divisible by n

Then, (c-d)b is also divisible by
n……………………………………..(2)

From equation (1) & (2),

$$(a-b)c+(c-d)b \text{ is also divisible by } n$$

But, $(a - b)c + (c - d)b = ac - bd$.

So, $ac - bd$ is divisible by n.

Hence, $ac \equiv bd \pmod{n}$.

Hence, If $a \equiv b \pmod{n}$ and $c \equiv d \pmod{n}$, then $ac \equiv bd \pmod{n}$.

***Space for learners:***

For Example,

$17 \equiv -2 \pmod{19}$

and $14 \equiv -5 \pmod{19}$.

Therefore,

$17 \times 14 \equiv (-2) \times (-5) \equiv 10 \pmod{19}$ [By Multiplication rule of Congruence]

Or , $238 \equiv 10 \pmod{19}$.

## 1.4.1 Power Rule for Congruences

If $a \equiv b \pmod{n}$, and m is a positive integer, then $a^m \equiv b^m \pmod{n}$.

For example, suppose you wish to find the least residue (remainder) of $19^5$ modulo 9. Since $19 \equiv 1 \pmod 9$, it follows that $195 \equiv 1^5 \equiv 1 \pmod 9$, so the least residue is 1. This is a particularly simple application of the power rule.

## 1.4.2 Cancellation Rule of Congruence

If $ca \equiv cb \pmod{n}$ then $a \equiv b \pmod{n/d}$ where $d=\gcd(c,n)$

For example,

Consider, $33 \equiv 15 \pmod 9$

Now, $3*11 \equiv 3*5 \pmod 9$

$11 \equiv 5 \pmod 3$, by cancellation law of congruence. [since, gcd of 3 and 9 is 3]

Again, by same law we can write $-35 \equiv 45 \pmod 8$

As $-7 \equiv 9 \pmod 8$

## 1.5 LEAST RESIDUES

The least residue of a modulo n is the remainder r that you obtain when you divide a by n. The integer r is one of the numbers 0, 1, . . . , n − 1, and it satisfies $a \equiv r \pmod{n}$.

For example, the least residue of −33 modulo 7 is 2

Because, $-33 = 7 \times (-5) + 2$

Space for learners:

## 1.6 MODULAR ARITHMETIC

Modular arithmetic is the application of the usual arithmetic operations – namely addition, subtraction, multiplication and division – for congruences. Addition, subtraction and multiplication are often simpler to carry out in modular arithmetic than they are normally, because you can use congruences to reduce large numbers to small numbers.

Examples:

1. Find the least residue of 67+68 modulo 6
2. Find the least residue of 17*14 modulo 19

**Solution1:** We know that,

$67 \equiv 1 \pmod 6$ [Because 67-1=66, is divisible by 6]…………………(1)

$68 \equiv 2 \pmod 6$ [Because 68-2=66,is divisible by 6]………………….(2)

From equation (1) & (2),

$67+68 \equiv 1+2 \pmod 6$ [By Addition rule of congruence]

Or $135 \equiv 3 \pmod 6$ [135-3=132,is divisible by 6].

Therefore, the least residue of 67+68 modulo 6 is 3

**Solution 2:** We know that,

$17 \equiv -2 \pmod{19}$ [Because 17-(-2)=19,is divisible by 19]……………..(1)

$14 \equiv -5 \pmod{19}$ [Because 14-(-5)=19,is divisible by 19]………………(2)

From equation (1) & (2),

$17*14 \equiv (-2)*(-5) \equiv 10 \pmod{19}$ [By Multiplication Law of congruence].

Therefore, the least residue of 17*14 modulo 19 is 10.

***Space for learners:***

## 1.7 APPLICATION OF POWER LAW OF CONGRUENCE

Space for learners:

**Example 1**: Find the remainder when $25^{100} + 11^{500}$ is divided by 3.

**Example 2**: Show that $3^{1000} + 3$ is divisible by 28.

**Solution1:**

We know that, $25 \equiv 1 \pmod 3$.

Therefore, $25^{100} \equiv 1^{100} \pmod 3$ [By Power rule of Congruence]

Therefore, $25^{100} \equiv 1 \pmod 3$……………………….(1)

Again, $11 \equiv -1 \pmod 3$.

Therefore, $11^{500} \equiv (-1)^{500} \pmod 3$ [By Power rule of Congruence]

Therefore, $11^{500} \equiv 1 \pmod 3$………………………(2)

From equation (1) & (2),

$25^{100} + 11^{500} \equiv 2 \pmod 3$ [By addition law of Congruence]

Therefore, the remainder is 2.

**Solution 2:** We know that, $3^3 = 27 \equiv -1 \pmod{28}$.

Therefore, $(3^3)^{333} \equiv (-1)^{333} \pmod{28}$ [By power rule of Congruence]

Therefore, $3^{999} \equiv -1 \pmod{28}$.

$3^{1000} = 3^{999}.3 \equiv -1.3 \pmod{28}$ [ By Properties 6(b)]

Therefore, $3^{1000} \equiv -3 \pmod{28}$……………………………………….(1)

Again,

$3 \equiv 3 \pmod{28}$………………………………………(2)

From equation (1) & (2),

$3^{1000} + 3 \equiv -3+3 \equiv 0 \pmod{28}$ [By addition rule of Congruence]

Therefore, $3^{1000} + 3 \equiv 0 \pmod{28}$.

So, the remainder when we divide $3^{1000}+3$ by 28 is 0

Hence, we can say that $3^{1000}+3$ is divisible by 28.

Space for learners:

**CHECK TO YOUR PROGRESS**

1. Which of the following congruences are true?
(a) $11 \equiv 26 \pmod 5$ (b) $9 \equiv -9 \pmod 5$
(c) $28 \equiv 0 \pmod 7$ (d) $-4 \equiv -18 \pmod 7$
(e) $-8 \equiv 5 \pmod{13}$ (f) $38 \equiv 0$ (mod 13

(2). Determine the integers in between 50 and 100 which are congruent to 1 modulo 4.

(3). List all integers x in the range $1 \leq x \leq 100$ that satisfy $x \equiv 7$ ( mod 17 ).

(4). Find the least residues of the following integers modulo 10.
(a) 17 (b) 50 (c) 6 (d) −1 (e) −38

(5). Find the least residues of the following integers modulo 7.
(a) $3 \times 6$ (b) $22 \times 29$ (c) $51 \times 74$

(6).Show that $2^{20}-1$ is divisible by 41

(7). What is the remainder when $3^{5555}$ is divided by 80?

## 1.8 PERMUTATION

A permutation is an arrangement in a definite order of a number of objects taken some or all at a time.

The number of permutations of n different objects taken r at a time, where $0 < r \leq n$ and the objects do not repeat is $n(n-1)(n-2)\ldots(n-r+1)$, which is denoted by ${}^nP_r$orP(n,r)

${}^np_r = \frac{n!}{(n-r)!}$where $0 \leq r \leq n$

### 1.8.1 Factorial Representation

The notation n! represents the product of first n natural numbers, i.e., the product $1 \times 2 \times 3 \times \ldots \times (n-1) \times n$ is denoted as n!. We read this symbol as 'n factorial'.

Thus, $1 \times 2 \times 3 \times 4 \ldots \times (n-1) \times n = n!$

$$1 = 1!$$

$$1 \times 2 = 2!$$

$$1\times 2 \times 3 = 3!$$

$1 \times 2 \times 3 \times 4 = 4!$

We define $0! = 1$

Again,

$5! = 5 \times 4! = 5 \times 4 \times 3! = 5 \times 4 \times 3 \times 2! = 5 \times 4 \times 3 \times 2 \times 1!$

**Example 1**. Evaluate (a) $3!$ (b) $2! + 4!$ (c) $2! \times 3!$

**Solution**: (a) $3! = 3\times2\times1= 6$

(b) $2! = 2\times1 = 2$

$4! = 4\times3\times 2\times1 = 24$

Therefore, $2! + 4! = 2 + 24 = 26$

(c) $2! \times 3! =2 \times6= 12$

**Example 2**: Find the value of (a) ${}^np_0$ (b) ${}^np_1$ (c) ${}^np_n$ (d) ${}^6p_3$

**Solution 2(a)**

**We know that**

$${}^np_r= \frac{n!}{(n-r)!}$$

Therefore,

$${}^np_0= \frac{n!}{(n-0)!}$$

$$=\frac{n!}{(n)!}$$
$$=1$$

So, ${}^np_0= 1$. Similarly ${}^5p_0= {}^7p_0=1$.

2. (b) We know that,

$${}^np_r= \frac{n!}{(n-r)!}$$
$$\Rightarrow {}^np_1= \frac{n!}{(n-1)!}$$

$$= \frac{n*(n-1)!}{(n-1)!}$$

Space for learners:

$= n$

So, ${}^{n}p_1 = n$, similarly $5p_1 = 5$, ${}^{6}p_1 = 6$.

2(c) We know that

$${}^{n}p_r = \frac{n!}{(n-r)!}$$

$${}^{n}p_n = \frac{n!}{(n-n)!}$$

$$= \frac{n!}{0!}$$

$$= n! \ \text{[since, } 0! = 1]$$

Therefore, ${}^{n}p_n = n!$. Similarly, ${}^{5}p_5 = 5!$ , ${}^{7}p_7 = 7!$

2 (d): We know that,

$${}^{n}p_r = \frac{n!}{(n-r)!}$$

$So,\quad {}^{6}p_3 = \frac{6!}{(6-3)!}$

$$= \frac{6!}{3!}$$

$$= \frac{6\times5\times4\times3!}{3!}$$

$$=6\times5\times4$$

$$=120$$

**Example 3:**

(a) Prove that ${}^{n}p_r = {}^{n-1}p_r + r.{}^{n-1}p_{r-1}$

(b) Find the value of n,if ${}^{n}p_7 = 42\times {}^{n}p_5$

(c) *Find the valueof n if* ${}^{n}p_5 : {}^{n}p_3 = 2:1$

**Solution 3(a):**

$$RHS = {}^{n-1}p_r + r.{}^{n-1}p_{r-1}$$

$$= \frac{(n-1)!}{(n-1-r)!} + r\times\frac{(n-1)!}{\{(n-1)-(r-1)\}!} \ \left[\text{Since, } {}^{n}p_r = \frac{n!}{(n-r)!}\right]$$

$$= \frac{(n-1)!}{(n-r-1)!} + r\times\frac{(n-1)!}{(n-r)!}$$

$$= \frac{(n-1)!}{(n-r-1)!} + r\times\frac{(n-1)!}{(n-r)\times(n-r-1)!} \ [\text{Since, } n!=n\times(n-1)!\ ]$$

Space for learners:

$=\frac{(n-1)!}{(n-r-1)!}\left(1+\frac{r}{n-r}\right)$

$=\frac{(n-1)!}{(n-r-1)!}\times\frac{n}{n-r}$

$=\frac{n!}{(n-r)!}$

$= {}^{n}p_{r}$

=L.H.S

*Hence*, ${}^{n}p_{r} = {}^{n-1}p_{r} + r.{}^{n-1}p_{r-1}$

3(b):

Given,

${}^{n}p_{7} = 42\times {}^{n}p_{5}$

$\Rightarrow \frac{n!}{(n-7)!}=42\times\frac{n!}{(n-5)!}$

$\Rightarrow \frac{1}{(n-7)!}=\frac{42}{(n-5)!}$

$\Rightarrow (n-5)! = 42\times(n-7)!$

$\Rightarrow (n-5)(n-6)(n-7)! = 42\times(n-7)!$

$\Rightarrow (n-5)(n-6)= 42$

$\Rightarrow (n-5)(n-6)=7\times6$

$\Rightarrow (n-5)=7$ or $(n-6)=6$

$\Rightarrow n=12$

The required value of n is 12

**Solution:3(c):**

Given,

${}^{n}p_{5} : {}^{n}p_{3} = 2:1$

$\Rightarrow \frac{n!}{(n-5)!}:\frac{n!}{(n-3)!}=2:1$

$\Rightarrow \frac{\frac{n!}{(n-5)!}}{\frac{n!}{(n-3)!}}=\frac{2}{1}$

$\Rightarrow \frac{n!}{(n-5)!}\times\frac{(n-3)}{n!}=\frac{2}{1}$

***Space for learners:***

$\Rightarrow \frac{(n-3)!}{(n-5)!} = 2$

$\Rightarrow \frac{(n-3)(n-4)(n-5)!}{(n-5)!} = 2$

$\Rightarrow (n-3)(n-4)=2$

$\Rightarrow (n-3)(n-4)=2\times1$

$\Rightarrow n-3=2$ or $n-4=1$

$\Rightarrow n=5$

## 1.8.2 Fundamental Principle of Counting (Multiplication Principle)

"If an event can occur in m different ways, following which another event can occur in n different ways, then the total number of occurrences of the events in the given order is $m\times n$."

**Example1**: How many 3-digit numbers can be formed with the digits 1,4,7,8 and 9 if the digits are not repeated?

**Example2**: There are 4 books on fairy tales, 5 novels and 3 plays. In how many ways can you arrange these so that books on fairy tales are together, novels are together and plays are together and in the order, books on fairy tales, novels and plays.

**Solution 1:**

Three digit numbers will have units, ten's and hundred's place.

Out of 5 given digits any one can take the unit's place.

This can be done in 5 ways. ... (i)

After filling the unit's place, any of the four remaining digits can take the ten's place.

This can be done in 4 ways. ... (ii)

After filling in ten's place, hundred's place can be filled from any of the three remaining digits.

This can be done in 3 ways. ... (iii)

*Space for learners:*

∴ By counting principle, the number of 3 digit numbers = $5\times4\times3 = 60$

**Solution2:**

There are 4 books on fairy tales and they have to be put together.

They can be arranged in 4! ways.

Similarly, there are 5 novels.

They can be arranged in 5! ways.

And there are 3 plays.

They can be arranged in 3! ways.

So, by the counting principle all of them together can be arranged in $4!\times5!\times3!$ ways = 17280 ways.

**CHECK TO YOUR PROGRESS**

8 (a). **Evaluate:**

(i) 6! (ii) 7! (iii) 7! + 3! (iv) $6!\times4!$

8 (b). Which of the following statements are true?

(i) $2!\times3! = 6!$ (ii) $2! + 4! = 6!$ (iii) $4! - 2! = 2!$

9.Find the value of

$(a)$ ${}^{15}p_4$ (b) ${}^{11}p_5$ (c) ${}^{9}p_0$

10.Find the value of n if $2*{}^{9}p_n = {}^{10}p_n$

11.$Find\ theValue\ of\ r\ if$ ${}^{18}p_{r-1}:{}^{17}p_{r-1} = 9:7$

12. Find the value of r if $5*{}^{4}p_r = 6*{}^{5}p_{r-1}$

13.How many words of 4 letters with or without meaning can be formed from the letters of the word RICE.

14.Without repetition how many 4 digits numbers can be formed with the digits 1,3,5,7,9.

Space for learners:

### 1.6.3 Permutation with Repetition

Space for learners:

The number of permutations of n different objects taken r at a time, where repetition is allowed, is $n^r$.

**Example1**: Find the number of 4 digit numbers that can be formed using the digits 1, 2, 4, 5, 7, 8 when repetition is allowed.

**Example2**: Ten different letters of an alphabet are given. Words with 5 letters are formed from these letters. Find the number of words which have at least one letter repeated.

**Example3**: How many numbers lying between 100 and 1000 can be formed with the digits 0, 1, 2, 3, 4, 5, if the repetition of the digits is not allowed?

**Solution1:**

The number of 4 digit numbers that can be formed using the digits 1, 2, 4, 5, 7, 8 when repetition is allowed = $6^4 = 1296$

**Solution 2:**

The number of 5 letter words using ten different letters when repetition is allowed = $10^5$

Again,

The number of 5 letter words using ten different letters when repetition is not allowed=${}^{10}p_5$

$Therefore, the$number of 5 letter words using ten different letters in which at least one

letter repeated = $10^5 - {}^{10}p_5 = 100000 - 30240 = 69760$

**Solution3:**

Every number between 100 and 1000 is a 3-digit number. We, first, have to count the permutations of 6 digits taken 3 at a time.

This number would be ${}^6p_3$.

But, these permutations will include those also where 0 is at the 100's place. For example, 092, 042. . .etc. are such numbers which are actually 2-digit numbers and hence the number of such numbers has to be subtracted from ${}^6p_3$ to get the required number. To get the number of such numbers, we fix 0 at the 100's place and rearrange the remaining 5 digits taking 2 at a time. This number is ${}^5p_2$ .

So, the required number is = $^6p_3 - ^5p_2$

$= 100$

## 1.8.4 Permutations when all the Objects are Not Distinct Objects

The number of permutations of n objects, where p1 objects are of one kind, p2 are of second kind, ..., pk are of kth kind and the rest, if any, are of different kind is $\frac{n!}{p1!\,p2!\ldots\ldots pk!}$

**Example1**: How many words can be formed with the letters of the words COMMITTEE?

**Example2**: In how many ways can 4 red, 3 yellow and 2 green balls be arranged in a row if the balls of the same colour are indistinguishable?

**Solution1:**

Here, there are 9 objects (letters) of which there are 2M's, 2 T's, 2 E's and rest are all different.

Therefore, the required number of arrangements $=\frac{9!}{2!2!2!} = \frac{9\times8\times7\times6\times5\times4\times3\times2\times1}{2\times2\times2}$

$=9\times 7 \times 6 \times 5 \times 4 \times 3 \times 2 \times 1$

=45360

**Solution2:**

Total no of balls=9

So, out of 9 balls, 4 balls are red, 3 balls are yellow and 2 balls are green.

Therefore, the total no of arrangements$=\frac{9!}{4!\,3!\,2!}$

$=\frac{9\times8\times7\times6\times5\times4!}{4!\times6\times2}$

$=\frac{9\times8\times7\times5}{2}$

$=9\times4\times7\times5$

=1260

Space for learners:

## 1.8.5 Circular Permutation

Space for learners:

Circular Permutation is the total number of ways in which n distinct objects can be arranged around a fix circle.

It is of two types

Case1- Clockwise and Anticlockwise orders are different.

Here, the number of circular permutations of n dissimilar things is $(n-1)!$

Case2- Clockwise and Anticlockwise orders are same.

Here, the number of circular permutations of n things is $\frac{1}{2}[(n-1)!]$

**Example1**: Find the number of ways of arranging 7 persons around a circle.

**Example2**: Find the number of ways of arranging 6 boys and 6 girls around a circular table so that (i) all the girls sit together (ii) no two girls sit together iii) boys and girls sit alternatively

**Example3**: Find the number of ways of arranging 6 red roses and 3 yellow roses of different sizes into a garland. In how many of them (i) all the yellow roses are together (ii) no two yellow roses are together

**Solution1:**

Number of persons, n = 7

∴ The number of ways of arranging 7 persons around a circle = $(n - 1)! = 6! = 720$

**Solution2:**

(i) Treat all the 6 girls as one unit. Then we have 6 boys and 1 unit of girls. They can be arranged around a circular table in 6! Ways. Now, the 6 girls can be arranged among themselves in 6! Ways.

∴The number of required arrangements = $6! \times 6! = 720 \times 720 = 5,18,400$

(ii) First arrange the 6 boys around a circular table in 5! ways. Then we can find 6 gaps between them. The 6 girls can be arranged in these 6 gaps in 6! ways.

∴The number of required arrangements = $5! \times 6! = 120 \times 720 = 86,400$

Space for learners:

(iii) The arrangements of boys and girls sit alternatively in same as the arrangements of no two girls sit together or arrangements of no two boys sit together.

First arrange the 6 girls around a circle table in 5! ways.

Then we can find 6 gaps between them.

The 6 boys can be arranged in these 6 gaps in 6! ways.

∴The number of required arrangements = 5! × 6! = 120× 720 = 86,400

**Solution3:**

Total number of roses = 6 + 3 = 9

∴ The number of ways of arranging 6 red roses and 3 yellow roses of different sizes into a garland

$=\frac{1}{2}[(9-1)!$

$=\frac{1}{2}(8!)$

$=\frac{1}{2}\times 40320$

$=20160$

(i) Treat all the 3 yellow roses as one unit. Then we have 6 red roses and one unit of yellow roses.

They can be arranged in garland form in (7 - 1)! = 6! ways.

Now, the 3 yellow roses can be arranged among themselves in 3! ways. But in the case of garlands, clockwise arrangements look alike.

∴The number of required arrangements $=\frac{1}{2}\times 6!\times 3!$

$=\frac{1}{2}\times 720\times 6$

$=2160$

Space for learners:

(ii) First, arrange the 6 red roses in garland form in 5! ways.

(iii) Then we can find 6 gaps between them. The 3 yellow roses can be arranged in these 6 gaps in ${}^6p_3$ ways. But in the case of garlands, clock-wise and anti-clockwise arrangements look alike.

∴The number of required arrangements= $\frac{1}{2} \times 5! \times {}^6p_3$

$=\frac{1}{2} \times 120 \times 6 \times 5 \times 4$

$=7200$

## 1.8.6 Restricted Permutation

Permutation with some specific restrictions is called restricted permutations. Following are some Permutation corresponding to some common restrictions.

The number of permutation of n different things taken r of them at a time in which k particular things

(a) Never Occur is = ${}^{n-k}p_r$

(b) Always occur is ${}^{n-k}p_{r-k} \times {}^rp_k$

(c) Are placed in some specific places in ${}^{n-k}p_{r-k}$

**Example1**:How many wprds can be formed with the letters of the word EQUATION taking 5 at a time if

(a) None of words contains Q,U and T

(b) A and O occur In each word

**Example2**: How many arrangements of the letters of the word 'BENGALI' can be made (i) if the vowels are never together. (ii) if the vowels are to occupy only odd places.

**Solution1:**

(a) There are 8 letters in the word EQUATION. If none of the words contain the 3 letters Q,U and T, then there will be remain 8-3=5 letters.

So, the permutation will be the arrangement of these 5 letters.

Therefore, the required no of words = ${}^5p_5 = 120$

Space for learners:

(b)Since, A and O are always present, So any two of the 5 gaps are to be filled up by the two letters A and O, Which can be done in ${}^5p_2$ ways. After filling 2 of the 5 gaps, the remaining 5-2=3 gaps can be filled up by the 3 letters from the remaining 8-2=6 letters, which will be filled in ${}^6p_3$ ways.

Therefore, the required no of words= ${}^6p_3 \times {}^5p_2$ =2400

***Solution 2:***

(i) Considering vowels a, e, i as one letter, we can arrange 4+1 letters in 5! Ways in each of which vowels are together.

These 3 vowels can be arranged among themselves in 3! ways.

∴ Total number of words = 5! × 3! = 120 × 6 = 720

(ii) There are 4 odd places and 3 even places.

3 vowels can occupy 4 odd places in ${}^4p_3$ ways

And 4 constants can be arranged in ${}^4p_4$ ways.

∴Required number of words = ${}^4p_3 \times {}^4p_4$

=24×24

=576

## 1.9 COMBINATION

A combination is a selection of all or part of a set of object without regard to the order in which objects are selected.

The number of combinations of n things taken r at a time is denoted by ${}^nC_r$ and it is defined by ${}^nC_r = \frac{n!}{(n-r)!\,r!}$ For $0 \leq r \leq n$

### 1.9.1 Restricted Combination

If there are certain restrictions on Combination like a particular object occurring always and occurring never, then it is called Restricted Combination.

The numbers of Combinations of n different things taking r of them at a time if x particular things are

(i) Always included is ${}^{n-x}C_{r-x}$

(ii) Always excluded is ${}^{n-x}C_r$

**Example1:**

Show That

**(a)** $^nC_0=1$

(b) $^nc_r= {}^np_r/r!$

(c) $^nc_1=n$

(d) $^nc_r={}^nc_{n-r}$

(e) $^nc_r+{}^nc_{r-1}={}^{n+1}c_r$

**Solution:**

(a) LHS= $^nC_0$

$=\frac{n!}{(n-0)!\,0!}$ [$^nC_r =\frac{n!}{(n-r)!\,r!}$]

$=\frac{n!}{n!\,1}$ [ 0!=1]

=1

=RHS

**(b)** LHS=$^nC_r$

$=\frac{n!}{(n-r)!\,r!}$

$={}^np_r/r!$ [Since, $^nP_r =\frac{n!}{(n-r)!}$]

=RHS

**(c)** LHS= $^nC_1$

$=\frac{n!}{(n-1)!\times 1!}$ [since, $^nC_r=\frac{n!}{(n-r)!\,r!}$]

$=\frac{n\times(n-1)!}{(n-1)!\times1}$

=n

=RHS

**(d)** LHS= $^nC_r$

$=\frac{n!}{(n-r)!\times r!}$

$=\frac{n!}{(n-r)!\times(n-(n-r))!}$

$={}^nC_{n-r}$

Space for learners:

=RHS

**(e)** LHS= ${}^nC_r + {}^nC_{r-1}$

$=\frac{n!}{(n-r)!\,r!}+\frac{n!}{(n-r+1)!(r-1)!}$

$=\frac{n!}{(n-r)!\,r(r-1)!}+\frac{n!}{(n-r+1)(n-r)!(r-1)!}$

$=\frac{n!}{(n-r)!\,(r-1)!}\left(\frac{1}{r}+\frac{1}{n-r+1}\right)$

$=\frac{n!}{(n-r)!\,(r-1)!}\left(\frac{n-r+1+r}{r(n-r+1)}\right)$

$=\frac{(n+1)\times n!}{r\times(r-1)!(n-r+1)\times(n-r)!}$

$=\frac{(n+1)!}{r!\times(n-r+1)!}$

$=\frac{(n+1)!}{r!\times(n+1-r)!}$

$={}^{n+1}C_r$

=RHS

**Example2:**

(a) Find the value of ${}^9C_7$

(b) If ${}^{12}C_r = {}^{12}C_{r+2}$ find the value of r

(c) If ${}^nC_3 \div {}^nC_2 = 8$,find the value of n.

(d) If ${}^nP_r = 110$ and ${}^nC_r = 55$,find the value of r

**Solution:**

**(a)** ${}^9C_7$

$=\frac{9!}{(9-7)!\times 7!}$

$=\frac{9!}{2!\times 7!}$

$=\frac{9\times 8\times 7!}{2\times 7!}$

$=\frac{9\times 8}{2}$

=36

**(b)** Given

Space for learners:

$^{12}C_r = ^{12}C_{r+2}$

$\Rightarrow {}^{12}C_{12-r} = {}^{12}C_{r+2}$ $[Since, {}^nc_r = {}^nc_{n-r}]$

$\Rightarrow 12 - r = r + 2[if\ {}^nc_r = {}^nc_s\ then\ r = s] \Rightarrow 2r = 10$

$$\Rightarrow r = 5$$

**(c)** Given,

$^nC_3 \div {}^nC_2 = 8$

$\Rightarrow \frac{n!}{(n-3)!\times 3!} \div \frac{n!}{(n-2)!\times 2!} = 8 \Rightarrow \frac{n!}{(n-3)!\times 3!} \times \frac{(n-2)!\times 2!}{n!} = 8 \Rightarrow \frac{(n-2)!\times 2!}{(n-3)!\times 3!} = 8$

$$\Rightarrow \frac{(n-2)\times(n-3)!\times 2}{(n-3)!\times 6} = 8 \Rightarrow \frac{(n-2)}{3} = 8$$

$\Rightarrow (n-2) = 24 \Rightarrow n = 26$

**(d)**

Given,

$^nP_r = 110$

Again, $^nC_r = 55$

We know that,

$^nP_r = {}^nC_r \times r!$

$$\Rightarrow 110 = 55 * r!$$

$$\Rightarrow r! = \frac{110}{55}$$

$$\Rightarrow r! = 2$$

$$Therefore, r = 2$$

Space for learners:

## CHECK TO YOUR PROGRESS

15. Find the value of n if $^{n}p_4=30*^{n}c_5$

16. If $^{n}C_6:^{n-3}C_3=91:4$, find the value of $n$?

17. If $^{n}C_9=^{n}C_8$ , find $^{n}C_{17}$ .

18. Verify each of the following statement :

(i) $^{5}c_2=^{5}c_3$

(ii) $^{4}C_3\times^{3}c_2=^{12}c_6$

(iii) $^{4}C_2+^{4}C_3=^{8}c_5$

(iv) $^{10}c_2+^{10}c_3=^{11}c_3$

Space for learners:

***Example3***: A group consists of 4 girls and 7 boys. In how many ways can a team of 5 members be selected if the team has (i) no girl ? (ii) at least one boy and one girl ? (iii) at least 3 girls ?

**Solution:**

(i) Since, the team will not include any girl, therefore, only boys are to be selected. 5 boys out of 7 boys can be selected in $^{7}C_5$ ways.

Therefore, the required number of ways $=^{7}C_5$

$$=\frac{7!}{(7-5)!\times5!}=\frac{7!}{2!\times5!}$$

$$=\frac{7\times6\times5!}{2\times5!}$$

$$=\frac{7\times6}{2}=21$$

(*ii*) Since, at least one boy and one girl are to be there in every team. Therefore, the team can consist of

(a) 1 boy and 4 girls (b) 2 boys and 3 girls

(c) 3 boys and 2 girls (d) 4 boys and 1 girl.

Space for learners:

1 boy and 4 girls can be selected in $^7C_1 \times {}^4C_4$ ways.

2 boys and 3 girls can be selected in $^7C_2 \times {}^4C_3$ ways.

3 boys and 2 girls can be selected in $^7C_3 \times {}^4C_2$ ways.

4 boys and 1 girl can be selected in $^7C_4 \times {}^4C_1$ ways.

Therefore,

the required number of ways = $^7C_1 \times {}^4C_4 + {}^7C_2 \times {}^4C_3 + {}^7C_3 \times {}^4C_2 + {}^7C_4 \times {}^4C_1$

$$= 7 + 84 + 210 + 140$$

$$= 441$$

(iii) Since, the team has to consist of at least 3 girls, the team can consist of

(a) 3 girls and 2 boys. Or (b) 4 girls and 1 boy.

Note that the team cannot have all 5 girls, because, the group has only 4 girls

3 girls and 2 boys can be selected in $^4C_3 \times {}^7C_2$ ways.

4 girls and 1 boy can be selected in $^4C_4 \times {}^7C_1$ ways

Therefore, the required number of ways = $^4C_3 \times {}^7C_2 + {}^4C_4 \times {}^7C_1$

= 84+7

= 91

***Example4***: A question paper consists of 10 questions divided into two parts A and B. Each part contains five questions. A candidate is required to attempt 6 questions in all of which at least 2 should be from part A and at least 2 from part B. In how many ways can the candidate select the questions if he can answer all questions equally well?

**Solution**: The candidate has to select six questions in all of which at least two should be from Part A and two should be from Part B. He can select questions in any of the following ways:

| | Part A | Part B |
|---|---|---|
| (i) | 2 | 4 |
| (ii) | 3 | 3 |
| (iii) | 4 | 2 |

If the candidate follows choice (i), the number of ways in which he can do so is

$^5C_2 \times {}^5C_4 = 10 \times 5 = 50$

If the candidate follows choice (ii), the number of ways in which he can do so is

$^5C_3 \times {}^5C_3 = 10 \times 10 = 100$

Similarly, if the candidate follows choice (iii), then the number of ways in which he can do so is

$^5C_4 \times {}^5C_2 = 5 \times 10 = 50$

Therefore, the candidate can select the question in $50 + 100 + 50 = 200$ ways

**Example5:**

In how many ways can a selection of 4 persons be made from 10 persons such that one particular person is always (i) included (ii) excluded

**Solution:** This is the example of Restricted Combination.

(i) The number of ways of selecting 4 persons from 10 persons such that a particular person is always included is $= {}^9C_3$

$$= \frac{9!}{6! \times 3!} = \frac{9 \times 8 \times 7 \times 6!}{6! \times 6} = 84$$

(ii) The number of ways of selecting 4 persons from 10 persons such that a particular person is always excluded is $= {}^9C_4$

$$= \frac{9!}{5! \times 4!} = \frac{9 \times 8 \times 7 \times 6 \times 5!}{5! \times 4 \times 3 \times 2 \times 1} = 126$$

Space for learners:

**Example 6:** A committee of 5 members is to be formed from 6 male teachers and 4 female teachers. How many ways the committee be formed if there be at least one female teacher in the committee?

**Solution:**

The possible selections are as follows:

1 $\quad {}^4c_1 \times {}^6C_4$

2 $\quad {}^4c_2 \times {}^6C_3$

3 $\quad {}^4c_3 \times {}^6C_2$

4 $\quad {}^4c_4 \times {}^6C_1$

Therefore, the total no of Selections

$${}^4c_1 \times {}^6C_4 + {}^4c_2 \times {}^6C_3 + {}^4c_3 \times {}^6C_2 + {}^4c_4 \times {}^6C_1$$

$$= \frac{4!}{3!\times 1!} \times \frac{6!}{4!\times 2!} + \frac{4!}{2!\times 2!} \times \frac{6!}{3!\times 3!} + \frac{4!}{1!\times 3!} \times \frac{6!}{4!\times 2!} + \frac{4!}{0!\times 4!} \times \frac{6!}{5!\times 1!}$$

$$= 4 \times \frac{6\times 5}{2} + \frac{4\times 3}{2} \times \frac{6\times 5\times 4}{3\times 2\times 1} + 4 \times \frac{6\times 5}{2} + 1\times 6 = 60 + 6 \times 20 + 60 + 6 = 246$$

## 1.10 SUMMING UP

- Two integers a and b are congruent modulo n if they each have the same remainder on division by n.
- If $a \equiv b \pmod{n}$, and m is a positive integer, then $a^m \equiv b^m \pmod{n}$ can be termed as the power rule of congruence.
- The least residue of a modulo n is the remainder r that you obtain when you divide a by n.
- Addition, subtraction and multiplication are often simpler to carry out in modular arithmetic than they are normally, because you can use congruences to reduce large numbers to small numbers.
- Permutation is an arrangement in a definite order of a number of objects taken some or all at a time.
- The number of permutations of n different objects taken r at a time, where repetition is allowed, is $n^r$

**Space for learners:**

- Circular Permutation is the total number of ways in which n distinct objects can be arranged around a fix circle.
- Permutation with some specific restrictions is called restricted permutations.
- A combination is a selection of all or part of a set of object without regard to the order in which objects are selected. If there are certain restrictions on Combination like a particular object occurring always and occurring never, then it is called Restricted Combination.

Space for learners:

## 1.11 ANSWERS TO CHECK YOUR PROGRESS

$1(a) True$

$(b) False$

$(c) True$

(d) True

(e) True

(f) False

2. n-=52, 56, 60,……….. .96

3. X=7, 24, 41,58,75,92

4(a) Since, $17 = 1 \times 10 + 7$, the least residue is 7.

(b) Since, $50 = 5 \times 10 + 0$, the least residue is 0.

(c) Since, $6 = 0 \times 10 + 6$, the least residue is 6.

(d) Since, $-1 = (-1) \times 10 + 9$, the least residue is 9.

(e) Since, $-38 = (-4) \times 10 + 2$, the least residue is 2.

5(a) $3 \times 6 \equiv 18 \equiv 4 \pmod 7$ So the least residue is 4.

(b) $22 \times 29 \equiv 1 \times 1 \equiv 1 \pmod 7$,So the least residue is 1

(c) $51 \times 74 \equiv 2 \times 4 \equiv 8 \equiv 1 \pmod 7$,so least residue is 1

6. We have, $32 \equiv -9 \pmod{41}$

$\Rightarrow 2^5 \equiv -9 (mod\ 41)$

$\Rightarrow (2^5)^2 \equiv (-9)^2 (mod\ 41)$ $[By\ Power\ rule]$ $Again,\ \ (-9)2 = 81 \equiv\ -1\ (mod\ 41)$

$i.e., (2^5)^2 \equiv (-1)(mod\ 41)$

$Similarly, \{(2^5)^2\}^2 \equiv (-1)^2 (mod\ 41)[By\ power\ rule]$

$i.e., 2^{20} \equiv 1 (mod\ 41)$

7. We notice that $3^4 = 81 \equiv 1$ (mod 80).

That is, we have $3^4 \equiv 1$ (mod 80)

By power rule of Congruence,

$(3^4)^{1388} \equiv 1$ (mod80). $[since, 3^{5555} = (3^4)^{1388} . 3^3]$

Therefore, $(3^4)^{1388} . 3^3 \equiv 3^3$ (mod80).

Therefore, $3^{5555} \equiv 27$ (mod80).

So, the required remainder is 27

8(a) (i) 720

(ii) 5040

(iii) 5046

(iv) 17280

8(b) (i) False

(ii)False

(iii) False

9 (a) 32760

(b) 55440

(c) 1

10. n=5

11. r=5

Space for learners:

12. r=8, 3

13.24

14. 120

15. 8

16. 15

17. 1

18(i) True

(ii) False

(iii) False

(iv) True

## 1.12 POSSIBLE QUESTIONS

Question1: Find the least residue of 1492 (mod 4), (mod 10) .

Question2 : Does 33x ≡ 12 (mod 6) imply 11x ≡ 4 (mod 6) ? Why ?

Question3: What is the remainder when $2^{50}$ *is divided by* 7?

*Question*4: Find the least residue of $11^6$ modulo 9.

Question5: Check whether 4 and 6 are congruent modulo 5 or not?

Question6. Every integer a is congruent modulo 1 to every integer b. State True or False.

Question 7: If ${}^nP_r = {}^nP_{r+1}$ and ${}^nC_r = {}^nC_{r-1}$, find the value of n and r.

Question 8: If ${}^nC_{r-1} = 36$, ${}^nC_r = 84$ and ${}^nC_{r+1} = 126$, find the value of n and r?

Question 9: How many numbers less than 1000 can be formed using the digits 0,1,2,3,4,5,6 if repetition of digits being allowed?

Question 10: In how many ways can 3 boys and 4 girls be arranged so that no two boys will be side by side?

***Space for learners:***

Space for learners:

Question 11: Find the number of numbers greater than 4000 which can be formed using the digits 0, 2, 4, 6, 8 without repetition.

Question 12: 9 different letters of an alphabet are given. Find the number of 4 letter words that can be formed using these 9 letters which have (i) no letter repeated (ii) atleast one letter repeated.

Question 13: Find the number of ways of arranging the letters of the word.

(a) INDEPENDENCE (b) MATHEMATICS

Question 14: In how many ways 6 books be put into 5 bags?

Question 15: How many ways 3 students can be selected from 50 students?

Question 16: In how many ways can the letters in the word ENGINEERINGis arranged such that no two E's are together?

Question 17: A cricket team consisting of 11 players is to be selected from 6 bowlers and 8 batsman including at least 4 bowlers. In how many ways can this be done?

Question 18: There are 5 black and 6 red balls in a bag. How many selection can be made taking 2 black and 3 red balls?

## 1.12 REFERENCES AND SUGGESTED READINGS


- Permutation and Combination by Ramesh Chandra
- http:// www.wikipedia.org
- http://mathworld.wolfram.com

# UNIT 2: SETS

Space for learners:

**Unit Structure:**



## 2.1 INTRODUCTION

In mathematics, a set is a collection of elements. The elements that make up a set can be any kind of mathematical objects: numbers, symbols, points in space, lines, other geometrical shapes, variables, or even other sets. The set with no element is the empty set; a set with a single element is a singleton. A set may have a finite number of elements or be an infinite set. Two sets are equal if and only if they have precisely the same elements.

Sets are ubiquitous in modern mathematics. Indeed, set theory, more specifically Zermelo–Fraenkel set theory, has been the standard way to provide rigorous foundations for all branches of mathematics since the first half of the 20th century.

The concept of a set emerged in mathematics at the end of the 19th century. The German word for set, Menge, was coined by Bernard Bolzano in his work Paradoxes of the Infinite.

A set is a gathering together into a whole of definite, distinct objects of our perception or our thought—which are called elements of the set.

Bertrand Russell called a set a class: "When mathematicians deal with what they call a manifold, aggregate, Menge, ensemble, or some equivalent name, it is common, especially where the number of terms involved is finite, to regard the object in question (which is in fact a class) as defined by the enumeration of its terms, and as consisting possibly of a single term, which is in that case is the class."

Space for learners:

## 2.2 UNIT OBJECTIVES

This unit will give you about the

- idea of set
- definition of sets
- representation sets
- operations of sets
- questions and answers for your progress

## 2.3 DEFINITIONS OF SETS

**Set - Definition**

A set is an unordered collection of different elements. A set can be written explicitly by listing its elements using set bracket. If the order of the elements is changed or any element of a set is repeated, it does not make any changes in the set.

**Some Example of Sets**

- A set of all positive integers
- A set of all the planets in the solar system
- A set of all the states in India
- A set of all the lowercase letters of the alphabet

**Representation of a Set**

Sets can be represented in two ways −

- Roster or Tabular Form
- Set Builder Notation

**Roster or Tabular Form**

The set is represented by listing all the elements comprising it. The elements are enclosed within braces and separated by commas.

Space for learners:

Example 1 Set of vowels in English alphabet, A={a,e,i,o,u}.

Example 2 Set of odd numbers less than 10, B={1,3,5,7,9}.

**Set Builder Notation**

The set is defined by specifying a property that elements of the set have in common. The set is described as A={x:p(x)}A={x:p(x)}

Example 1 The set {a,e,i,o,u}{a,e,i,o,u} is written as

A={x:x is a vowel in English alphabet}.

Example 2 The set {1,3,5,7,9} is written as

B={x:1≤x<10 and (x%2)≠0}.

If an element x is a member of any set S, it is denoted by x∈S and if an element y is not a member of set S, it is denoted by y∉S.

Example If S={1,1.2,1.7,2},1∈S, but 1.5∉S.

**Some Important Sets**

N − the set of all natural numbers = {1,2,3,4,.....}

Z − the set of all integers = {.....,−3,−2,−1,0,1,2,3,.....}

Z+ − the set of all positive integers

Q − the set of all rational numbers

R − the set of all real numbers

W − the set of all whole numbers

**Cardinality of a Set**

Cardinality of a set S, denoted by |S|, is the number of elements of the set. The number is also referred as the cardinal number. If a set has an infinite number of elements, its cardinality is ∞.

Example − |{1,4,3,5}|=4,|{1,2,3,4,5,…}|=∞|. If there are two sets X and Y,

- |X|=|Y| denotes two sets X and Y having same cardinality. It occurs when the number of elements in X is exactly equal to the number of elements in Y. In this case, there exists a bijective function 'f' from X to Y.
- |X|≤|Y| denotes that set X's cardinality is less than or equal to set Y's cardinality. It occurs when number of elements in X is

less than or equal to that of Y. Here, there exists an injective function 'f' from X to Y.

- |X|<|Y| denotes that set X's cardinality is less than set Y's cardinality. It occurs when number of elements in X is less than that of Y. Here, the function 'f' from X to Y is injective function but not bijective.
- If |X|≤|Y| and |X|≥|Y| then |X|=|Y|. The sets X and Y are commonly referred as equivalent sets.

**Types of Sets**

Sets can be classified into many types. Some of which are finite, infinite, subset, universal, proper, singleton set, etc.

**Finite Set**

A set which contains a definite number of elements is called a finite set.

Example − S={x|x∈N and 70>x>50}

**Infinite Set**

A set which contains infinite number of elements is called an infinite set.

Example − S={x|x∈N and x>10}

**Subset**

A set X is a subset of set Y (Written as X⊆Y) if every element of X is an element of set Y.

Example 1  Let, X={1,2,3,4,5,6} and Y={1,2}. Here set Y is a subset of set X as all the elements of set Y is in set X. Hence, we can write Y⊆X.

Example 2 − Let, X={1,2,3} and Y={1,2,3}. Here set Y is a subset (Not a proper subset) of set X as all the elements of set Y is in set X. Hence, we can write Y⊆X.

**Proper Subset**

The term "proper subset" can be defined as "subset of but not equal to". A Set X is a proper subset of set Y (Written as X⊂Y) if every element of X is an element of set Y and |X|<|Y|.

*__Space for learners:__*

Example − Let, X={1,2,3,4,5,6} and Y={1,2}. Here set Y⊂X since all elements in Y are contained in X too and X has at least one element is more than set Y.

**Universal Set**

It is a collection of all elements in a particular context or application. All the sets in that context or application are essentially subsets of this universal set. Universal sets are represented as U.

Example: We may define U as the set of all animals on earth. In this case, set of all mammals is a subset of U, set of all fishes is a subset of U, set of all insects is a subset of U, and so on.

**Empty Set or Null Set**

An empty set contains no elements. It is denoted by ∅. As the number of elements in an empty set is finite, empty set is a finite set. The cardinality of empty set or null set is zero.

Example − S={x|x∈NS={x|x∈N and 7<x<8}=∅

**Singleton Set or Unit Set**

Singleton set or unit set contains only one element. A singleton set is denoted by {s}.

Example − S={x|x∈N, 7<x<9}= {8}

**Equal Set**

If two sets contain the same elements they are said to be equal.

Example − If A={1,2,6} and B={6,1,2}, they are equal as every element of set A is an element of set B and every element of set B is an element of set A.

**Equivalent Set**

If the cardinalities of two sets are same, they are called equivalent sets.

Example  If A={1,2,6} and B={16,17,22}, they are equivalent as cardinality of A is equal to the cardinality of B. i.e. |A|=|B|=3.

**Disjoint Set**

Two sets A and B are called disjoint sets if they do not have even one element in common. Therefore, disjoint sets have the following properties –

***Space for learners:***

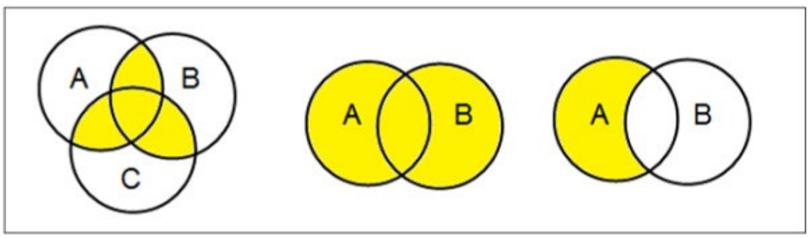
- $n(A \cap B) = \emptyset$
- $n(A \cup B) = n(A) + n(B)$

Example Let, A={1,2,6} and B={7,9,14}, there is not a single common element, hence these sets are overlapping sets.

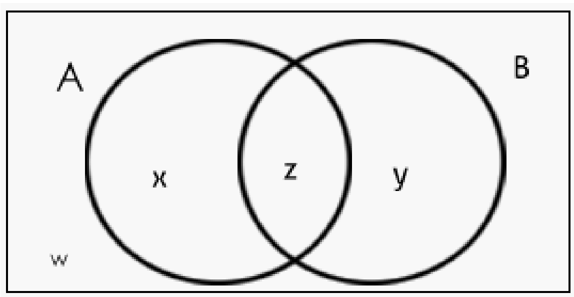
**Venn Diagrams**

Venn diagram, invented in 1880 by John Venn, is a schematic diagram that shows all possible logical relations between different mathematical sets.

Examples




Venn Diagram in case of two elements




Where;

X = number of elements that belong to set A only
Y = number of elements that belong to set B only
Z = number of elements that belong to set A and B both $(A \cap B)$
W = number of elements that belong to none of the sets A or B

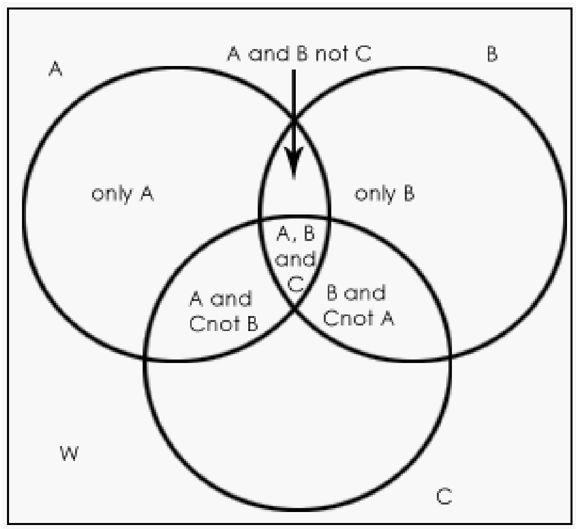
From the above figure, it is clear that

$n(A) = x + z$ ;
$n(B) = y + z$ ;
$n(A \cap B) = z$;
$n(A \cup B) = x + y + z$.

Total number of elements = $x + y + z + w$

Space for learners:

Venn Diagram in case of three elements




Where,

W = number of elements that belong to none of the sets A, B or C

Example 1: In a college, 200 students are randomly selected. 140 like tea, 120 like coffee and 80 like both tea and coffee.

How many students like only tea?

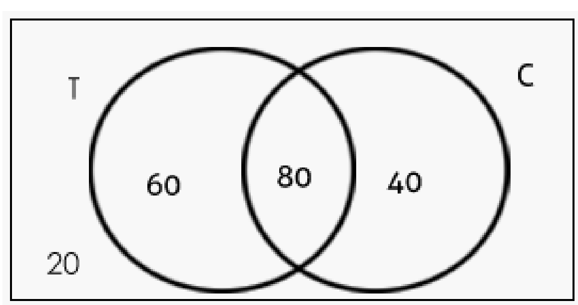
How many students like only coffee?

How many students like neither tea nor coffee?

How many students like only one of tea or coffee?

How many students like at least one of the beverages?

Solution: The given information may be represented by the following Venn diagram, where T = tea and C = coffee.




Number of students who like only tea = 60

**_Space for learners:_**

Number of students who like only coffee = 40

Number of students who like neither tea nor coffee = 20

Number of students who like only one of tea or coffee = 60 + 40 = 100

Number of students who like at least one of tea or coffee = n (only Tea) + n (only coffee) + n (both Tea & coffee) = 60 + 40 + 80 = 180

Example 2: In a survey of 500 students of a college, it was found that 49% liked watching football, 53% liked watching hockey and 62% liked watching basketball. Also, 27% liked watching football and hockey both, 29% liked watching basketball and hockey both and 28% liked watching football and basket ball both. 5% liked watching none of these games.

***How many students like watching all the three games?***

Find the ratio of number of students who like watching only football to those who like watching only hockey.

Find the number of students who like watching only one of the three given games.

Find the number of students who like watching at least two of the given games.

***Solution:***

n(F) = percentage of students who like watching football = 49%

n(H) = percentage of students who like watching hockey = 53%

n(B)= percentage of students who like watching basketball = 62%

$n(F \cap H) = 27\%$ ; $n(B \cap H) = 29\%$ ; $n(F \cap B) = 28\%$

Since 5% like watching none of the given games so, $n(F \cup H \cup B) = 95\%$.

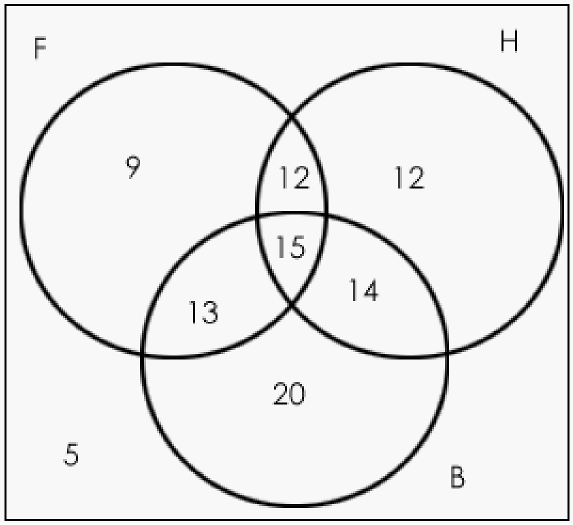
Now applying the basic formula,

$95\% = 49\% + 53\% + 62\% - 27\% - 29\% - 28\% + n(F \cap H \cap B)$

Solving, you get $n(F \cap H \cap B) = 15\%$.

Now, make the Venn diagram as per the information given.

***Space for learners:***




Note: All values in the Venn diagram are in percentage.

Number of students who like watching all the three games = 15 % of 500 = 75.

Ratio of the number of students who like only football to those who like only hockey = (9% of 500)/(12% of 500) = 9/12 = 3:4.

The number of students who like watching only one of the three given games = (9% + 12% + 20%) of 500 = 205

The number of students who like watching at least two of the given games=(number of students who like watching only two of the games) +(number of students who like watching all the three games)= (12 + 13 + 14 + 15)% i.e. 54% of 500 = 270.

**CHECK YOUR PROGRESS**

**Q1 :Which of the following are sets? Justify our answer.**

(i) The collection of all months of a year beginning with the letter J.

(ii) The collection of ten most talented writers of India.

(iii) A team of eleven best-cricket batsmen of the world.

(iv) The collection of all boys in your class.

(v) The collection of all natural numbers less than 100.

(vi) A collection of novels written by the writer Munshi Prem Chand.

***Space for learners:***

(vii) The collection of all even integers.

(viii) The collection of questions in this Chapter.

(ix) A collection of most dangerous animals of the world.

**Q2. Let A = {1, 2, 3, 4, 5, 6}. Insert the appropriate symbol or in the blank spaces:**

(i) 5…A (ii) 8…A (iii) 0…A

(iv) 4…A (v) 2…A (vi) 10…A

**Q3 :Write the following sets in the set-builder form:**

(i) (3, 6, 9, 12) (ii) {2, 4, 8, 16, 32}

(iii) {5, 25, 125, 625} (iv) {2, 4, 6 …}

(v) {1, 4, 9 … 100}

**Q4. Which of the following are examples of the null set**

(i) Set of odd natural numbers divisible by 2

(ii) Set of even prime numbers

(iii) {x:x is a natural numbers, x < 5 and x > 7 }

(iv) {y:y is a point common to any two parallel lines}

**Q5. Which of the following sets are finite or infinite**

(i) The set of months of a year

(ii) {1, 2, 3 ...}

(iii) {1, 2, 3 ... 99, 100}

(iv) The set of positive integers greater than 100

(v) The set of prime numbers less than 99

**Q6. State whether each of the following set is finite or infinite:**

(i) The set of lines which are parallel to the x-axis

(ii) The set of letters in the English alphabet

(iii) The set of numbers which are multiple of 5

(iv) The set of animals living on the earth

(v) The set of circles passing through the origin (0, 0)

***Space for learners:***

**Q7. In the following, state whether A = B or not:**

(i) A = {a, b, c, d}; B = {d, c, b, a}

(ii) A = {4, 8, 12, 16}; B = {8, 4, 16, 18}

(iii) A = {2, 4, 6, 8, 10}; B = {x: x is positive even integer and x ≤ 10} (iv) A = {x: x is a multiple of 10}; B = {10, 15, 20, 25, 30 ...}

**Q8. Are the following pair of sets equal? Give reasons.**

(i) A = {2, 3}; B = {x: x is solution of $x^2 + 5x + 6 = 0$}

(ii) A = {x: x is a letter in the word FOLLOW}; B = {y: y is a letter in the word WOLF}

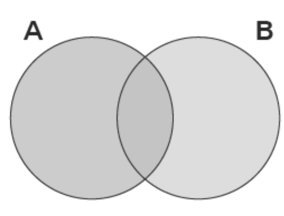
*Space for learners:*

## 2.4 OPERATIONS OF SETS

A set is defined as a collection of objects. Each object inside a set is called an 'Element'. A set can be represented in three forms. They are statement form, roster form, and set builder form. Set operations are the operations that are applied on two more sets to develop a relationship between them. There are four main kinds of set operations which are:

1. Union of sets
2. Intersection of sets
3. Complement of a set
4. Difference between sets/Relative Complement

Before we move on to discuss the various set operations, let us recall the concept of Venn diagrams as it is important in understanding the operations on sets. A Venn diagram is a logical diagram that shows the possible relationship between different finite sets. It looks as shown below.




**Basic Set Operations:**

Space for learners:

Now that we know the concept of a set and Venn diagram, let us discuss each set operation one by one in detail. The various set operations are:

**Union of Sets**

For two given sets A and B, $A \cup B$ (read as A union B) is the set of distinct elements that belong to set A and B or both. The number of elements in $A \cup B$ is given by $n(A \cup B) = n(A) + n(B) - n(A \cap B)$, where n(X) is the number of elements in set X. To understand this set operation of the union of sets better, let us consider an example: If A = {1, 2, 3, 4} and B = {4, 5, 6, 7}, then the union of A and B is given by $A \cup B$ = {1, 2, 3, 4, 5, 6, 7}.

**Intersection of Sets**

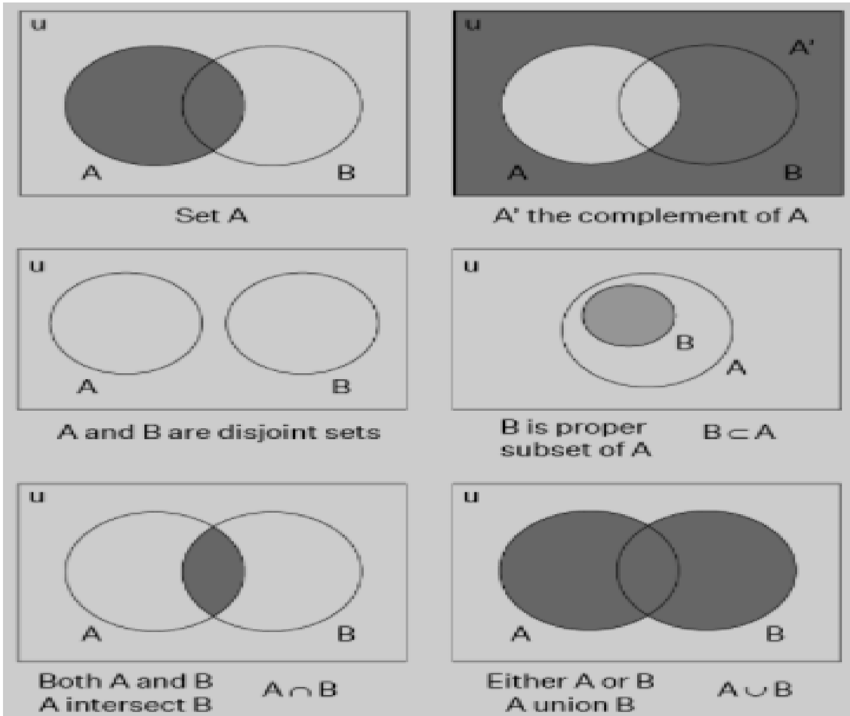
For two given sets A and B, $A \cap B$ (read as A intersection B) is the set of common elements that belong to set A and B. The number of elements in $A \cap B$ is given by $n(A \cap B) = n(A)+n(B)-n(A \cup B)$, where n(X) is the number of elements in set X. To understand this set operation of the intersection of sets better, let us consider an example: If A = {1, 2, 3, 4} and B = {3, 4, 5, 7}, then the intersection of A and B is given by $A \cap B$ = {3, 4}.

**Set Difference**

The set operation difference between sets implies subtracting the elements from a set which is similar to the concept of the difference between numbers. The difference between sets A and B denoted as $A - B$ lists all the elements that are in set A but not in set B. To understand this set operation of set difference better, let us consider an example: If A = {1, 2, 3, 4} and B = {3, 4, 5, 7}, then the difference between sets A and B is given by A - B = {1, 2}.

**Complement of Sets**

The complement of a set A denoted as A′ or Ac (read as A complement) is defined as the set of all the elements in the given universal set(U) that are not present in set A. To understand this set operation of complement of sets better, let us consider an example: If U = {1, 2, 3, 4, 5, 6, 7, 8, 9} and A = {1, 2, 3, 4}, then the complement of set A is given by A' = {5, 6, 7, 8, 9}.




Space for learners:

The above image shows various set operations with the help of Venn diagrams which makes it more clear. When the elements of one set B completely lie in the other set A, then B is said to be a proper subset of A. When two sets have no elements in common, then they are said to be disjoint sets. Now, let us explore the properties of the set operations that we discussed.

**Properties of Set Operations:**

The properties of set operations are similar to the properties of fundamental operations on numbers. The important properties on set operations are stated below:

**Commutative Law -** For any two given sets A and B, the commutative property is defined as,

$A \cup B = B \cup A$

This means that the set operation union of two sets is commutative.

$A \cap B = B \cap A$

This means that the set operation intersection of two sets is commutative.

**Associative Law -** For any three given sets A, B and C the associative property is defined as,

$(A \cup B) \cup C = A \cup (B \cup C)$

This means the set operation union of sets is associative.

$(A \cap B) \cap C = A \cap (B \cap C)$

This means the set operation intersection of sets is associative.

**De-Morgan's Law -** The law states that for any two sets A and B, we have $(A \cup B)' = A' \cap B'$ and $(A \cap B)' = A' \cup B'$

$A \cup A = A$

$A \cap A = A$

$A \cap \emptyset = \emptyset$

$A \cup \emptyset = A$

$A \cap B \subseteq A$

$A \cup B \subseteq A$

**Important Notes on Set Operations:**

Set operation formula for union of sets is $n(A\cup B) = n(A) + n(B) - n(A\cap B)$ and set operation formula for intersection of sets is $n(A\cap B) = n(A)+n(B)-n(A\cup B)$.

The union of any set with the universal set gives the universal set and the intersection of any set A with the universal set gives the set A.

Union, intersection, difference, and complement are the various operations on sets.

The complement of a universal set is an empty set $U' = \phi$. The complement of an empty set is a universal set $\phi' = U$.

**CHECK YOUR PROGRESS**

9. Let A={1,2,3,4} and let B={3,4,5,6} . Then: find $A\cap B$, $A\cup B$ $A-B$ and $A^C$

10. Let A={y,z} and let B={x,y,z} . Then: find $A\cap B$, $A\cup B$, $A-B$, and $A^C$

11. If A = {2, 3, 4, 5} B = {4, 5, 6, 7} C = {6, 7, 8, 9} D = {8, 9, 10, 11}, find

*Space for learners:*

(a) $A \cup B$

(b) $A \cup C$

(c) $B \cup C$

(d) $B \cup D$

(e) $(A \cup B) \cup C$

(f) $A \cup (B \cup C)$

(g) $B \cup (C \cup D)$

12. If A = {4, 7, 10, 13, 16, 19, 22}   B = {5, 9, 13, 17, 20}

C = {3, 5, 7, 9, 11, 13, 15, 17}   D = {6, 11, 16, 21} then find

(a) A - C

(b) D - A

(c) D - B

(d) A - D

(e) B - C

(f) C - D

(g) B - A

(h) B - D

(i) D - C

(j) A - B

(k) C - B

(l) C - A

## 2.5 SUMMING UP

- A set can be written explicitly by listing its elements using set bracket. If the order of the elements is changed or any element of a set is repeated, it does not make any changes in the set.
- Sets can be represented in two ways − Roster or Tabular Form, Set Builder Notation

***Space for learners:***

Space for learners:

- Cardinality of a set S, denoted by |S|, is the number of elements (cardinal number) of the set. If a set has an infinite number of elements, its cardinality is $\infty$.
- A set which contains a definite number of elements is called a finite set. A set which contains infinite number of elements is called an infinite set.
- A set X is a subset of set Y (Written as $X \subseteq Y$) if every element of X is an element of set Y. A Set X is a proper subset of set Y (Written as $X \subset Y$) if every element of X is an element of set Y and $|X|<|Y|$.
- Universal set is a collection of all elements in a particular context or application. The cardinality of empty set or null set is zero. Singleton set or unit set contains only one element.
- If two sets contain the same elements they are said to be equal. If the cardinalities of two sets are same, they are called equivalent sets. Disjoint sets do not have even one element in common.

5. Set operations are the operations applied on two more sets to develop a relationship between them. There are four main kinds of set operations: Union of sets, Intersection of sets, Complement of a set, Difference between sets/Relative Complement.

## 2.6 ANSWERS TO CHECK YOUR PROGRESS

**Answer 1:**

(i) The collection of all months of a year beginning with the letter J is a well-defined collection of objects

because one can definitely identify a month that belongs to this collection.

Hence, this collection is a set.

(ii) The collection of ten most talented writers of India is not a well-defined collection because the criteria for

determining a writer's talent may vary from person to person.

Hence, this collection is not a set.

(iii) A team of eleven best cricket batsmen of the world is not a well-defined collection because the criteria for determining a batsman's talent may vary from person to person.

Hence, this collection is not a set.

(iv) The collection of all boys in your class is a well-defined collection because you can definitely identify a boy

who belongs to this collection.

Hence, this collection is a set.

(v) The collection of all natural numbers less than 100 is a well-defined collection because one can definitely

identify a number that belongs to this collection.

Hence, this collection is a set.

(vi) A collection of novels written by the writer Munshi Prem Chand is a well-defined collection because one can

definitely identify a book that belongs to this collection.

Hence, this collection is a set.

(vii) The collection of all even integers is a well-defined collection because one can definitely identify an even

integer that belongs to this collection.

Hence, this collection is a set.

(viii) The collection of questions in this chapter is a well-defined collection because one can definitely identify a

question that belongs to this chapter.

Hence, this collection is a set.

(ix) The collection of most dangerous animals of the world is not a well-defined collection because the criteria

for determining the dangerousness of an animal can vary from person to person.

Hence, this collection is not a set.

**Answer 2:**

(i) 5 A

(ii) 8 A

(iii) 0 A

(iv) 4 A

***Space for learners:***

(v) 2 A

(vi) 10 A

**Answer 3:**

(i) {3, 6, 9, 12} = {x: x = 3n, n N and 1 ≤ n ≤ 4}

(ii) {2, 4, 8, 16, 32}

It can be seen that 2 = 21, 4 = 22, 8 = 23, 16 = 24, and 32 = 25.

{2, 4, 8, 16, 32} = {x: x = 2n, n N and 1 ≤ n ≤ 5}

(iii) {5, 25, 125, 625}

It can be seen that 5 = 51, 25 = 52, 125 = 53, and 625 = 54.

{5, 25, 125, 625} = {x: x = 5n, n N and 1 ≤ n ≤ 4}

(iv) {2, 4, 6 …}

It is a set of all even natural numbers.

{2, 4, 6 …} = {x: x is an even natural number}

(v) {1, 4, 9 … 100}

It can be seen that 1 = 12, 4 = 22, 9 = 32 …100 = 102.

{1, 4, 9… 100} = {x: x = n2, n N and 1 ≤ n ≤ 10}

**Answer 4:**

(i) A set of odd natural numbers divisible by 2 is a null set because no odd number is divisible by 2.

(ii) A set of even prime numbers is not a null set because 2 is an even prime number.

(iii) {x: x is a natural number, x < 5 and x > 7} is a null set because a number cannot be simultaneously less

than 5 and greater than 7.

(iv) {y: y is a point common to any two parallel lines} is a null set because parallel lines do not intersect. Hence,

they have no common point.

**Answer 5:**

(i) The set of months of a year is a finite set because it has 12 elements.

***Space for learners:***

(ii) {1, 2, 3 …} is an infinite set as it has infinite number of natural numbers.

(iii) {1, 2, 3 …99, 100} is a finite set because the numbers from 1 to 100 are finite in number.

(iv) The set of positive integers greater than 100 is an infinite set because positive integers greater than 100 are

infinite in number.

(v) The set of prime numbers less than 99 is a finite set because prime numbers less than 99 are finite in number.

**Answer 6:**

(i) The set of lines which are parallel to the x-axis is an infinite set because lines parallel to the x-axis are infinite

in number.

(ii) The set of letters in the English alphabet is a finite set because it has 26 elements.

(iii) The set of numbers which are multiple of 5 is an infinite set because multiples of 5 are infinite in number.

(iv) The set of animals living on the earth is a finite set because the number of animals living on the earth is finite

(although it is quite a big number).

(v) The set of circles passing through the origin (0, 0) is an infinite set because infinite number of circles can pass

through the origin.

**Answer 7:**

(i) A = {a, b, c, d}; B = {d, c, b, a}

The order in which the elements of a set are listed is not significant.

A = B

(ii) A = {4, 8, 12, 16}; B = {8, 4, 16, 18} It can be seen that 12 A but 12 B.

$A \neq B$

(iii) A = {2, 4, 6, 8, 10}

B = {x: x is a positive even integer and $x \leq 10$}

= {2, 4, 6, 8, 10}

***Space for learners:***

Therefore, A = B

**Answer 8:**

(i) A = {2, 3}; B = {x: x is a solution of x2 + 5x + 6 = 0}

The equation x2 + 5x + 6 = 0 can be solved as:

x(x + 3) + 2(x + 3) = 0 (x + 2)(x + 3) = 0

x = -2 or x = -3

A = {2, 3}; B = {-2, -3}

$A \neq B$

(ii) A = {x: x is a letter in the word FOLLOW} = {F, O, L, W}

B = {y: y is a letter in the word WOLF} = {W, O, L, F}

The order in which the elements of a set are listed is not significant.

A = B

**Answer 9:**

A∩B={3,4}

A∪B={1,2,3,4,5,6}

A−B={1,2}

$A^C$={all real numbers except 1,2,3 and 4}

**Answer 10:** A∩B={y,z} A∪B={x,y,z}A−B=∅ $A^c$={everything except y and z}

**Answer 11:**

(a) {2, 3, 4, 5, 6, 7}

(b) {2, 3, 4, 5, 6, 7, 8, 9}

(c) {4, 5, 6, 7, 8, 9}

(d) {4, 5, 6, 7, 8, 9, 10, 11}

(e) {2, 3, 4, 5, 6, 7, 8, 9}

(f) {2, 3, 4, 5, 6, 7, 8, 9}

(g) {4, 5, 6, 7, 8, 9, 10, 11}

**Answer 12:**

(a) {4, 10, 16, 19, 22}

(b) {6, 11, 21}

***Space for learners:***

(c) {6, 11, 16, 21}

(d) {4, 7, 10, 13, 19, 22}

(e) {20}

(f) {3, 5, 7, 9, 13, 15, 17}

(g) {5, 19, 17, 20}

(h) {5, 9, 13, 17, 20}

(i) {6, 16, 21}

(j) {4, 7, 10, 16, 19, 22}

(k) {3, 7, 11, 15}

(l) {3, 5, 9 11, 15, 17}

## 2.7 POSSIBLE QUESTIONS

**1.** If A = {2, 3, 4, 5}   B = {4, 5, 6, 7}   C = {6, 7, 8, 9}

D = {8, 9, 10, 11}, find

(a) A ∪ B

(b) A ∪ C

(c) B ∪ C

(d) B ∪ D

(e) (A ∪ B) ∪ C

(f) A ∪ (B ∪ C)

(g) B ∪ (C ∪ D)

**2.** If A = {4, 6, 8, 10, 12} B = {8, 10, 12, 14} C = {12, 14, 16} D = {16, 18}, find

(a) A ∩ B

(b) B ∩ C

(c) A ∩ (C ∩ D)

(d) A ∩ C

(e) B ∩ D

(f)(A ∩ B) ∪ C

(g) A ∩ (B ∪ D)

*Space for learners:*

(h) $(A \cap B) \cup (B \cap C)$

(i) $(A \cup D) \cap (B \cup C)$

**3.** If A = {4, 7, 10, 13, 16, 19, 22}   B = {5, 9, 13, 17, 20}

C = {3, 5, 7, 9, 11, 13, 15, 17}   D = {6, 11, 16, 21} then find

(a) A - C

(b) D - A

(c) D - B

(d) A - D

(e) B - C

(f) C - D

(g) B - A

(h) B - D

(i) D - C

(j) A - B

(k) C - B

(l) C - A

**4.** If A and B are two sets such that $A \subset B$, then what is $A \cup B$?

**5.** Find the union, intersection and the difference (A - B) of the following pairs of sets.

(a) A = The set of all letters of the word FEAST

B = The set of all letters of the word TASTE

(b) $A = \{x : x \in W, 0 < x \leq 7\}$

$B = \{x : x \in W, 4 < x < 9\}$

(c) $A = \{x \mid x \in N$, x is a factor of $12\}$

$B = \{x \mid x \in N$, x is a multiple of $2, x < 12\}$

(d) A = The set of all even numbers less than 12

B = The set of all odd numbers less than 11

(e) $A = \{x : x \in I, -2 < x < 2\}$

$B = \{x : x \in I, -1 < x < 4\}$

***Space for learners:***

(f) $A = \{a, l, m, n, p\}$

$B = \{q, r, l, a, s, n\}$

**6.** Let $X = \{2, 4, 5, 6\}$ $Y = \{3, 4, 7, 8\}$ $Z = \{5, 6, 7, 8\}$, find

(a) $(X - Y) \cup (Y - X)$

(b) $(X - Y) \cap (Y - X)$

(c) $(Y - Z) \cup (Z - Y)$

(d) $(Y - Z) \cap (Z - Y)$

**7.** Let $\xi = \{1, 2, 3, 4, 5, 6, 7\}$ and $A = \{1, 2, 3, 4, 5\}$ $B = \{2, 5, 7\}$ show that

(a) $(A \cup B)' = A' \cap B'$

(b) $(A \cap B)' = A' \cup B'$

(c) $(A \cap B) = B \cap A$

(d) $(A \cup B) = B \cup A$

**8.** Let $P = \{a, b, c, d\}$ $Q = \{b, d, f\}$ $R = \{a, c, e\}$ verify that

(a) $(P \cup Q) \cup R = P \cup (Q \cup R)$

(b) $(P \cap Q) \cap R = P \cap (Q \cap R)$

## 2.8 REFERENCES AND SUGGESTED READINGS


- Descriptive Set Theory by David Marker.
- Set Theory by Burak Kaya
- Set Theory Some Basics And A Glimpse Of Some Advanced Techniques
- Lectures On Set Theory
- Set Theory by Anush Tserunyan
- An Introduction To Set Theory
- Set Theory for Computer Science
- The Axioms of Set Theory.


**Space for learners:**

# UNIT 3: RELATIONS

Space for learners:

**Unit Structure:**



## 3.1 INTRODUCTION

In mathematics, a binary relation over sets X and Y is a subset of the Cartesian product X × Y; that is, it is a set of ordered pairs (x, y) consisting of elements x in X and y in Y. It encodes the common concept of relation: an element x is related to an element y, if and only if the pair (x, y) belongs to the set of ordered pairs that defines the binary relation. A binary relation is the most studied special case n = 2 of an n-ary relation over sets $X_1, ..., X_n$, which is a subset of the Cartesian product $X_1 \times ... \times X_n$.[1][2]

An example of a binary relation is the "divides" relation over the set of prime numbers {\displaystyle \mathbb {P} }\mathbb {P} and the set of integers {\displaystyle \mathbb {Z} }\mathbb {Z} , in which each prime p is related to each integer z that is a multiple of p, but not to an integer that is not a multiple of p. In this relation, for instance, the prime number 2 is related to numbers such as −4, 0, 6, 10, but not to 1 or 9, just as the prime number 3 is related to 0, 6, and 9, but not to 4 or 13.

Binary relations are used in many branches of mathematics to model a wide variety of concepts. These include, among others:

the "is greater than", "is equal to", and "divides" relations in arithmetic;

the "is congruent to" relation in geometry;

the "is adjacent to" relation in graph theory;

the "is orthogonal to" relation in linear algebra.

A function may be defined as a special kind of binary relation. Binary relations are also heavily used in computer science.

A binary relation over sets X and Y is an element of the power set of X × Y. Since the latter set is ordered by inclusion (⊆), each relation has a place in the lattice of subsets of X × Y. A binary relation is either a homogeneous relation or a heterogeneous relation depending on whether X = Y or not.

Since relations are sets, they can be manipulated using set operations, including union, intersection, and complementation, and satisfying the laws of an algebra of sets. Beyond that, operations like the converse of a relation and the composition of relations are available, satisfying the laws of a calculus of relations, for which there are textbooks by Ernst Schröder,[4] Clarence Lewis, and Gunther Schmidt. A deeper analysis of relations involves decomposing them into subsets called concepts, and placing them in a complete lattice.

In some systems of axiomatic set theory, relations are extended to classes, which are generalizations of sets. This extension is needed for, among other things, modeling the concepts of "is an element of" or "is a subset of" in set theory, without running into logical inconsistencies such as Russell's paradox.

The terms correspondence, dyadic relation and two-place relation are synonyms for binary relation, though some authors use the term "binary relation" for any subset of a Cartesian product X × Y without reference to X and Y, and reserve the term "correspondence" for a binary relation with reference to X and Y.

## 3.2 UNIT OBJECTIVES

In going through this unit, you will be able to:

- learn about the relations.
- learn types of relations

***Space for learners:***

- understand closure properties of relations
- know the equivalence and partial order of relations
- know the basis of functions.

Space for learners:

## 3.3 TYPES OF RELATION

**Binary Relation**

Let P and Q be two non- empty sets. A binary relation R is defined to be a subset of P x Q from a set P to Q. If (a, b) ∈ R and R ⊆ P x Q then a is related to b by R i.e., aRb. If sets P and Q are equal, then we say R ⊆ P x P is a relation on P e.g.

(i) Let A = {a, b, c}

B = {r, s, t}

Then R = {(a, r), (b, r), (b, t), (c, s)} is a relation from A to B.

(ii) Let A = {1, 2, 3} and B = A

R = {(1, 1), (2, 2), (3, 3)} is a relation (equal) on A.

**Example:** If a set has n elements, how many relations are there from A to A.

**Solution:** If a set A has n elements, A x A has $n^2$ elements. So, there are $2^{n2}$ relations from A to A.

**Example:** If a set A = {1, 2}. Determine all relations from A to A.

**Solution:** There are $2^2$= 4 elements i.e., {(1, 2), (2, 1), (1, 1), (2, 2)} in A x A. So, there are $2^4$= 16 relations from A to A. i.e.

1. {(1, 2), (2, 1), (1, 1), (2, 2)}, {(1, 2), (2, 1)}, {(1, 2), (1, 1)}, {(1, 2), (2, 2)},
2. {(2, 1), (1, 1)},{(2,1), (2, 2)}, {(1, 1),(2, 2)},{(1, 2), (2, 1), (1 , 1)}, {(1, 2), (1, 1),
3. (2, 2)}, {(2,1), (1, 1), (2, 2)}, {(1, 2), (2, 1), (2, 2)}, {(1, 2), ( 2, 1), (1, 1), (2, 2)} and ∅.

**Domain and Range of Relation**

**Domain of Relation:** The Domain of relation R is the set of elements in P which are related to some elements in Q, or it is the set of all first entries of the ordered pairs in R. It is denoted by DOM (R).

**Range of Relation:** The range of relation R is the set of elements in Q which are related to some element in P, or it is the set of all second entries of the ordered pairs in R. It is denoted by RAN (R).

Space for learners:

**Example:**

1. Let A = {1, 2, 3, 4}
2. B = {a, b, c, d}
3. R = {(1, a), (1, b), (1, c), (2, b), (2, c), (2, d)}.

**Solution:**

DOM (R) = {1, 2}

RAN (R) = {a, b, c, d}

**Complement of a Relation**

Consider a relation R from a set A to set B. The complement of relation R denoted by R is a relation from A to B such that

R = {(a, b): {a, b) ∉ R}.

**Example:**

1. Consider the relation R from X to Y
2. X = {1, 2, 3}
3. Y = {8, 9}
4. R = {(1, 8) (2, 8) (1, 9) (3, 9)}

**Q. Find the complement relation of R.**

**Solution:**

X x Y = {(1, 8), (2, 8), (3, 8), (1, 9), (2, 9), (3, 9)}

Now we find the complement relation R from X x Y

R = {(3, 8), (2, 9)}

**Representation of Relations**

Relations can be represented in many ways. Some of which are as follows:

**1. Relation as a Matrix:** Let P = [$a_1$,$a_2$,$a_3$,.......$a_m$] and Q = [$b_1$,$b_2$,$b_3$......$b_n$] are finite sets, containing m and n number of elements respectively. R is a relation from P to Q. The relation R can be represented by m x n matrix M = [$M_{ij}$], defined as

$$M_{ij} = \begin{cases} 0, \text{ if } (a_i, b_i) \notin R \\ 1, \text{ if } (a_i, b_i) \in R \end{cases}$$

**Example**

1. Let P = {1, 2, 3, 4}, Q = {a, b, c, d} and R = {(1, a), (1, b), (1, c), (2, b), (2, c), (2, d)}.

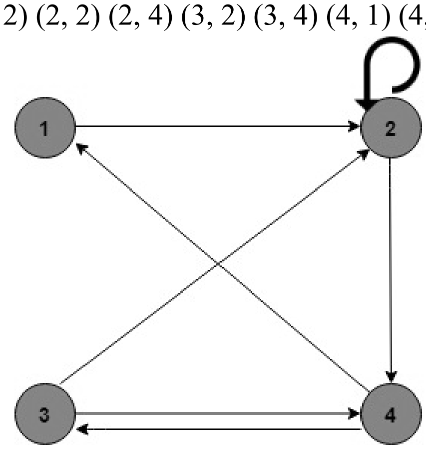
The matrix of relation R is shown as fig:

$$M_R = \begin{array}{c c} & \begin{array}{cccc} a & b & c & d \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \left\{ \begin{array}{cccc} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array} \right\} \end{array}$$

**2. Relation as a Directed Graph:** There is another way of picturing a relation R when R is a relation from a finite set to itself.

**Example**

1. A = {1, 2, 3, 4}
2. R = {(1, 2) (2, 2) (2, 4) (3, 2) (3, 4) (4, 1) (4, 3)}




**3. Relation as an Arrow Diagram:** If P and Q are finite sets and R is a relation from P to Q. Relation R can be represented as an arrow diagram as follows.

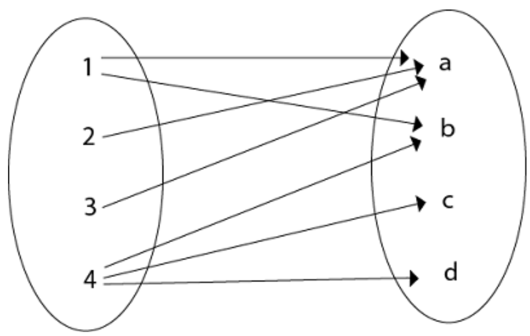
Draw two ellipses for the sets P and Q. Write down the elements of P and elements of Q column-wise in three ellipses. Then draw an arrow from the first ellipse to the second ellipse if a is related to b and a ∈ P and b ∈ Q.

***Space for learners:***

**Example**

1. Let P = {1, 2, 3, 4}
2. Q = {a, b, c, d}
3. R = {(1, a), (2, a), (3, a), (1, b), (4, b), (4, c), (4, d)

The arrow diagram of relation R is shown in fig:




**4. Relation as a Table:** If P and Q are finite sets and R is a relation from P to Q. Relation R can be represented in tabular form.

Make the table which contains rows equivalent to an element of P and columns equivalent to the element of Q. Then place a cross (X) in the boxes which represent relations of elements on set P to set Q.

**Example**

1. Let P = {1, 2, 3, 4}
2. Q = {x, y, z, k}
3. R = {(1, x), (1, y), (2, z), (3, z), (4, k)}.

The tabular form of relation as shown in fig:

| | x | y | z | k |
|---|---|---|---|---|
| 1 | x | x | | |
| 2 | | | x | |
| 3 | | | x | |
| 4 | | | | x |

***Space for learners:***

**Composition of Relations**

*Space for learners:*

Let A, B, and C be sets, and let R be a relation from A to B and let S be a relation from B to C. That is, R is a subset of A × B and S is a subset of B × C. Then R and S give rise to a relation from A to C indicated by R◦S and defined by:

1. a (R◦S)c **if for** some $b \in B$ we have aRb and bSc.   is,
2. $R \circ S = \{(a, c)|$ there exists $b \in B$ **for** which $(a, b) \in R$ and $(b, c) \in S\}$

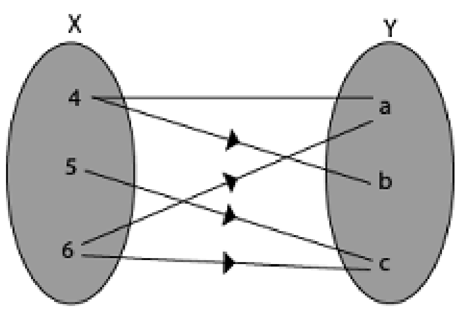
The relation R◦S is known the composition of R and S; it is sometimes denoted simply by RS.

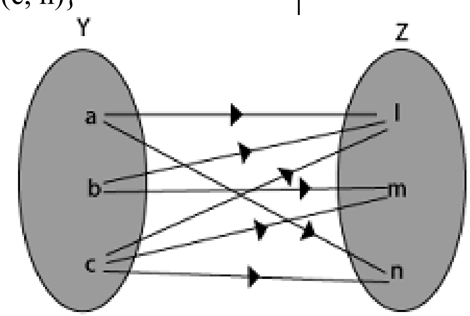
Let R is a relation on a set A, that is, R is a relation from a set A to itself. Then R◦R, the composition of R with itself, is always represented. Also, R◦R is sometimes denoted by $R^2$. Similarly, $R^3 = R^2 \circ R = R \circ R \circ R$, and so on. Thus $R^n$ is defined for all positive n.

**Example:** Let X = {4, 5, 6}, Y = {a, b, c} and Z = {l, m, n}. Consider the relation $R_1$ from X to Y and $R_2$ from Y to Z.

$R_1$ = {(4, a), (4, b), (5, c), (6, a), (6, c)}

$R_2$ = {(a, l), (a, n), (b, l), (b, m), (c, l), (c, m), (c, n)}




Find the composition of relation **(i)** $R_1$ o $R_2$ **(ii)** $R_1$o $R_1^{-1}$

**Solution:**

(i) The composition relation $R_1$ o $R_2$ as shown in fig:

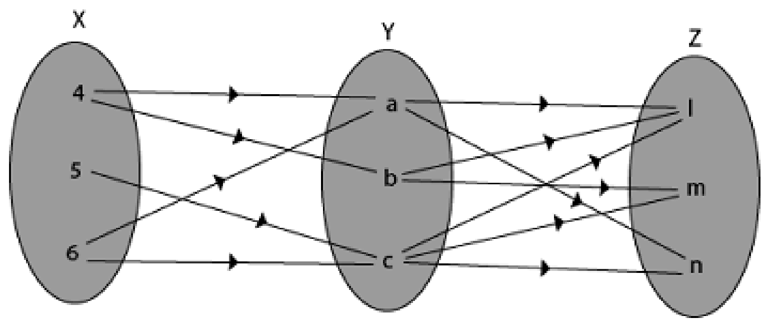



Fig : $R_1$ o $R_2$

**$R_1$ o $R_2$** = {(4, l), (4, n), (4, m), (5, l), (5, m), (5, n), (6, l), (6, m), (6, n)}

(ii) The composition relation $R_1$o $R_1^{-1}$ as shown in fig:

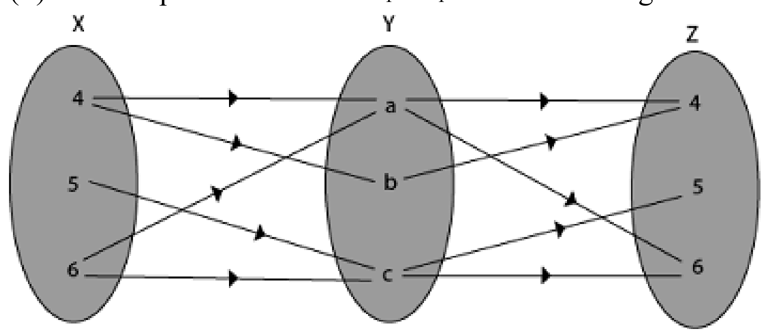



Fig : $R_1$ o $R_1^{-1}$

**Composition of Relations and Matrices**

There is another way of finding R∘S. Let $M_R$ and $M_S$ denote respectively the matrix representations of the relations R and S. Then

**Example**

Let P = {2, 3, 4, 5}. Consider the relation R and S on P defined by R = {(2, 2), (2, 3), (2, 4), (2, 5), (3, 4), (3, 5), (4, 5), (5, 3)}

S = {(2, 3), (2, 5), (3, 4), (3, 5), (4, 2), (4, 3), (4, 5), (5, 2), (5, 5)}.

1. Find the matrices of the above relations.
2. Use matrices to find the following composition of the relatio n R and S.
3. (i)RoS (ii)RoR (iii)SoR

**Solution:** The matrices of the relation R and S are a shown in fig:

Space for learners:

***Space for learners:***

$$M_R = \begin{array}{cc} & \begin{array}{cccc} 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{Bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{Bmatrix} \end{array} \quad \text{and} \quad M_S = \begin{array}{cc} & \begin{array}{cccc} 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{Bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{Bmatrix} \end{array}$$

(i) To obtain the composition of relation R and S. First multiply $M_R$ with $M_S$ to obtain the matrix $M_R$ x $M_S$ as shown in fig:

The non zero entries in the matrix $M_R$ x $M_S$ tells the elements related in RoS. So,

$$M_R \times M_S = \begin{array}{cc} & \begin{array}{cccc} 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{Bmatrix} 2 & 2 & 1 & 4 \\ 2 & 1 & 0 & 2 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{Bmatrix} \end{array}$$

Hence the composition R o S of the relation R and S is

R o S = {(2, 2), (2, 3), (2, 4), (3, 2), (3, 3), (4, 2), (4, 5), (5, 2), (5, 3), (5, 4), (5, 5)}.

(ii) First, multiply the matrix $M_R$ by itself, as shown in fig

$$M_R \times M_R = \begin{array}{cc} & \begin{array}{cccc} 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{Bmatrix} 1 & 2 & 2 & 3 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{Bmatrix} \end{array}$$

Hence the composition R o R of the relation R and S is

R o R = {(2, 2), (3, 2), (3, 3), (3, 4), (4, 2), (4, 5), (5, 2), (5, 3), (5, 5) }

(iii) Multiply the matrix $M_S$ with $M_R$ to obtain the matrix $M_S$ x $M_R$ as shown in fig:

$$M_S \times M_R = \begin{array}{c} \\ 2 \\ 3 \\ 4 \\ 5 \end{array}\begin{array}{c} \begin{array}{cccc} 2 & 3 & 4 & 5 \end{array} \\ \left\{\begin{array}{cccc} 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 2 & 2 & 2 \\ 1 & 1 & 1 & 1 \end{array}\right\} \end{array}$$

The non-zero entries in matrix $M_S$ x $M_R$ tells the elements related in S o R.

Hence the composition S o R of the relation S and R is

S o R = {(2, 4) , (2, 5), (3, 3), (3, 4), (3, 5), (4, 2), (4, 4), (4, 5), (5, 2) , (5, 3), (5, 4), (5, 5)}.

**More on Types of Relations**

**1. Reflexive Relation:** A relation R on set A is said to be a reflexive if **(a, a) ∈ R** for every **a ∈** A.

**Example:** If A = {1, 2, 3, 4} then R = {(1, 1) (2, 2), (1, 3), (2, 4), (3, 3), (3, 4), (4, 4)}. Is a relation reflexive?

**Solution:** The relation is reflexive as for every a ∈ A. (a, a) ∈ R, i.e. (1, 1), (2, 2), (3, 3), (4, 4) ∈ R.

**2. Irreflexive Relation:** A relation R on set A is said to be **irreflexive if (a, a) ∉ R** for every **a ∈ A**.

**Example:** Let A = {1, 2, 3} and R = {(1, 2), (2, 2), (3, 1), (1, 3)}. Is the relation R reflexive or irreflexive?

**Solution:** The relation R is not reflexive as for every a ∈ A, (a, a) ∉ R, i.e., (1, 1) and (3, 3) ∉ R. The relation R is not irreflexive as (a, a) ∉ R, for some a ∈ A, i.e., (2, 2) ∈ R.

**3. Symmetric Relation:** A relation R on set A is said to be symmetric iff (a, b) ∈ R ⟺ (b, a) ∈ R.

**Example:** Let A = {1, 2, 3} and R = {(1, 1), (2, 2), (1, 2), (2, 1), (2, 3), (3, 2)}. Is a relation R symmetric or not?

**Solution:** The relation is symmetric as for every (a, b) ∈ R, we have (b, a) ∈ R, i.e., (1, 2), (2, 1), (2, 3), (3, 2) ∈ R but not reflexive because (3, 3) ∉ R.

***Space for learners:***

**Example of Symmetric Relation:**

1. Relation ⊥r is symmetric since a line a is ⊥r to b, then b is ⊥r to a.
2. Also, Parallel is symmetric, since if a line a is ∥ to b then b is also ∥ to a.

**Antisymmetric Relation:** A relation R on a set A is antisymmetric iff (a, b) ∈ R and (b, a) ∈ R then a = b.

**Example:** Let A = {1, 2, 3} and R = {(1, 1), (2, 2)}. Is the relation R antisymmetric?

**Solution:** The relation R is antisymmetric as a = b when (a, b) and (b, a) both belong to R.

**Example:** Let A = {4, 5, 6} and R = {(4, 4), (4, 5), (5, 4), (5, 6), (4, 6)}. Is the relation R antisymmetric?

**Solution:** The relation R is not antisymmetric as 4 ≠ 5 but (4, 5) and (5, 4) both belong to R.

**5. Asymmetric Relation:** A relation R on a set A is called an Asymmetric Relation if for every (a, b) ∈ R implies that (b, a) does not belong to R.

**6. Transitive Relations:** A Relation R on set A is said to be transitive iff (a, b) ∈ R and (b, c) ∈ R ⟺ (a, c) ∈ R.

**Example:** Let A = {1, 2, 3} and R = {(1, 2), (2, 1), (1, 1), (2, 2)}. Is the relation transitive?

**Solution:** The relation R is transitive as for every (a, b) (b, c) belong to R, we have (a, c) ∈ R i.e, (1, 2) (2, 1) ∈ R ⇒ (1, 1) ∈ R.

**Note 1: The Relation ≤, ⊆ and / are transitive, i.e., a ≤ b, b ≤ c then a ≤ c**

**(ii) Let a ⊆ b, b ⊆ c then a ⊆ c**

**(iii) Let a/b, b/c then a/c.**

**Note 2: ⊥r is not transitive since a ⊥r b, b ⊥r c then it is not true that a ⊥r c. Since no line is ∥ to itself, we can have a ∥ b, b ∥ a but a ∦ a.**

**Thus ∥ is not transitive, but it will be transitive in the plane.**

***Space for learners:***

**7. Identity Relation:** Identity relation I on set A is reflexive, transitive and symmetric. So identity relation I is an Equivalence Relation.

**Example:** A= {1, 2, 3} = {(1, 1), (2, 2), (3, 3)}

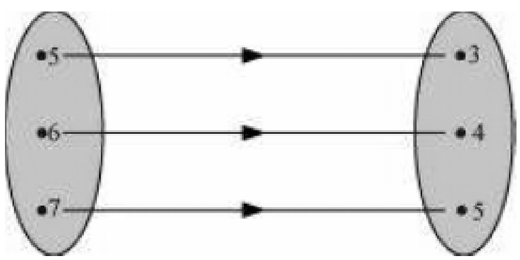
**8. Void Relation:** It is given by R: A →B such that R = ∅ (⊆ A x B) is a null relation. Void Relation R = ∅ is symmetric and transitive but not reflexive.

**9. Universal Relation:** A relation R: A →B such that R = A x B (⊆ A x B) is a universal relation. Universal Relation from A →B is reflexive, symmetric and transitive. So this is an equivalence relation.

Space for learners:

**CHECK YOUR PROGRESS**

1. The given figure shows a relationship between the sets P and Q. write this relation




(i) in set-builder form (ii) in roster form. What is its domain and range?

2. Let A = {1, 2, 3, 4, 6}. Let R be the relation on A defined by

{(a, b): a, b ∈A, bis exactly divisible by a}.

(i) Write R in roster form

(ii) (ii) Find the domain of R

(iii) (iii) Find the range of R.

3. Determine the domain and range of the relation R defined by R = {(x, x+ 5): x ∈{0, 1, 2, 3, 4, 5}}.

4. Which of the following relations are functions? Give reasons. If it is a function, determine its domain and range.

(i) {(2, 1), (5, 1), (8, 1), (11, 1), (14, 1), (17, 1)}

(ii) {(2, 1), (4, 2), (6, 3), (8, 4), (10, 5), (12, 6), (14, 7)}

(iii) {(1, 3), (1, 5), (2, 5)}

Space for learners:

## 3.4 CLOSURE PROPERTIES OF RELATIONS

Consider a given set A, and the collection of all relations on A. Let P be a property of such relations, such as being symmetric or being transitive. A relation with property P will be called a P-relation. The P-closure of an arbitrary relation R on A, indicated P (R), is a P-relation such that

$R \subseteq P(R) \subseteq S$

**(1) Reflexive and Symmetric Closures:** The next theorem tells us how to obtain the reflexive and symmetric closures of a relation easily.

**Theorem:** Let R be a relation on a set A. Then:

- $R \cup \Delta_A$ is the reflexive closure of R
- $R \cup R^{-1}$ is the symmetric closure of R.

**Example:**

Let A = {k, l, m}. Let R is a relation on A defined by R = {(k, k), (k , l), (l, m), (m, k)}.

Find the reflexive closure of R.

**Solution:** $R \cup \Delta$ is the smallest relation having reflexive property, Hence,

**RF = R ∪ Δ = {(k, k), (k, l), (l, l), (l, m), (m, m), (m, k)}.**

**Example:** Consider the relation R on A = {4, 5, 6, 7} defined by

R = {(4, 5), (5, 5), (5, 6), (6, 7), (7, 4), (7, 7)}

Find the symmetric closure of R.

**Solution:** The smallest relation containing R having the symmetric property is $R \cup R^{-1}$,i.e.

RS = R ∪ R-1 = {(4, 5), (5, 4), (5, 5), (5, 6), (6, 5), (6, 7), (7, 6), (7, 4), (4, 7), (7, 7)}.

**(2) Transitive Closures:** Consider a relation R on a set A. The transitive closure R of a relation R of a relation R is the smallest transitive relation containing R.

Recall that $R^2 = R \circ R$ and $R^n = R^{n-1} \circ R$. We define

$$R^* = \bigcup_{i=1}^{\infty} R^i$$

The following Theorem applies:

**Theorem1:** $R^*$ is the transitive closure of R

Suppose A is a finite set with n elements.

$R^* = R \cup R^2 \cup \ldots\ldots \cup R^n$

**Theorem 2:** Let R be a relation on a set A with n elements. Then

Transitive (R) = $R \cup R^2 \cup \ldots\ldots \cup R^n$

**Example:** Consider the relation R = {(1, 2), (2, 3), (3, 3)} on A = {1, 2, 3}. Then $R^2 = R \circ R$ = {(1, 3), (2, 3), (3, 3)} and $R^3 = R^2 \circ R$ = {(1, 3), (2, 3), (3, 3)} Accordingly,
Transitive (R) = {(1, 2), (2, 3), (3, 3), (1, 3)}

**Example:** Let A = {4, 6, 8, 10} and R = {(4, 4), (4, 10), (6, 6), (6, 8), (8, 10)} is a relation on set A. Determine transitive closure of R.

**Solution:** The matrix of relation R is shown in fig:

$$M_R = \begin{array}{c} \\ 4 \\ 6 \\ 8 \\ 10 \end{array} \begin{array}{c} \begin{array}{cccc} 4 & 6 & 8 & 10 \end{array} \\ \begin{Bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{Bmatrix} \end{array}$$

Now, find the powers of $M_R$ as in fig:

Space for learners:

$$M_{R^2} = \begin{array}{cc} & \begin{array}{cccc} 4 & 6 & 8 & 10 \end{array} \\ \begin{array}{c} 4 \\ 6 \\ 8 \\ 10 \end{array} & \left\{ \begin{array}{cccc} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array} \right\} \end{array} \qquad M_{R^3} = \begin{array}{cc} & \begin{array}{cccc} 4 & 6 & 8 & 10 \end{array} \\ \begin{array}{c} 4 \\ 6 \\ 8 \\ 10 \end{array} & \left\{ \begin{array}{cccc} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array} \right\} \end{array}$$

$$M_{R^4} = \begin{array}{cc} & \begin{array}{cccc} 4 & 6 & 8 & 10 \end{array} \\ \begin{array}{c} 4 \\ 6 \\ 8 \\ 10 \end{array} & \left\{ \begin{array}{cccc} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array} \right\} \end{array}$$

Hence, the transitive closure of $M_R$ is $M_R^*$ as shown in Fig (where $M_R^*$ is the ORing of a power of $M_R$).

$$M_{R^*} = M_R \vee M_{R^2} \vee M_{R^3} \vee M_{R^4}; \quad M_{R^*} = \begin{array}{cc} & \begin{array}{cccc} 4 & 6 & 8 & 10 \end{array} \\ \begin{array}{c} 4 \\ 6 \\ 8 \\ 10 \end{array} & \left\{ \begin{array}{cccc} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array} \right\} \end{array}$$

**Thus**, $R^* = \{(4, 4), (4, 10), (6, 8), (6, 6), (6, 10), (8, 10)\}$.

Note: While ORing the power of the matrix R, we can eliminate MRn because it is equal to MR* if n is even and is equal to MR3 if n is odd.

## 3.5 EQUIVALENCE OF RELATIONS

**Equivalence Relations**

A relation R on a set A is called an equivalence relation if it satisfies following three properties:

1. Relation R is Reflexive, i.e. aRa ∀ a∈A.
2. Relation R is Symmetric, i.e., aRb ⟹ bRa

***Space for learners:***

3. Relation R is transitive, i.e., aRb and bRc ⟹ aRc.

**Example:** Let A = {1, 2, 3, 4} and R = {(1, 1), (1, 3), (2, 2), (2, 4), (3, 1), (3, 3), (4, 2), (4, 4)}.

Show that R is an Equivalence Relation.

Solution:

Reflexive: Relation R is reflexive as (1, 1), (2, 2), (3, 3) and (4, 4) ∈ R.

Symmetric: Relation R is symmetric because whenever (a, b) ∈ R, (b, a) also belongs to R.

**Example:** (2, 4) ∈ R ⟹ (4, 2) ∈ R.

Transitive: Relation R is transitive because whenever (a, b) and (b, c) belongs to R, (a, c) also belongs to R.

**Example:** (3, 1) ∈ R and (1, 3) ∈ R ⟹ (3, 3) ∈ R.

So, as R is reflexive, symmetric and transitive, hence, R is an Equivalence Relation.

Note1: If $R_1$ and $R_2$ are equivalence relation then $R_1 \cap R_2$ is also an equivalence relation.

**Example:** A = {1, 2, 3}

$$R_1 = \{(1, 1), (2, 2), (3, 3), (1, 2), (2, 1)\}$$

$$R_2 = \{(1, 1), (2, 2), (3, 3), (2, 3), (3, 2)\}$$

$$R_1 \cap R_2 = \{(1, 1), (2, 2), (3, 3)\}$$

Note2: If $R_1$ and $R_2$ are equivalence relation then $R_1 \cup R_2$ may or may not be an equivalence relation.

**Example:** A = {1, 2, 3}

$$R_1 = \{(1, 1), (2, 2), (3, 3), (1, 2), (2, 1)\}$$

$$R_2 = \{(1, 1), (2, 2), (3, 3), (2, 3), (3, 2)\}$$

$$R_1 \cup R_2 = \{(1, 1), (2, 2), (3, 3), (1, 2), (2, 1), (2, 3), (3, 2)\}$$

Hence, Reflexive or Symmetric are Equivalence Relation but transitive may or may not be an equivalence relation.

Inverse Relation

***Space for learners:***

Let R be any relation from set A to set B. The inverse of R denoted by $R^{-1}$ is the relations from B to A which consist of those ordered pairs which when reversed belong to R that is:

$R^{-1}$ = {(b, a): (a, b) ∈ R}

**Example**: A = {1, 2, 3}

B = {x, y, z}

Solution: R = {(1, y), (1, z), (3, y)

$R^{-1}$ = {(y, 1), (z, 1), (y, 3)}

Clearly $(R^{-1})^{-1}$ = R

Note1: Domain and Range of $R^{-1}$ is equal to range and domain of R.

**Example:** R = {(1, 1), (2, 2), (3, 3), (1, 2), (2, 3), (3, 2)}

$R^{-1}$ = {(1, 1), (2, 2), (3, 3), (2, 1), (3, 2), (2, 3)}

Note2: If R is an Equivalence relation then $R^{-1}$ is always an Equivalence relation.

**Example:** Let A = {1, 2, 3}

R = {(1, 1), (2, 2), (3, 3), (1, 2), (2, 1)}

$R^{-1}$ = {(1, 1), (2, 2), (3, 3), (2, 1), (1, 2)}

$R^{-1}$ is a Equivalence Relation.

Note3: If R is a Symmetric Relation then $R^{-1}$=R and vice-versa.

**Example:** Let A = {1, 2, 3}

R = {(1, 1), (2, 2), (1, 2), (2, 1), (2, 3), (3, 2)}

$R^{-1}$ = {(1, 1), (2, 2), (2, 1), (1, 2), (3, 2), (2, 3)}

Note 4: Reverse Order of Law $(S \circ T)^{-1} = T^{-1}S^{-1}$ $(R \circ S \circ T)^{-1} = T^{-1} \circ S^{-1} \circ R^{-1}$.

Space for learners:

## 3.5 PARTIAL ORDER OF RELATIONS

Space for learners:

A relation R on a set A is called a partial order relation if it satisfies the following three properties:

1. Relation R is Reflexive, i.e. aRa ∀ a∈A.
2. Relation R is Antisymmetric, i.e., aRb and bRa ⟹ a = b.
3. Relation R is transitive, i.e., aRb and bRc ⟹ aRc.

**Example:** Show whether the relation (x, y) ∈ R, if, x ≥ y defined on the set of +ve integers is a partial order relation.

Solution: Consider the set A = {1, 2, 3, 4} containing four +ve integers. Find the relation for this set such as R = {(2, 1), (3, 1), (3, 2), (4, 1), (4, 2), (4, 3), (1, 1), (2, 2), (3, 3), (4, 4)}.

Reflexive: The relation is reflexive as for every a ∈ A. (a, a) ∈ R, i.e. (1, 1), (2, 2), (3, 3), (4, 4) ∈ R.

Antisymmetric: The relation is antisymmetric as whenever (a, b) and (b, a) ∈ R, we have a = b.

Transitive: The relation is transitive as whenever (a, b) and (b, c) ∈ R, we have (a, c) ∈ R.

**Example:** (4, 2) ∈ R and (2, 1) ∈ R, implies (4, 1) ∈ R.

As the relation is reflexive, antisymmetric and transitive. Hence, it is a partial order relation.

**Example:** Show that the relation 'Divides' defined on N is a partial order relation.

**Solution:**

Reflexive: We have a divides a, ∀ a∈N. Therefore, relation 'Divides' is reflexive.

Antisymmetric: Let a, b, c ∈N, such that a divides b. It implies b divides a iff a = b. So, the relation is antisymmetric.

Transitive: Let a, b, c ∈N, such that a divides b and b divides c.

Then a divides c. Hence the relation is transitive. Thus, the relation being reflexive, antisymmetric and transitive, the relation 'divides' is a partial order relation.

**Example3:** (a) The relation ⊆ of a set of inclusion is a partial ordering or any collection of sets since set inclusion has three desired properties:

Space for learners:

1. A ⊆ A for any set A.
2. If A ⊆ B and B ⊆ A then B = A.
3. If A ⊆ B and B ⊆ C then A ⊆ C

(b) The relation ≤ on the set R of real no that is Reflexive, Antisymmetric and transitive.

(c) Relation ≤ is a Partial Order Relation.

n-Ary Relations

By an n-ary relation, we mean a set of ordered n-tuples. For any set S, a subset of the product set Sn is called an n-ary relation on S. In particular, a subset of S3 is called a ternary relation on S.

**Partial Order Set (POSET):**

The set A together with a partial order relation R on the set A and is denoted by (A, R) is called a partial orders set or POSET.

**Total Order Relation**

Consider the relation R on the set A. If it is also called the case that for all, a, b ∈ A, we have either (a, b) ∈ R or (b, a) ∈ R or a = b, then the relation R is known total order relation on set A.

**Example:** Show that the relation '<' (less than) defined on N, the set of +ve integers is neither an equivalence relation nor partially ordered relation but is a total order relation.

Solution:

Reflexive: Let a ∈ N, then a < a

⟹ '<' is not reflexive.

As, the relation '<' (less than) is not reflexive, it is neither an equivalence relation nor the partial order relation.

But, as ∀ a, b ∈ N, we have either a < b or b < a or a = b. So, the relation is a total order relation.

**Equivalence Class**

Consider, an equivalence relation R on a set A. The equivalence class of an element a ∈ A, is the set of elements of A to which element a is related. It is denoted by [a].

**Example:** Let R be an equivalence relations on the set A = {4, 5, 6, 7} defined by

$$R = \{(4, 4), (5, 5), (6, 6), (7, 7), (4, 6), (6, 4)\}.$$

Determine its equivalence classes.

Solution: The equivalence classes are as follows:

$$\{4\} = \{6\} = \{4, 6\}$$

$$\{5\} = \{5\}$$

$$\{7\} = \{7\}.$$

**Circular Relation**

Consider a binary relation R on a set A. Relation R is called circular if $(a, b) \in R$ and $(b, c) \in R$ implies $(c, a) \in R$.

**Example:** Consider R is an equivalence relation. Show that R is reflexive and circular.

Solution: Reflexive: As, the relation, R is an equivalence relation. So, reflexivity is the property of an equivalence relation. Hence, R is reflexive.

Circular: Let $(a, b) \in R$ and $(b, c) \in R$

$$\Rightarrow (a, c) \in R \quad (\because R \text{ is transitive})$$

$$\Rightarrow (c, a) \in R \quad (\because R \text{ is symmetric})$$

Thus, R is Circular.

Compatible Relation

A binary relation R on a set A that is Reflexive and symmetric is called Compatible Relation.

Every Equivalence Relation is compatible, but every compatible relation need not be an equivalence.

**Example:** Set of a friend is compatible but may not be an equivalence relation.

Friend Friend

$a \rightarrow b$, $b \rightarrow c$ but possible that a and c are not friends.

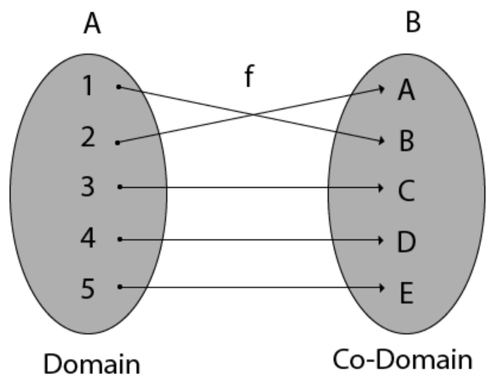
Space for learners:

## 3.6 INTRODUCTION TO FUNCTION

Space for learners:

**Functions**

It is a mapping in which every element of set A is uniquely associated at the element with set B. The set of A is called Domain of a function and set of B is called Co domain.




**Domain, Co-Domain, and Range of a Function:**

Domain of a Function: Let f be a function from P to Q. The set P is called the domain of the function f.

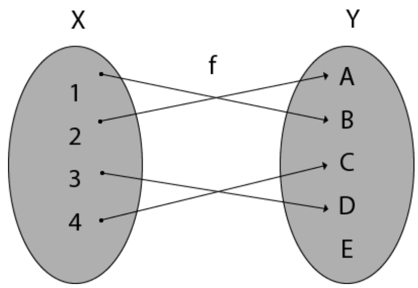
Co-Domain of a Function: Let f be a function from P to Q. The set Q is called Co-domain of the function f.

Range of a Function: The range of a function is the set of picture of its domain. In other words, we can say it is a subset of its co-domain. It is denoted as f (domain).

1. If f: $P \rightarrow Q$, then $f(P) = \{f(x): x \in P\} = \{y: y \in Q \mid \exists\, x \in P$, such that $f(x) = y\}$.

**Example:** Find the Domain, Co-Domain, and Range of function.

1. Let x = {1, 2, 3, 4}
2. y = {a, b, c, d, e}
3. f = {(1, b), (2, a), (3, d), (4, c)}




Space for learners:

**Solution:**

Domain of function: {1, 2, 3, 4}

Range of function: {a, b, c, d}

Co-Domain of function: {a, b, c, d, e}

Functions as a Set

If P and Q are two non-empty sets, then a function f from P to Q is a subset of P x Q, with two important restrictions

1. $\forall$ a $\in$ P, (a, b) $\in$ f for some b $\in$ Q
2. If (a, b) $\in$ f and (a, c) $\in$ f then b = c.

Note1: There may be some elements of the Q which are not related to any element of set P.

2. Every element of P must be related with at least one element of Q.

**Example:** If a set A has n elements, how many functions are there from A to A?

Solution: If a set A has n elements, then there are nn functions from A to A.

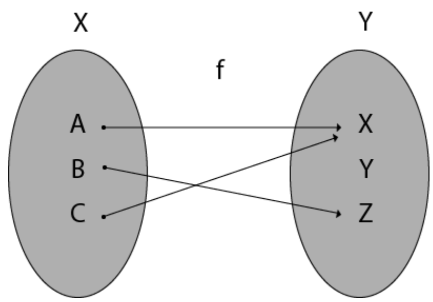
**Representation of a Function**

The two sets P and Q are represented by two circles. The function f: P $\rightarrow$ Q is represented by a collection of arrows joining the points which represent the elements of P and corresponds elements of Q

**Example:** Let X = {a, b, c} and Y = {x, y, z} and f: X $\rightarrow$ Y such that

f= {(a, x), (b, z), (c, x)}

Then f can be represented diagrammatically as follows




**Example:** Let X = {x, y, z, k} and Y = {1, 2, 3, 4}. Determine which of the following functions. Give reasons if it is not. Find range if it is a function.

a. f = {(x, 1), (y, 2), (z, 3), (k, 4)

b. g = {(x, 1), (y, 1), (k, 4)

c. h = {(x, 1), (x, 2), (x, 3), (x, 4)

d. l = {(x, 1), (y, 1), (z, 1), (k, 1)}

e. d = {(x, 1), (y, 2), (y, 3), (z, 4), (z, 4)}.

**Solution:**

1. It is a function. Range (f) = {1, 2, 3, 4}

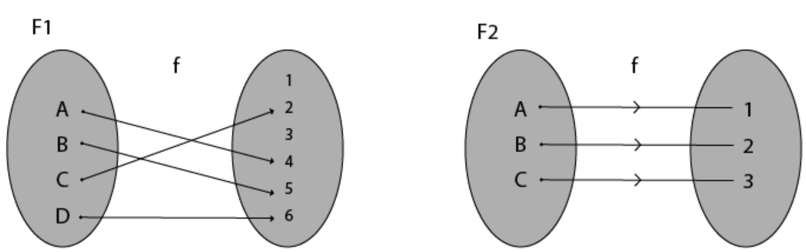
2. It is not a function because every element of X does not relate with some element of Y i.e., Z is not related with any element of Y.

3. h is not a function because h (x) = {1, 2, 3, 4} i.e., element x has more than one image in set Y.

4. d is not a function because d (y) = {2, 3} i.e., element y has more than image in set Y.

**Types of Functions**

**Injective (One-to-One) Functions:** A function in which one element of Domain Set is connected to one element of Co-Domain Set.



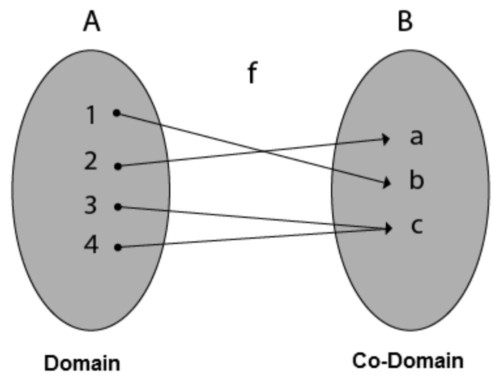

**F1 and F2 show one to one Function**

Space for learners:

**Surjective (Onto) Functions:** A function in which every element of Co-Domain Set has one pre-image.

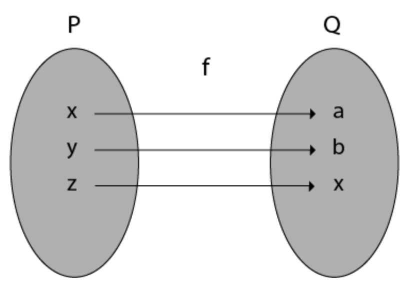
**Example:** Consider, A = {1, 2, 3, 4}, B = {a, b, c} and f = {(1, b), (2, a), (3, c), (4, c)}.

It is a Surjective Function, as every element of B is the image of some A




Note: In an Onto Function, Range is equal to Co-Domain.

**Bijective (One-to-One Onto) Functions:** A function which is both injective (one to - one) and surjective (onto) is called bijective (One-to-One Onto) Function.




**Example:** Consider P = {x, y, z} Q = {a, b, c} and f: P → Q such that

f = {(x, a), (y, b), (z, c)}

The f is a one-to-one function and also it is onto. So it is a bijective function.

**Into Functions:** A function in which there must be an element of co-domain Y does not have a pre-image in domain X.

**Example:** Consider, A = {a, b, c}

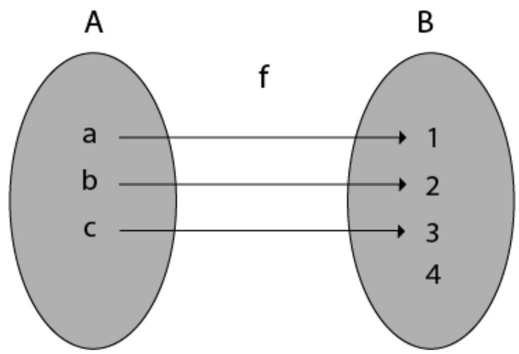
B = {1, 2, 3, 4} and f: A → B such that

Space for learners:

$f = \{(a, 1), (b, 2), (c, 3)\}$

In the function f, the range i.e., $\{1, 2, 3\} \neq$ co-domain of Y i.e., $\{1, 2, 3, 4\}$

Therefore, it is an into function

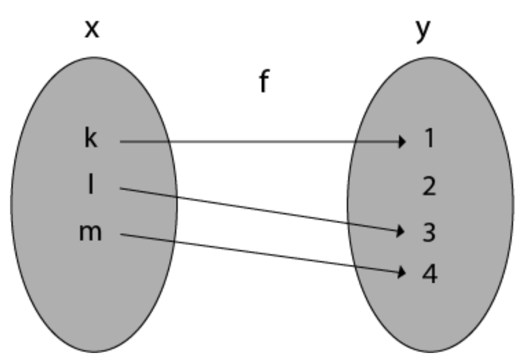



**One-One Into Functions:** Let $f: X \to Y$. The function f is called one-one into function if different elements of X have different unique images of Y.

**Example:** Consider, $X = \{k, l, m\}$ $Y = \{1, 2, 3, 4\}$ and $f: X \to Y$ such that

$f = \{(k, 1), (l, 3), (m, 4)\}$

The function f is a one-one into function



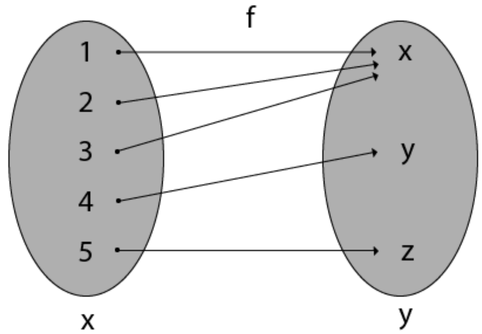

**Many-One Functions:** Let $f: X \to Y$. The function f is said to be many-one functions if there exist two or more than two different elements in X having the same image in Y.

**Example:** Consider $X = \{1, 2, 3, 4, 5\}$ $Y = \{x, y, z\}$ and $f: X \to Y$ such that

$f = \{(1, x), (2, x), (3, x), (4, y), (5, z)\}$

Space for learners:

The function f is a many-one function

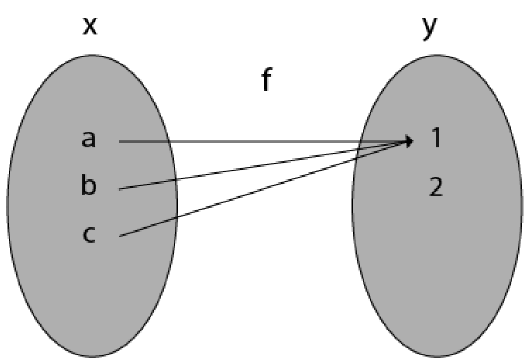



**Many-One Into Functions:** Let f: X → Y. The function f is called the many-one function if and only if is both many one and into function.

**Example:** Consider X = {a, b, c}   Y = {1, 2} and f: X → Y such that

f = {(a, 1), (b, 1), (c, 1)}

As the function f is a many-one and into, so it is a many-one into function.



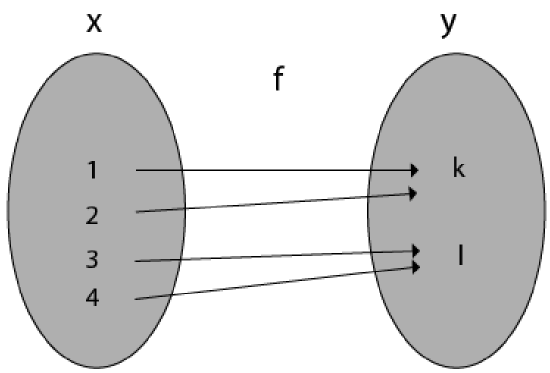

**Many-One Onto Functions:** Let f: X → Y. The function f is called many-one onto function if and only if is both many one and onto.

**Example:** Consider X = {1, 2, 3, 4}  Y = {k, l} and f: X → Y such that

f = {(1, k), (2, k), (3, l), (4, l)}

The function f is a many-one (as the two elements have the same image in Y) and it is onto (as every element of Y is the image of some element X). So, it is many-one onto function

Space for learners:




**Identity Functions**

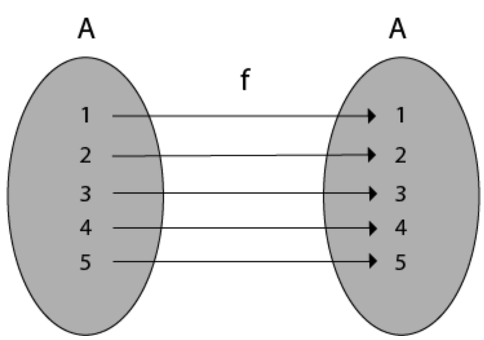
The function f is called the identity function if each element of set A has an image on itself i.e. f (a) = a ∀ a ∈ A.

It is denoted by I.

**Example:** Consider, A = {1, 2, 3, 4, 5} and f: A → A such that

. f = {(1, 1), (2, 2), (3, 3), (4, 4), (5, 5)}.

The function f is an identity function as each element of A is mapped onto itself. The function f is a one-one and onto




**Invertible (Inverse) Functions**

A function f: X → Y is invertible if and only if it is a bijective function.

Consider the bijective (one to one onto) function f: X → Y. As f is a one to one, therefore, each element of X corresponds to a distinct element of Y. As f is onto, there is no element of Y which is not the image of any element of X, i.e., range = co-domain Y.

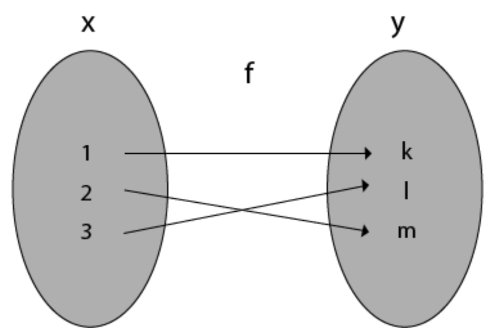
The inverse function for f exists if f-1 is a function from Y to X.

**Example:** Consider, X = {1, 2, 3}

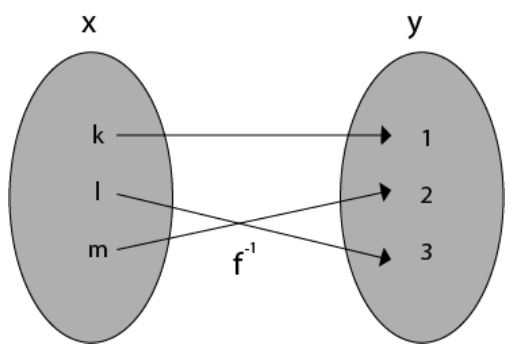
Y = {k, l, m} and f: X→Y such that

Space for learners:

$f = \{(1, k), (2, m), (3, l)\}$




The inverse function of f is shown in fig:




**Compositions of Functions**

Consider functions, $f: A \rightarrow B$ and $g: B \rightarrow C$. The composition of f with g is a function from A into C defined by (gof) (x) = g [f(x)] and is defined by gof.

To find the composition of f and g, first find the image of x under f and then find the image of f (x) under g.

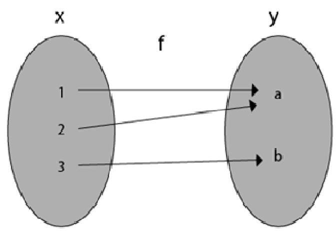
**Example:** Let X = {1, 2, 3}

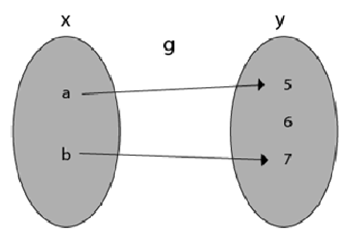
. Y = {a, b}

. Z = {5, 6, 7}.

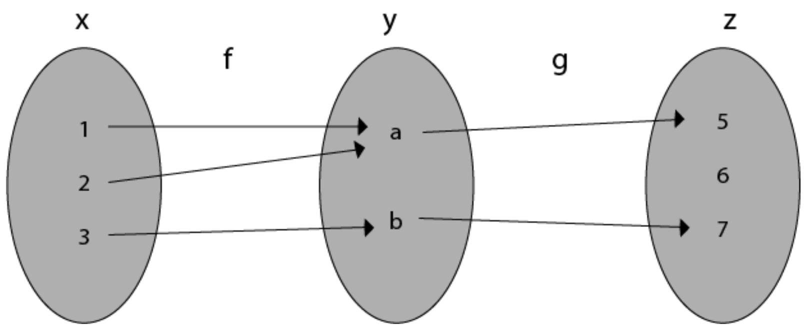
Consider the function f = {(1, a), (2, a), (3, b)} and g = {(a, 5), (b, 7)} as in figure. Find the composition of gof.

*Space for learners:*




Solution: The composition function gof is shown in fig:




(gof) (1) = g [f (1)] = g (a) = 5, (gof) (2) = g [f (2)] = g (a) = 5

(gof) (3) = g [f (3)] = g (b) = 7.

**Example:** Consider f, g and h, all functions on the integers, by f (n) =n2, g (n) = n + 1 and h (n) = n - 1.

Determine (i) hofog (ii) gofoh (iii) fogoh.

Solution:

(i) hofog (n) = n + 1,

hofog (n + 1) = (n+1)2

h [(n+1)2 ] = (n+1)2 - 1 = n2 + 1 + 2n - 1 = n2 + 2n.

(ii) gofoh (n) = n - 1, gof (n - 1) = (n-1)2

g [(n-1)2 ] = (n-1)2 + 1 = n2 + 1 - 2n + 1 = n2 - 2n + 2.

(iii) fogoh (n) = n - 1

fog (n - 1) = (n - 1) + 1

f (n) = n2.

Note:

- o If f and g are one-to-one, then the function (gof) (gof) is also one-to-one.
- o If f and g are onto then the function (gof) (gof) is also onto.

***Space for learners:***

- o Composition consistently holds associative property but does not hold commutative property.

Space for learners:

## 3.7 SUMMING UP

- A binary relation R between two non-empty sets P and Q is defined to be a subset of P x Q from a set P to Q.
- Domain of relation R is the set of elements in P which are related to some elements in Q, or it is the set of all first entries of the ordered pairs in R. Range of relation R is the set of elements in Q which are related to some element in P, or it is the set of all second entries of the ordered pairs in R.
- Relations can be represented in terms of matrix, Directed Graph, Table or Arrow Diagram.
- Relations may be of type reflexive, irreflexive, symmetric, asymmetric or transitive relations.
- A relation R on a set A is called an equivalence relation if R is reflexive, symmetric and transitive.
- A relation R on a set A is called a partial order relation if R is Reflexive, Antisymmetric and transitive.
- The set A together with a partial order relation R on the set A and is denoted by (A, R) is called a partial order set or POSET.
- A binary relation R on a set A is called circular if $(a, b) \in R$ and $(b, c) \in R$ implies $(c, a) \in R$.
- A mapping in which every element of set A is uniquely associated with the element of set B is called function. The set of A is called Domain of a function and set of B is called Co domain.
- Functions are of types injective, surjective, bijective and into and many more.

## 3.8 ANSWERS TO CHECK YOUR PROGRESS

**Answer 1:**

According to the given figure, P = {5, 6, 7}, Q = {3, 4, 5}

(i) R = {(x, y): y = x- 2; x ∈P} or R = {(x, y): y = x- 2 for x= 5, 6, 7}

(ii) R = {(5, 3), (6, 4), (7, 5)}

Domain of R = {5, 6, 7}

Range of R = {3, 4, 5}

**Answer 2:**

A = {1, 2, 3, 4, 6}, R = {(a, b): a, b ∈A, bis exactly divisible by a}

(i) R = {(1, 1), (1, 2), (1, 3), (1, 4), (1, 6), (2, 2), (2, 4), (2, 6), (3, 3), (3, 6), (4, 4), (6, 6)}

(ii) Domain of R = {1, 2, 3, 4, 6}

(iii) Range of R = {1, 2, 3, 4, 6}

**Answer 3:**

R = {(x, x+ 5): x ∈{0, 1, 2, 3, 4, 5}}

∴ R = {(0, 5), (1, 6), (2, 7), (3, 8), (4, 9), (5, 10)}

∴Domain of R = {0, 1, 2, 3, 4, 5}

Range of R = {5, 6, 7, 8, 9, 10}

**Answer 4:**

(i) {(2, 1), (5, 1), (8, 1), (11, 1), (14, 1), (17, 1)}

Since 2, 5, 8, 11, 14, and 17 are the elements of the domain of the given relation having their unique images, this

relation is a function.

Here, domain = {2, 5, 8, 11, 14, 17} and range = {1}

(ii) {(2, 1), (4, 2), (6, 3), (8, 4), (10, 5), (12, 6), (14, 7)}

Since 2, 4, 6, 8, 10, 12, and 14 are the elements of the domain of the given relation having their unique images, this

relation is a function.

Here, domain = {2, 4, 6, 8, 10, 12, 14} and range = {1, 2, 3, 4, 5, 6, 7}

***Space for learners:***

(iii) {(1, 3), (1, 5), (2, 5)}

Since the same first element i.e., 1 corresponds to two different images i.e., 3 and 5, this relation is not a function.

## 3.9 POSSIBLE QUESTIONS

**Short answer type questions:**

1. Which of these is not a type of relation?

a) Reflexive

b) Surjective

c) Symmetric

d) Transitive

2. An Equivalence relation is always symmetric.

a) True

b) False

3. Which of the following relations is symmetric but neither reflexive nor transitive for a set A = {1, 2, 3}.

a) R = {(1, 2), (1, 3), (1, 4)}

b) R = {(1, 2), (2, 1)}

c) R = {(1, 1), (2, 2), (3, 3)}

d) R = {(1, 1), (1, 2), (2, 3)}

4. Which of the following relations is transitive but not reflexive for the set S={3, 4, 6}?

a) R = {(3, 4), (4, 6), (3, 6)}

b) R = {(1, 2), (1, 3), (1, 4)}

c) R = {(3, 3), (4, 4), (6, 6)}

d) R = {(3, 4), (4, 3)}

**Space for learners:**

5. Let R be a relation in the set N given by R={(a,b): a+b=5, b>1}. Which of the following will satisfy the given relation?

a) (2,3) ∈ R

b) (4,2) ∈ R

c) (2,1) ∈ R

d) (5,0) ∈ R

6. Which of the following relations is reflexive but not transitive for the set T = {7, 8, 9}?

a) R = {(7, 7), (8, 8), (9, 9)}

b) R = {(7, 8), (8, 7), (8, 9)}

c) R = {0}

d) R = {(7, 8), (8, 8), (8, 9)}

7. Let I be a set of all lines in a XY plane and R be a relation in I defined as R = {(I1, I2):I1 is parallel to I2}. What is the type of given relation?

a) Reflexive relation

b) Transitive relation

c) Symmetric relation

d) Equivalence relation

8. Which of the following relations is symmetric and transitive but not reflexive for the set I = {4, 5}?

a) R = {(4, 4), (5, 4), (5, 5)}

b) R = {(4, 4), (5, 5)}

c) R = {(4, 5), (5, 4)}

d) R = {(4, 5), (5, 4), (4, 4)}

9. (a,a) ∈ R, for every a ∈ A. This condition is for which of the following relations?

a) Reflexive relation

***Space for learners:***

b) Symmetric relation

c) Equivalence relation

d) Transitive relation

10. $(a_1, a_2)$ ∈R implies that $(a_2, a_1)$ ∈ R, for all $a_1, a_2$ ∈A. This condition is for which of the following relations?

a) Equivalence relation

b) Reflexive relation

c) Symmetric relation

d) Universal relation

**Long answer type questions:**

1. Let A = {1, 3, 5, 7} and B = {p, q, r}. Let R be a relation from A into B defined by R = {(1, p), (3, r), (5, q), (7, p), (7, q)} find the domain and range of R.

2. Let A = {2, 4, 6} and B = {x, y, z}.

State which of the following are relation from A into B

(i) $R_1$ = {(2, x), (y, 4), (6, z)}

(ii) $R_2$ = {(4, y) (y, 4)}

(iii) $R_3$ = {(2, x) (4, y) (6, z)}

3. Let A = {3, 4, 5, 6} B = {1, 2, 3, 4, 5, 6} Let R = {(a, b) : a ∈ A, b ∈ B and a < b}.

Write R in the roster form. Find its domain and range.

4. Let A = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10} Let R be A relation on A defined by R = {a, b} : a ∈ A, b ∈ A, a is a multiple of b}. Find R, domain of R, range of R.

5. Determine the range and domains of the relation R defined by R = {(x - 1), (x + 2) : x ∈ (2, 3, 4, 5)}

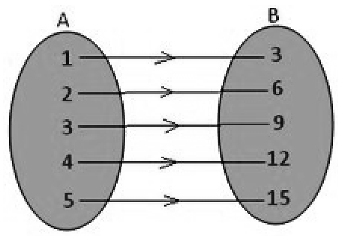
6. Let A = {1, 2, 3, 4, 5, 6} Define a relation R from A to A by R {(x, y) : y = x + 2}

- Depict this relation using an arrow diagram.
- Write down the domain and range of R

Space for learners:

7. The adjoining figure shows a relation between the set A and B. Write this relation in

find domain and range




(i) Set builder form.

(ii) Roster form.

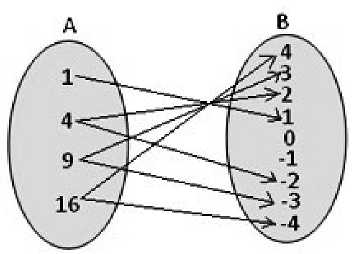
(iii) Find domain and range.

8. If A = {1, 4, 9, 16} and B = {1, 2, 3} Let R be a relation 'is square of' from A to B.

Find R domain and range of R.

9. Let A = {3, 4, 5} and B = {6, 8, 9, 10, 12}. Let R be the relation 'is a factor of' from A to B. Find R.

10. Adjoining figure shows relation between A and B. Write relation in




Range of a set

(i) Set builder form.

(ii) Roster form.
(iii) Find domain and range of R.

## 3.10 REFERENCES AND SUGGESTED READINGS

- Sets Relations Functions – by Gunther Gedia.
- Sets Relations Functions A programmed unit in modern mathematics – by Myra McFadden.

***Space for learners:***

- Concrete on the relation and function – by Jane Tennitope.
- Set theory – Charles C Printer.
- Naïve set theory – Paul Halmour.

***Space for learners:***

# UNIT 4: BOOLEAN ALGEBRA

Space for learners:

**Unit Structure:**

## 4.1 INTRODUCTION

Space for learners:

In this unit we will learn Boolean algebra which was developed by George Boole (1815-1864) a logician, to examine a given set of propositions (statements) with a view to checking their logical consistency and simplifying them by removing redundant statements or clauses. He used symbols to represent simple propositions. Compound propositions were expressed in terms of these symbols and connectives. Again, we will learn various properties of Boolean algebra with their proof. We will learn Boolean expression and Principle of Duality, how we can convert one Boolean expression to another. We will learn how we can simplify various Boolean expressions by using the Boolean properties.

We will learn Boolean function, literals, Minterms and Maxterm. Again, We will learn the truth table of Conjunction and Disjunction operation. we will learn the two most important canonical forms of Boolean algebra, Sum of Minterms (SOM) and Product of Maxterm (POM). We will learn how to write the simplified output expression of an Boolean function in SOM and POM form by using Boolean Identities as well as truth table. And finally we will learn the application of Boolean Algebra.

## 4.2 UNIT OBJECTIVES

After going through this unit, you will be able to:

- define Boolean algebra
- know about duality principle of Boolean algebra
- Know various properties of Boolean algebra with proof
- define Boolean expression and how to simplify it
- define literal, Minterm and Maxterm.
- define Boolean function
- define Conjunction, Disjunction and Complement operation
- define the canonical form SOM and POM
- know how to convert output expression into SOM and POM
- know the application of Boolean Algebra

## 4.3 BOOLEAN ALGEBRA

Space for learners:

Definitions: A non-empty set B with two binary operations $\vee$ and $\wedge$, a unary operation $'$, and two distinct elements 0 and I is called a Boolean Algebra if the following axioms holds for any elements a, b, c $\in$ B

**[B1]: Commutative Laws:**

$a \vee b = b \vee a$ and $a \wedge b = b \wedge a$

**[B2]: Distributive Law:**

$a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$ and $a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$

**[B3]: Identity Laws:**

$a \vee 0 = a$ and $a \wedge I = a$

**[B4]: Complement Laws:**

$a \vee a' = I$ and $a \wedge a' = 0$

We shall call 0 as zero element, 1 as unit element and $a'$ the complement of a.

We denote a Boolean Algebra by $(B, \vee, \wedge, \sim, 0, I)$

**Example 1**. Let,$D_6 = \{1, 2, 3, 6\}$ has four element. Define $\vee$ , $\wedge$ and $'$ on $D_6$ by $a \vee b = \text{lcm}(a, b)$ , $a \wedge b = \gcd(a, b)$ and $a' = 6/a$ . Then $D_6$ is a Boolean Algebra with 1 as the zero element and 6 as the unit element.

**Solution** :

We prepare the following tables for the operations $\vee$ , $\wedge$ , $'$

Table for operation ($\vee$ )

| $\vee$ | 1 | 2 | 3 | 6 |
|---|---|---|---|---|
| 1 | 1 | 2 | 3 | 6 |
| 2 | 2 | 2 | 6 | 6 |
| 3 | 3 | 6 | 3 | 6 |
| 6 | 6 | 6 | 6 | 6 |

Table for operation ($\wedge$)

| $\wedge$ | 1 | 2 | 3 | 6 |
|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 |
| 2 | 1 | 2 | 1 | 2 |

| | | | | |
|---|---|---|---|---|
| 3 | 1 | 1 | 3 | 3 |
| 6 | 1 | 2 | 3 | 6 |

Table for operation (' )

| ' | 1 | 2 | 3 | 6 |
|---|---|---|---|---|
| | 6 | 3 | 2 | 1 |

we observe that all the entries in the tables are element of $D_6$ . Therefore '$\vee$ ' and '$\wedge$ ' are binary operations on set $D_6$ . Also , ' ' ' is a unary operation on $D_6$ .

we observe the following properties -

**Commutativity :**

The entries in the composition tables for $\vee$ and $\wedge$ are symmetric about the diagonal starting from the upper left corner. Therefore, $\vee$ and $\wedge$ are commutative binary operations on $D_6$.

**Distributivity :**

From the composition tables of $\vee$ and $\wedge$ , we have

$1\vee(2\wedge 3) = 1\vee 1 = 1$ and $(1\vee 2)\wedge(1\vee 3) = 2\wedge 3 = 1$

$\therefore 1\vee(2\wedge 3) = (1\vee 2)\wedge(1\vee 3)$

Similarly,

$1\vee(2\wedge 6) = (1\vee 2)\wedge(1\vee 6)$

$1\vee(3\wedge 6) = (1\vee 3)\wedge(1\vee 6)$

$2\vee(3\wedge 6) = (2\vee 3)\wedge(2\vee 6)$ etc.

Thus, $\vee$ is distributive over $\wedge$

Also,

$1\wedge(2\vee 3) = 1\wedge 6 = 1$ and $(1\wedge 2)\vee(1\wedge 3) = 2\wedge 3 = 1$

$\therefore 1\wedge(2\vee 3) = (1\wedge 2)\vee(1\wedge 3)$

Similarly,

$1\wedge(2\vee 6) = (1\wedge 2)\vee(1\wedge 6)$

$1\wedge(3\vee 6) = (1\vee 3)\wedge(1\vee 6)$

$2\wedge(3\vee 6) = (2\wedge 3)\vee(2\wedge 6)$ etc.

Thus, $\wedge$ is distributive over $\vee$ .

Existence of identity elements :

***Space for learners:***

For binary operation '$\vee$' , we observe that the first row of the composition table coincides with the top most row and the first column coincides with the left most column. These two intersect at 1. so, 1 is the identity element for '$\vee$' . Similarly, 6 is the identity element for '$\wedge$ '.

Thus, 1 and 6 are respectively the zero and unit element.

Complement laws :

we have, $1\vee 1' =1 \vee 6 = 6$, $2 \vee 2' = 2 \vee 3 = 6$, $3\vee 3' = 3\vee 2 =6$, $6\vee 6' =6\vee 1=6$ $1\wedge 1'=1\wedge 6=1$, $2\wedge 2'=2\wedge 3=1$, $3\wedge 3' =3\wedge 2=1$, $6\wedge 6' =6\wedge 1=1$

$\therefore$ $1'= 6/1 =6$, $2' = 6/2 =3$, $3' = 6/3 =2$ , $6' = 6/ 6 =1$

Thus , the set $D_6$ with the given binary operations and a unary operation satisfies all the axioms of Boolean algebra. Hence , ( $D_6$ , '$\vee$ ' , '$\wedge$' , ' ' ' ) is a Boolean algebra .

## 4.4 PRINCIPLE OF DUALITY

By the dual of a proposition concerning a Boolean algebra B, we mean the proposition obtained by substituting $\vee$ for $\wedge$, $\wedge$for V, 0 for 1, and 1 for 0, i.e., by exchanging $\wedge$ and V, and exchanging 0 and. Any pair of expression satisfying this property is called Dual expression. Again, this characteristics of Boolean algebra is called the Principle of Duality.

For example, The dual of $x\wedge (y\vee z) = (x\wedge Y) V (x\wedge Z)$ is $xV (y\wedge Z) = (xVY) \wedge (xVZ)$, and vice versa.

## 4.5 PROPERTIES OF A BOOLEAN ALGEBRA

1. Idempotent Laws: (i) $a \vee a = a$
(ii) $a \wedge a = a$

2. Boundedness Laws: (i) $a \vee I = I$
(ii) $a \wedge 0 = 0$

3. Absorption Laws: (i) $a \vee (a \wedge b) = a$
(ii) $a \wedge (a \vee b) = a$

4. Associative Laws: (i) $(a \vee b) \vee c = a \vee (b \vee c)$
(ii) $(a \wedge b) \wedge c = a \wedge (b \wedge c)$

Space for learners:

5.De Morgan's Law: (i) $(a \vee b)' = a' \wedge b'$
(ii) $(a \wedge b)' = a' \vee b'$.

6.(i) $(a \vee b) = (a' \wedge b')'$ (ii) $(a \wedge b) = (a' \vee b')'$

7. (i) $a \wedge (a' \vee b) = (a \wedge b)$ (ii) $a \vee (a' \wedge b) = (a \vee b)$

Proof: It is sufficient to prove first part of each law since second part follows from the first by principle of duality.

1. (i). We have

$a = a \vee 0$ (by identity law in a Boolean algebra)

$= a \vee (a \wedge a')$ (by complement law)

$= (a \vee a) \wedge (a \vee a')$ (by distributive law)

$= (a \vee a) \wedge I$ (complement law)

$= a \vee a$ (identity law) ,

1(ii). We know that,

$a = a \wedge 1$(by identity law in a Boolean algebra)

$= a \wedge (a \vee a')$ (by complement law)

$= (a \wedge a) \vee (a \wedge a')$ (by distributive law)

$= (a \wedge a) \vee 0$ (complement law)

$= (a \wedge a)$ (identity law)

2(i) : We have

$a \vee I = (a \vee I) \wedge I$ (identity law)

$= (a \vee I) \wedge (a \vee a')$ (complement law)

$= a \vee (I \wedge a')$ (Distributive law)

$= a \vee a'$ (identity law)

$= I$ (complement law).

Space for learners:

2(ii) it is the dual of 2(ii)

3(i): we note that

$a \vee (a \wedge b) = (a \wedge I) \vee (a \wedge b)$ (identity law)

$= a \wedge (I \vee b)$ (distributive law)

$= a \wedge (b \vee I)$ (commutativity)

$= a \wedge I$ (Identity law)

$= a$ (identity law)

3(ii) it is the dual of 3(i)

4(i) Let,

$$L = (a \vee b) \vee c \qquad R = a \vee (b \vee c)$$

Then $a \wedge L = a \wedge [(a \vee b) \vee c]$

$= [a \wedge (a \vee b)] \vee (a \wedge c)$ (distributive Law)

$= a \vee (a \wedge c)$ (absorption law)

$= a$ (absorption law)

And

$a \wedge R = a \wedge [a \vee (b \vee c)]$

$= (a \wedge a) \vee (a \wedge (b \vee c)]$ (distributive law)

$= a \vee (a \wedge (b \vee c)]$ (idempotent law)

$= a$ (absorption Law)

Thus $a\wedge L = a \wedge R$ and so, by duality, $a \vee L = a \vee R$ .

Further, $a' \wedge L = a' \wedge [(a \vee b) \vee c]$

$= [a' \wedge (a \vee b)] \vee (a' \wedge c)$ (distributive law)

Space for learners:

$= [(a' \wedge a) \vee (a' \wedge b)] \vee (a' \wedge c)$ (distributive law)

$= [0 \vee (a' \wedge b)] \vee (a' \wedge c)$ (complement Law)

$= (a' \wedge b)] \vee (a' \wedge c)$ (Identity law)

$= a' \wedge (b \vee c)$ (distributive law)

On the other hand

$a' \wedge R = a' \wedge [a \vee (b \vee c)]$

$= (a' \wedge a) \vee [a' \wedge (b \vee c)]$ (distributive law)

$= 0 \vee [a' \wedge (b \vee c)]$ (complement law)

$= a' \wedge (b \vee c)]$ (identity law)

Hence $a' \wedge L = a' \wedge R$ and so by duality $a' \vee L = a' \vee R$

Therefore ,

$L = (a \vee b) \vee c$

$= 0 \vee [(a \vee b) \vee c] = 0 \vee L$ (identity law)

$= (a \wedge a') \vee [(a \vee b) \vee c]$

$= (a \wedge a') \vee L$ (complement law)

$= (a \vee L) \wedge (a' \vee L)$ (distributive law)

$= (a \vee R) \wedge (a' \vee R)$ (using $A \vee L = a \vee R$ and $a' \vee L = a' \vee R$]

$= (a \wedge a') \vee R$ (distributive law)

$= 0 \vee R$ (complement law)

$= R$ (identity law)

Hence, $(a \vee b) \vee c = a \vee (b \vee c)$

***Space for learners:***

4(ii). It is the dual of 4(i)

Space for learners:

5(i) . we have

$(a \vee b) \vee (a' \wedge b') = (b \vee a) \vee (a' \wedge b')$ (commutative)

$= b \vee (a \vee (a' \wedge b'))$ (associative)

$= b \vee [(a \vee a' \wedge (a \vee b')]$ (distributive)

$= b \vee [I \wedge (a \vee b')$ (complement)

$= b \vee (a \vee b')$ (identity)

$= b \vee (b' \vee a)$ (commutative)

$= (b \vee b') \vee a$ (associative law)

$= I \vee a$ (complement law)

$= I$ (Identity law)

Also, $(a \vee b) \wedge (a' \wedge b') = [(a \vee b) \wedge a'] \wedge b'$ (associativity)

$= [a \wedge a') \vee (b \wedge a')] \wedge b'$ (distributive law)

$= [0 \vee (b \wedge a')] \wedge b'$ (complement)

$= (b \wedge a') \wedge b'$ (identity)

$= b \wedge b' \wedge a'$

$= 0 \wedge a'$

$= 0$

Hence , $a' \wedge b'$ is complement of $a \vee b$,

i.e. $(a \vee b)' = a' \wedge b'$.

5(ii).It is the dual of 5(i)

6(i) We know that,

$(a \vee b)' = a' \wedge b'$ [By De Morgan's Law]

$(a \vee b)'' = (a' \wedge b')'$. [Taking Complement on both sides]

$(a \vee b) = (a' \wedge b')'$ [ Since, $a'' = a$]

6(ii). It is the dual of 6(i)

7(i)

LHS= $a \wedge (a' \vee b)$

$= (a \wedge a') \vee (a \vee b)$ [By Distributive Law]

$= 0 \vee (a \vee b)$ [By Complement Law]

$= (a \vee b)$ [Since, $0 \vee x = x$]

=RHS

7(ii) is the dual of 7(i)

## 4.5.1 Theorem

Uniqueness of the complement: If $x \vee y = 1$ and $x \wedge y = 0$, then $y = x'$.

Proof. We know that,

$y = y \vee 0$ [Since, $x \vee 0 = x$]

$= y \vee (X \wedge X')$[ by Complement law B4]

$= (y \vee x) \wedge (y \vee x')$ [by Distributive law B2]

$= (x \vee y) \wedge (y \vee x')$ [by Commutative Laws B1]

$= 1 \wedge (y \vee x')$[ by hypothesis]

$= (yVx') \wedge 1$[by Commutative Laws B1]

$= y \vee x'$ [Since, $x \wedge 1 = x$]

Again, $x' = x' \vee 0$ [Since, $x \vee 0 = x$]

$= x' \vee (x \wedge y)$ [by hypothesis]

$= (x' \vee x) \wedge (x' \vee y)$ [by Distributive law B2]

$= (x \vee x') \wedge (x' \vee y)$ [by Commutative Laws B1]

Space for learners:

$= 1 \wedge (x' \vee y)$ [by Complement law B4]

$= (x' \vee y) \wedge 1$ [By Complement law B4]

$= (x' \vee y)$ [since, $x \wedge 1 = x$]

$= (y \vee x')$ [by Commutative Laws B1]

$= y$ [Already proved that $y = y \vee x'$]

Hence, If $x \vee y = 1$ and $x \wedge y = 0$, then $y = x'$.

**CHECK TO YOUR PROGRESS**

1.Find the duals of the following Boolean Expression
(a) $x \vee y$ (b) $(a \vee b) \vee c = a \vee (b \vee c)$ (c) $a \wedge (a' \vee b) = (a \wedge b)$
(d) $(a \wedge b)' = a' \vee b'$

2. Let $D_{30}$ = { 1 , 2 , 3 , 5 , 6 , 10 , 15 , 30 } be the set of all divisors of 30 and let $\vee$ and $\wedge$ be two operations on B as defined below : $a \vee b$ = L C M of a and b , $a \wedge b$ = GCD of a and b. Also , for each $a \in B$ , let us define $a' = 30/a$ Then , show that ( $D_{30}$ , $\vee$ , $\wedge$ , ' ) is a Boolean algebra.

3.Let $D_8$ = { 1 , 2 , 4 , 8 } , be the of all divisors of 8 and let $\vee$ and $\wedge$ be two operations be defined on $D_8$ as follows : $a \vee b$ = L C M of a and b, $a \wedge b$ = GCD of a and b, $a' = 8/a$ show that ( $D_8$ , $\vee$,$\wedge$ , ' ) is not a Boolean algebra

4. Let B = { Φ , {1}, {2} , { 1,2} } = power set of set{1,2}. Show that ( B, $\cup$ ,$\cap$ , ' ) is a Boolean algebra.

5.Let B={ Φ , {1}, {2}, {3} , { 1,2} , { 1,3} , {2,3} , {1,2,3} }= power set of set {1,2,3} . Show that( B, $\cup$ ,$\cap$ , ' ) is a Boolean algebra.

## 4.6 BOOLEAN EXPRESSION

Space for learners:

Let (A, ∧, ∨,') be a Boolean algebra. Then expression involving members of A and the operations ∧,∨ and complementation are called Boolean expression or Boolean Polynomials.

Let $x_1, x_2, \ldots, x_n$ be a set of n variables (or letters or symbols). A Boolean Polynomial (Boolean expression, Boolean form or Boolean formula) $p(x_1, x_2, \ldots, x_n)$ in the variables $x_1, x_2, \ldots, x_n$ is defined recursively as follows:

1. The symbols 0 to 1 are Boolean polynomials
2. $x_1, x_2, \ldots, x_n$ are all Boolean polynomials
3. if $p(x_1, x_2, \ldots, x_n)$ and $q(x_1, x_2, \ldots, x_n)$ are two Boolean polynomials, then

$p(x_1, x_2, \ldots, x_n) \vee q(x_1, x_2, \ldots, x_n)$ and $p(x_1, x_2, \ldots, x_n) \wedge q(x_1, x_2, \ldots, x_n)$ are also Boolean polynomials.

4. If $p(x_1, x_2, \ldots, x_n)$ is a Boolean polynomial, then

$(p(x_1, x_2, \ldots, x_n))'$ is also Boolean polynomials

5. There are no Boolean polynomials in the variables $x_1, x_2, \ldots, x_n$ other than those obtained in accordance with rules 1 to 4.

For example, for variables x, y and z ,

the expressions $p1(x, y, z) = (x \vee y) \wedge z$

$p2 (x, y, z) = (x \vee y') \vee (y \wedge 1)$

$p3(x, y, z) = (x \vee (y' \wedge z)) \vee (x \wedge (y \wedge 1))$ are Boolean expressions.

**Note:** A Boolean expression of n variables there may or may not contain all the n variables.

## 4.6.1 Minimization of Boolean Expression

1.Simplify the following Boolean Expression

(a) $(A'\wedge B'\wedge C') \vee (A'\wedge B\wedge C') \vee (A\wedge B'\wedge C') \vee (A\wedge B\wedge C')$

(b) $(A\wedge B) \vee (A\wedge C)' \vee [(A\wedge B'\wedge C) \wedge (A\wedge B) \vee C]$

**Solutions:**

1. (a)

$(A'\wedge B'\wedge C') \vee (A'\wedge B\wedge C') \vee (A\wedge B'\wedge C') \vee (A\wedge B\wedge C')$

**Space for learners:**

$= [( A' \wedge C') \wedge( B' \vee B)] \vee [ (A \wedge C' ) \wedge ( B' \vee B) ]$ [Distributive law]

$= [( A' \wedge C') \wedge 1 ] \vee [(A \wedge C' ) \wedge 1 ]$ [Since, $B' \vee B =1$]

$= ( A' \wedge C') \vee (A \wedge C' )$ [Since $A \wedge 1= A$]

$= C' \wedge(A' \vee A)$ [Distributive law]

$= C' \wedge 1$ [Since, $B' \vee B =1$ ]

$= C'$ [Since $A \wedge 1= A$]

1. (b)

$(A\wedge B) \vee (A\wedge C)' \vee [ (A\wedge B'\wedge C) \wedge ( A\wedge B )\vee C]$

$= (A\wedge B) \vee (A\wedge C)' \vee (A\wedge B'\wedge C \wedge A\wedge B )\vee (A\wedge B'\wedge C\wedge C)$

$= (A\wedge B) \vee (A' \vee C' ) \vee 0 \vee (A\wedge B'\wedge C)$ [By De Morgans law and $B\wedge B'=0, C\wedge C=C$]

$=(A\wedge B) \vee (A' \vee C' ) \vee (A\wedge B'\wedge C)$

$= (A\wedge B) \vee (A' \vee C' ) \vee( B'\wedge C)$ [Since, $A\vee (A\wedge B)=A\vee B$]

$= A' \vee (A\wedge B) \vee C' \vee( C \wedge B')$

$= A' \vee B \vee C' \vee B'$

$= A' \vee C' \vee 1$ [since, $B \vee B'=1$]

$= A' \vee 1$ [Since, $C' \vee 1 =1$]

$=1$

## 4.7 BOOLEAN FUNCTION

Each Boolean expression represents a Boolean function. Any function specifying a Boolean expression is called a Boolean function.

Let, $B=\{0,1\}$.Then $B_n=\{ (x_1,x_2,\ldots\ldots x_n) | x_i \in B \text{ for } 1\leq i \leq n \}$ is the set of all possible n-tuples of 0s and 1s.The variable x is called a Boolean variable if it assumes values only from B,that is,if its only possible values are 0 and 1. A function from $B_n$ to B is called a Boolean function of degree n.

Thus if $f(x,y)=x\wedge y$,then f is the Boolean function and $x\wedge y$ is the boolean expression (or the value of the function f)

*Space for learners:*

## 4.8 CONJUNCTION (∧) OPERATION

The Conjunction or Boolean product of two variables x and y, which is denoted by xy or x∧y gives a value 1 when both x and y have the value 1 and the value 0 otherwise.

| X | y | x ∧y |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

## 4.9 DISJUNCTION (∨) OPERATION

The Disjunction or Boolean sum of two variables x and y, which is denoted by x+y or x∨y gives a value 1 when either x or y or both has the value 1 and the value 0 otherwise

| X | y | x ∨y |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

## 4.10 COMPLEMENTATION (′)

It is an expression with the value 1 when x has the value 0 and the value 0 when x has the value 1.

**Examples1**: Find the values of Boolean function f(x,y)=x∧y′

**Examples2**: Find the values of Boolean function f(x, y, z)=(x∧y) ∨ z′

**Solution1**:It is a Boolean function of two variables. The values are displayed in the table given below

| X | y | y′ | f(x,y)= x∧y′ |
|---|---|---|---|
| 1 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 |

Space for learners:

| | | | |
|---|---|---|---|
| 0 | 1 | 0 | 0 |
| 1 | 1 | O | 0 |

**Solution2**: It is a Boolean function of 3 variables. The values are displayed in the table given below:

| X | y | z | $x \wedge y$ | $z'$ | $f(x,y,z)=(x \wedge y) \vee z'$ |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 1 |

Space for learners:

## 4.11 DEFINITION OF LITERAL

A literal is a Boolean variable or complemented variable such as x, x′, y, y′, and so on.

## 4.12 FUNDAMENTAL PRODUCT OR MINTERM

A fundamental product is a literal or a product of two or more literal in which no two literals involve the same variable. Fundamental product is also called a minterm or complete product.

A minterm in n variable is a product of n literals in which each variable is represented by the variable itself or its complement.

A minterm of the Boolean variables $x_1,x_2,\ldots x_n$ is a Boolean product $y_1 \wedge y_2 \wedge y_3 \ldots \wedge y_n$, where $y_i=x_i$ or $y_i=x'_i$

For example, for a 3 variable Boolean function there are 8 nos of possible minterms, which are

$x \wedge y \wedge z$, $x' \wedge y \wedge z$, $x \wedge y' \wedge z$, $x \wedge y \wedge z'$

$x \wedge y' \wedge z'$ $x' \wedge y \wedge z'$ $x' \wedge y' \wedge z$ $x' \wedge y' \wedge z'$

## 4.13 DEFINITION OF MAXTERM

A maxterm in n variable is a sum of n literals in which each variable is represented by the variable itself or its complement.

A maxterm of the Boolean variables $x_1, x_2, \ldots x_n$ is a Boolean sum $y_1 \vee y_2 \vee y_3 \ldots \vee y_n$, where $y_i = x_i$ or $y_i = x'_i$

For example, for a 3 variable Boolean function there are 8 nos of possible maxterms, which are

$x \vee y \vee z$, $x' \vee y \vee z$, $x \vee y' \vee z$, $x \vee y \vee z'$

$x \vee y' \vee z'$ $x' \vee y \vee z'$ $x' \vee y' \vee z$ $x' \vee y' \vee z'$

Space for learners:

## 4.14 CANONICAL FORM OR NORMAL FORM

A Boolean function can be uniquely described by its truth table or in one of the canonical forms.Two dual canonical forms are-

(1) The sum of Minterms(SOM) or Sum of Product(SOP) or Disjunctive normal form(DNF)

(2) The Product of Maxterm or Product of Sum(POS) or Conjunctive Normal form(CNF)

### 4.14.1 Sum of Minterms (SOM) / Sum of Products (SOP) / Disjunctive Normal Form (DNF)

A Boolean function(expression) is said to be in Disjunctive normal form in n variables $x_1, x_2, \ldots\ldots x_n$ if it can be written as join(Sum) of terms of the type $f_1(x_1) \wedge f_2(x_2) \wedge \ldots\ldots f_n(x_n)$ where $f_i(x_i)$ or $x'_i$ for all $i=1,2,\ldots.n$ and no two terms are same. Also 1 and 0 are said to be in disjunctive normal form.

Here, $f_1(x_1) \wedge f_2(x_2) \wedge \ldots\ldots f_n(x_n)$ are called minterms or minimal polynomials.

For example, $(x \wedge y \wedge z') \vee (x \wedge y' \wedge z') \vee (x' \wedge y \wedge z')$ is a Boolean expression in SOM form.

## 4.14.2 Rules for Converting Output Expression into SOM

Space for learners:

There are two ways by which we can convert the output expression in SOM form-

**First Way**

(a) Examine each term in the given logic function. Retain if it is a minterm; continue to examine the next term in the same manner.

(b) Check for variables that are missing in each product, which is not a minterm. Multiply(∧) the product by $(x \vee x')$ term, for each variable x that is missing.

(c) Multiply(∧) all the products and eliminate the redundant term.

**Second Way:** Procedure for obtaining the output expression in SOM from a truth table:

1. Give a product term for each input combination in the table, containing an output value of 1.

2. Each product term contains its input variables in either complemented or uncomplemented form.

3.All the product terms are Summed (∨)together in order to produce the final SOM expression of the output.

***Example***
: *Find the SOM expansion for the function* $f(x, y, z) = (x \vee y) \wedge z'$

**Solution:** We will find the SOM expansion of $f(x, y, z)= (x \vee y) \wedge z'$ in two ways.

**First way:** By using Boolean identities

Given,

$f(x,y,z)= (x \vee y) \wedge z'$

$= (x \wedge z') \vee (y \wedge z')$ [Distributive law]

$= (x \wedge 1 \wedge z') \vee (1 \wedge y \wedge z')$ [Identity law]

$= [x \wedge (y \vee y') \wedge z')] \vee [(x \vee x') \wedge y \wedge z')]$ [complement law]

$= (x \wedge y \wedge z') \vee (x \wedge y' \wedge z') \vee (x \wedge y \wedge z') \vee (x' \wedge y \wedge z')$ [Distributive law]

$=(x \wedge y \wedge z') \vee (x \wedge y' \wedge z') \vee (x' \wedge y \wedge z')$ [By idempotent law]

**Second Way:**

we can construct the sum of Minterm expansion by determining the values of for all possible values of the variables x, y and z.

| X | Y | Z | $x \vee y$ | $z'$ | $f(x, y, z)=(x \vee y) \wedge z'$ | Minterm |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 0 | 0 | |
| 1 | 1 | 0 | 1 | 1 | 1 | $x \wedge y \wedge z'$ |
| 1 | 0 | 1 | 1 | 0 | 0 | |
| 1 | 0 | 0 | 1 | 1 | 1 | $x \wedge y' \wedge z'$ |
| 0 | 1 | 1 | 1 | 0 | 0 | |
| 0 | 1 | 0 | 1 | 1 | 1 | $x' \wedge y \wedge z'$ |
| 0 | 0 | 1 | 0 | 0 | 0 | |
| 0 | 0 | 0 | 0 | 1 | 0 | |

Now, the Sum of Product expansion of f is the Boolean Sum($\vee$) of the three minterms corresponding to the three rows of the table that give the value 1 for the function.

Therefore, $f(x, y, z)= (x \wedge y \wedge z') \vee (x \wedge y' \wedge z') \vee (x' \wedge y \wedge z')$

**Example1**. Convert the function into SOM form

$[ ( x\wedge y') \vee z' ] \wedge ( x' \vee z) '$

**Solution:**

$[ ( x\wedge y') \vee z' ] \wedge ( x' \vee z) '$

$= [ ( x' \vee y'') \vee z' ] \wedge ( x' \vee z) '$ [By De Morgan's Law]

$= [ ( x' \vee y \vee z' ] \wedge (x' \vee z ) '$ [ Since, $y''=y$]

$= [ ( x' \vee y \vee z' ] \wedge ( z \vee x' ) '$ [By Commutative law]

$= [ ( x' \vee y \vee z' ] \wedge ( z' \wedge x'' )$ [By De Morgan's Law]

$= [ ( x' \vee y \vee z' ] \wedge ( z' \wedge x )$ [since $x''=x$]

$= ( x' \wedge z' \wedge x ) \vee ( y \wedge z' \wedge x ) \vee (z' \wedge z' \wedge x )$ [By distributive law]

$= 0 \vee ( y \wedge z' \wedge x ) \vee (z' \wedge z' \wedge x )$ [since $x \wedge x'=0$]

$=0 \vee ( y \wedge z' \wedge x ) \vee (z' \wedge x )$ [Since $z' \wedge z' = z'$, Idempotent Laws]

Space for learners:

$= (x \wedge y \wedge z') \vee (z' \wedge x)$ [Since $0 \vee a = a$]

$= (x \wedge y \wedge z') \vee [(z' \wedge x) \wedge 1]$ [Since, $a \wedge 1 = 1$]

$= (x \wedge y \wedge z') \vee [(z' \wedge x) \wedge (y \vee y')]$ [Since, $a \vee a' = 1$]

$= (X \wedge y \wedge z') \vee [(z' \wedge x \wedge y) \vee (z' \wedge x \wedge y')]$ [By distributive law]

$= (x \wedge y \wedge z') \vee [(x \wedge z' \wedge y) \vee (x \wedge z' \wedge y')]$

$= (x \wedge y \wedge z') \vee (x \wedge y' \wedge z')$

### 4.14.3 Product of Maxterms (POM) / Product of Sums (POS) / Conjunctive Normal Form (CNF)

A Boolean function(expression) is said to be in Conjunctive normal form in n variables $x_1, x_2, \ldots\ldots x_n$ if it can be written as meet(Product) of terms of the type $f_1(x_1) \vee f_2(x_2) \vee \ldots\ldots f_n(x_n)$ where $f_i(x_i)$ or $x'_i$ for all $i=1,2,\ldots.n$ and no two terms are same.Also 1 and 0 are said to be in disjunctive normal form.

Here, $f_1(x_1) \vee f_2(x_2) \vee \ldots\ldots f_n(x_n)$ are called Maxterms or maximal polynomials

For example,

$(x \vee y \vee z') \wedge (x \vee y' \vee z') \wedge (x' \vee y \vee z')$ is a boolaen expression in POM form

### 4.14.4 Rules for Converting the Output Expression into POM

There are two ways by which we can convert the output expression in POM form

**First Way**

(a) Examine each term in the given logic function. Retain if it is a Maxterm; continue to examine the next term in the same manner.

(b) Check for variables that are missing in each sum, which is not a maxterm. Add $(x \wedge x')$ to the sum term,for each variable x that is missing.

(c) Expand the expression using the distributive property and eliminate the redundant term.

**Second Way**

Space for learners:

The procedure for obtaining the output expression of a Boolean function in POM form from a truth table.

(a) Give a sum term for each input combination in the table,which has an output value 0.

(b) Each sum terms contains all its input variables in complemented or uncomplemented form. If the input variable is 0, then it appears in an uncomplemented form; if the input variable is 1, it appears in the complemented form.

(c) All the sum terms are AND operated (∧) together to obtain the final POM expression.

Example: Convert the function in POM form

$$Y = A \vee (B' \wedge C)$$

**Solution:**

**First way:**

Here,

$Y = A \vee (B' \wedge C)$

$= (A \vee B') \wedge (A \vee C)$ [By Distributive law]

$= (A \vee B' \vee 0) \wedge (A \vee C \vee 0)$ [Since $a \vee 0 = a$]

$= [A \vee B' \vee (C \wedge C')] \wedge [A \vee C \vee (B \wedge B')]$ [Since, $a \wedge a' = 0$]

$= (A \vee B' \vee C) \wedge (A \vee B' \vee C') \wedge (A \vee C \vee B) \wedge (A \vee C \vee B')$ [ Distributive law]

$= (A \vee B \vee C) \wedge (A \vee B' \vee C) \wedge (A \vee B' \vee C')$ [Since, $A \wedge A = A$]

**Second Way:**

| A | B | C | $B'$ | $B' \wedge C$ | $Y = A \vee (B' \wedge C)$ | Maxterm |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | 0 | $(A \vee B \vee C)$ |
| 0 | 0 | 1 | 1 | 1 | 1 | |
| 0 | 1 | 0 | 0 | 0 | 0 | $(A \vee B' \vee C)$ |
| 0 | 1 | 1 | 0 | 0 | 0 | $(A \vee B' \vee C')$ |
| 1 | 0 | 0 | 1 | 0 | 1 | |
| 1 | 0 | 1 | 1 | 1 | 1 | |
| 1 | 1 | 0 | 0 | 0 | 1 | |
| 1 | 1 | 1 | 0 | 0 | 1 | |

From the truth table, we have seen that for the given 3-input function we find the Y value is 0 for the input combinations 000,010 and 011

***Space for learners:***

and their corresponding Maxterms are $(A\vee B\vee C)$ , $(A\vee B'\vee C)$ and $(A\vee B'\vee C')$

Therefore, the required POM form is

$(A\vee B\vee C)\wedge(A\vee B'\vee C)\wedge(A\vee B'\vee C')$

Space for learners:

**CHECK TO YOUR PROGRESS**

6.Simplify the following Boolean Expression

(a) $(A\vee B\vee C)\wedge(A\vee B'\vee C')\wedge(A\vee B\vee C')\wedge(A\vee B'\vee C)$

(b) $(A\wedge B)\vee(B\wedge B)\vee C\vee B'$

(c) $A\vee(A'\wedge B)\vee(A'\wedge B'\wedge C)\vee(A'\wedge B'\wedge C'\wedge D)$

7. Obtain the Canonical SOM expression for the function $Y(A,B)=A\vee B$

8. Obtain the Canonical SOM expression for the function $Y(A,B,C)=A\vee(B\wedge C)$

9. Obtain the Canonical POM expression for the function

$Y(A,B,C)=(A\vee B')\wedge(B\vee C)\wedge(A\vee C')$

10. Obtain the Canonical POM expression for the function

$F(x,y,z)=(X\vee Z)\wedge Y$

## 4.15 APPLICATION OF BOOLEAN ALGEBRA

Boolean algebra is useful in designing switching circuits. Subsequently we will use Boolean algebra to design logic circuits for logical and arithmetic operations performed by processors.

Boolean Algebra of Switching circuits:

Let B={ 0 ,1}, where 0 and 1 denote the two mutually exclusive states , off and on, of a switch respectively .

Let the operations of connecting the switches in parallel and connecting the switches in series be denoted by + and . respectively.

Let $0'=1$ and $1'=0$. Then, [ B , ∨ , ∧ , ‘ ] is a Boolean algebra, known as the Boolean algebra of switching circuits

The composition tables for the above operations are given below:

| ∨ | 0 | 1 |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 1 | 1 |

| ∧ | 0 | 1 |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 1 | 1 |

| ' | |
|---|---|
| 0 | 1 |
| 1 | 0 |
| | |

Boolean switching circuit: An arrangement of wires and switches formed by the repeated use of a combination of switches in parallel and series is called a Boolean switching circuit.

Equivalent switching circuits: Two switching circuits A and B, are said to be equivalent, denoted by A ~B if both are in the same state for the same states of their constituent switches. Thus, two switching circuits are said to be equivalent if and only if their corresponding Boolean functions are equal. This happens when their Boolean function have the same value, o or 1, for every possible assignment of the values o and 1 to their variables

## 4.16 SUMMING UP

- A non-empty set B with two binary operations ∨ and ∧, a unary operation ′, and two distinct elements 0 and 1 is called a Boolean Algebra if commutative, distributive, identity and complement properties hold for any elements a, b, c ∈ B.
- By the dual of a proposition concerning a Boolean algebra B, we mean the proposition obtained by substituting ∨ for ∧, ∧for V, 0 for 1, and 1 for 0, i.e., by exchanging ∧ and V, and exchanging 0 and. Any pair of expression satisfying this property is called Dual expression.



*Space for learners:*

Space for learners:

- Let (A, ∧, ∨,’) be a Boolean algebra. Then expression involving members of A and the operations ∧, ∨ and complementation are called Boolean expression or Boolean Polynomials.
- Any function specifying a Boolean expression is called a Boolean function. A literal is a Boolean variable or complemented variable such as x, x′, y, y′, and so on.
- A minterm in n variable is a product of n literals in which each variable is represented by the variable itself or its complement. A maxterm in n variable is a sum of n literals in which each variable is represented by the variable itself or its complement.
- Boolean algebra is useful in designing switching circuits. Subsequently we will use Boolean algebra to design logic circuits for logical and arithmetic operations performed by processors.

## 4.17 ANSWERS TO CHECK YOUR PROGRESS

1. (a) $x \wedge y$ (b) $(a \wedge b) \wedge c = a \wedge (b \wedge c)$

(c) $a \vee (a' \wedge b) = (a \vee b)$

(d) $(a \vee b)' = a' \wedge b'$

2. The given operations on $D_{30}$ satisfy the following properties:

(a) Closure properties

Let, a and b be any two arbitrary elements of $D_{30}$. Then, each one of a and b is a divisor of 30, that means LCM of a and b is a divisor of 30 and HCF of a and b is a divisor of 30.

So, for all a and b,

$$(a \vee b) \in D_{30} \text{ and } (a \wedge b) \in D_{30}.$$

So, $D_{30}$ is closed for each of the operations ∨and ∧

( b) Commutative laws:

Let, a and b be any two arbitrary elements of $D_{30}$

Then, LCM of a and b= LCM of b and a

So, for all a, b $\in D_{30}$, $a \vee b = b \vee a$,

And, HCF of a and b=HCF of b and a

So, for all a, b $\in D_{30}$, $a \wedge b = b \wedge a$

Hence, the given Boolean algebra follows Commutative law

(c). Associative laws:

Let a, b, c be arbitrary elements of B.

(i) LCM [{LCM (a, b)} and c] = LCM [a and {LCM (b, c)}]

$\Rightarrow (a \vee b) \vee c = a \vee (b \vee c)$ for all $a, b, c \in D_{30}$

(ii) HCF [{HCF (a, b)} and c] = HCF [a and {HCF (b, c)}]

$\Rightarrow (a \wedge b) \wedge c = a \wedge (b \wedge c)$ for all $a, b, c \in D_{30}$

So, it follows the Associative law

(d). Distributive laws:

Let a and b be any two arbitrary elements of $D_{30}$

Then, we know that HCF is distributive over LCM, and LCM is distributive over HCF

(i) $a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$ for all $a, b, c \in D_{30}$ . [Distributive law of HCF over LCM]

(ii) $a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$ [ distributive law of LCM over HCF]

(e). Existence of identity elements

Clearly, $1 \in D_{30}$ and $30 \in D_{30}$

Such that (i) $a \vee 1$ = LCM (a and 1) = a for all $a \in D_{30}$

. (ii) $a \wedge 30$=HCF(a and 30)=a for all $\in D_{30}$

This shows that 1 is the identity element for $\vee$ and 30is the identity element for $\wedge$

(f). Existence of complement

For each $a \in D_{30}$

Let us define its complement$a' = 30/a$

Then, we have (i) $(a \vee a')$=LCM (a, 30/a) =30

(ii) $a \wedge a'$ =HCF(a, 30/a )=1

Now,{$1'$= 30 and $30'$= 1};{$2'$= 15 and $15'$= 2}; { $3'$= 10 and $10'$= 3};{$5'$= 6 and $6'$= 5}.

Thus, each $a \in D_{30}$ has its complement $a'$ in B.

Space for learners:

Hence, ( $D_{30}$ , ∨ , ∧ , ‘ ,) is a Boolean algebra.

3, 4, 5: TRY YOURSELF

6. (a) A    6(b) 1    6(c) $A \vee B \vee C \vee D$

7. $(A \wedge B) \vee (A \wedge B') \vee (A' \wedge B)$

8. $(A \wedge B \wedge C) \vee (A \wedge B \wedge C') \vee (A \wedge B' \wedge C') \vee (A' \wedge B \wedge C)$

9. $(A \vee B' \vee C) \wedge (A \vee B' \vee C') \wedge (A \vee B \vee C) \wedge (A' \vee B \vee C) \wedge (A \vee B \vee C')$

10. $(X \vee Y \vee Z') \wedge (X \vee Y' \vee Z) \wedge (X' \vee Y \vee Z) \wedge (X' \vee Y \vee Z')$

## 4.18 POSSIBLE QUESTIONS

1. Find a Boolean product of the variables x, y and z or their complement that has the value 1 if and only if

(a) $x=y=0, z=1$  (b) $x=0, y=1, z=0$  (c) $x=0, y=z=1$  (d) $x=y=z=0$

2. Find the sum of product expansion of the function $f(x,y,z)=x$

3. With the values of truth table express the values of the following Boolean Function

(a) $F(x,y,z)=(x \wedge y') \vee (xyz)'$  (b) $F(x,y,z)=(x' \wedge y) \vee (y' \wedge z)$

4. With the help of the truth table of Conjunction and Disjunction Operation verify De Morgan’s Laws.

5. Using identities of Boolean algebra show that

$(x \wedge y') \vee (y \wedge z') \vee (x' \wedge z) = (x' \wedge y) \vee (y' \wedge z) \vee (x \wedge z')$

## 4.19 REFERENECS AND SUGGESTED READINGS


- Lattices and Boolean Algebra first concept, Second Edition By Vijay K Khanna
- Discrete Mathematical Structures with Application to Computer Science by J.P Tremblay & R. Manohar.


Space for learners:

# UNIT 5: ALGEBRAIC STRUCTURES

Space for learners:

**Unit Structure:**



## 5.0 INTRODUCTION

An algebraic structure consists of a non-empty set together with one or more binary compositions which satisfies some postulates. An Algebraic structure is the collection of any particular models of a given set of axioms.If $*$ is a binary operation on $G$. Then $(G, *)$ is an algebraic structure.$(R, +, .)$ is an algebraic structure equipped with two operations

Space for learners:

## 5.1 UNIT OBJECTIVES

After going through this unit, you will be able to:

- understand the basics of group and its various properties.
- know the cancellation laws of groups
- define subgroups and various operations on subgroups
- understand the Lagrange's Theorem
- understand ring and its operations
- give the definition of field
- understand the vector space

## 5.2 GROUP: THEOREM AND PROPERTIES

**Group :** A non empty set $G$ , together with a binary composition '$*$' (star) is said to form a group, if it satisfies the following postulates

(i) Closure property:

$a * b \in G$ , for all $a$ , $b \in G$

(ii) Associativity:

$(a * b) * c = a * (b * c)$ ,for all $a, b, c \in G$

(iii) Existence of identity:

There exists an unique element $e \in G$, called the identity element of $G$ such that

$a * e = a = e * a$ , for all $a \in G$

( the element $e$ is called the identity element )

(iv) Existence of inverse:

For every $\in G$ , there exist $a^{'} \in G$ ( depending upon $a$ ) such that $a * a^{'} = e = a^{'} * a$.

**Note:** (i) The group $G$ with the binary operation $*$ is sometimes denoted by <$G$ ,$*$>

(ii) In particular the group $< G , +>$ is called an addition group , the binary operation being addition.

(iii) In particular the group $< G , . >$ is called a multiplication group , the binary operation being multiplication.

**5.2 Basic Terms and their definition**

**Commutative group ( or Abelian group ):**

A commutative group is an order pair $(G , *)$ where G is a non-empty set and $*$ is a binary operation defined on $G$ such that the following properties hold.

(i) Closure property:

$a * b \in G$ , for all $a$ , $b \in G$

(ii) Associativity:

$a * (b * c) = (a * b) * c$ , for all $a , b , c \in G$

(iii) Existence of identity:

There exist an element $\in G$ , called the identity element of $G$ such that $a * e = a = e * a$ , $\forall a \in G$ .

(iv) Existence of inverse:

For any $\in G$ , $\exists$ an element $a^{'} \in G$ ( depending on $a$ ) such that $a * e^{'} = e = e^{'} * a$ , where $a^{'}$ is called inverse of $a$.

(v) Commutativity:

Space for learners:

The binary composition $*$ is commutative i.e. $* b = b * a$ , $\forall a , b \in G$ .

Semi-Group**:** A non-empty set $G$ together with binary composition $(.)$

is called a semi-group if

$$a . (b . c) = (a . b) . c \forall a , b , c \in G$$

Note: Every group is a semi-group .

**Monoid:** A non-empty set $G$ together with binary composition which

is associative and identity element exists is said to be a monoid.

**Finite group and Infinite group:**

If in a group , the set $G$ has a finite number of distinct elements is

known as a finite group and if the number of elements of the set are

infinite then it is known as an infinite group.

**Order of a group:**

The number of elements in a finite group is called the order of the

group and is denoted by $O(G)$ or $|G|$.

e.g: In the group $G = \{1 , -1 , i , -i\}$

The order of group is $O(G) = 4$.

**Order of an element in a group ( Period):**

The order of an element $a$ in a group $G$ is the least positive integer

$n (if exist)$ such that $a^n = e$ , the identity element of $G$ and we

write $O(a) = n$.

If $a^n \neq e \forall$ positive integer $n$ then $a$ is said to be of infinite order

or of zero order.

Ex.1 Show that the set of integer $I$ is a group with respect to the

***Space for learners:***

operation of addition.

Solution: (i) Closure property:

We know that sum of two integer is also an integer i.e. $a + b \in I$ ,

$$\forall a , b \in I$$

Thus $I$ is closed with respect to addition.

(ii) Associativity:

We know that addition of integers is an associative composition.

$$\therefore a + (b + c) = (a + b) + c \forall a , b , c \in I$$

(iii) Existence of identity :

The number $0 \in I$ , also we have $0 + a = a = a + 0 \;\; \forall a \in I$

$\therefore$the integer 0 is an identity.

(iv) Existence of inverse:

For any $a \in I$ , then $-a \in I$

$$\therefore a + (-a) = 0 = (-a) + a$$

$\therefore I$is a group with respect to addition

i.e. $< I , + >$is a group.

Space for learners:

**5.2 Group : Theorem and Properties**

1. Theorem: The identity element in a group is unique.

Proof: Let $G$ be a group. If possible, let $e$ and $e'$ be two identity element of a group $G$.

We have , $ee' = e$ if $e'$ is the identity

and $ee' = e'$ if $e$ is the identity.

But $ee'$ is a unique element of G.

$\therefore ee' = e$ and $ee' = e \quad => e = e'$

Hence the identity element is unique.

2. Theorem: The inverse of each element of a group is unique.

Proof: Let $a$ be any element of a group $G$ and let $e$ be the identity element. Suppose $b$ and $c$ are two inverse of $a$ i.e.

$ba = e = ab(i)$and $ca = e = ac \qquad (ii)$

We have , $(ba)c = ec(\therefore ba = e)$

$$= c(iii)(\therefore eisidentity)$$

Also , $b(ac) = b(e)(\therefore ac = e)$

$$= be = b(iv)$$

But in a group composition is associative

$\therefore b(ac) = (ba)c => b = c$

Hence , inverse of an element of a group is unique .

3. Theorem: The inverse of product of two elements of a group is the product of the inverse taken in

the reverse order.

OR

Prove that $(ab)^{-1} = b^{-1}a^{-1}\forall a\,, b \in G$ , where $G$is a group.

Proof: Let $a$ and $b$ be elements of $G$.

If $a^{-1}$ and $b^{-1}$ are inverse of $a$ and $b$ respectively.

Then $aa^{-1} = e = a^{-1}a$

and $bb^{-1} = e = b^{-1}b$ , where $e$ is the identity element.

Now , $(ab)(b^{-1}a^{-1}) = [(ab)b^{-1}]a^{-1}\left(\begin{matrix}\because compositionisan \\ associative\end{matrix}\right)$

$$= [a(bb^{-1})]a^{-1}(byassociativity)$$

$$= [ae]a^{-1}(\because bb^{-1} = e)$$

$$= aa^{-1}(\because ae = a)$$

Space for learners:

$$= e$$

Also , $(b^{-1}a^{-1})(ab) = b^{-1}[a^{-1}(ab)]$ , by associativity

$$= b^{-1}[(a^{-1}a)b]$$

$$= b^{-1}(eb)(\because a^{-1}a = e)$$

$$= b^{-1}b$$

$= e \quad (\because b^{-1}b = e)$

Thus , we have $(ab)(b^{-1}a^{-1}) = e = (b^{-1}a^{-1})(ab)$

$\therefore$by definition of inverse , we have

$$(ab)^{-1} = b^{-1}a^{-1}$$

Ex.2 Show that the set $N$ of all natural numbers $N$ is not a group with respect to addition.

Solution: (i) Closure: Property:

We know that sum of two natural number is natural number.

$\therefore$ $N$ is closed with respect to addition.

(ii) Associativity:

Also , addition of natural number is an associative composition.

(iii) Existence of identity :

But there exist no natural number $e \in N$ such that $e + a = a = a + e$

$$\forall a \in N$$

For addition , 0 is the identity but $0 \notin N$

Hence , condition of existence of identity is not satisfied.

$\therefore$ $N$ is not a group w.r.t. addition.

$\therefore < N , + >$ is not a group .

Space for learners:

## 5.3 Cancellation laws

If $a$ , $b$ , $c$ are any elements of $G$ then (i) $ab = ac \Rightarrow b = c$

( Left cancellation law )

(ii) $ba = ca \Rightarrow b = c$ (Right cancellation law)

Proof: Let $e$ be the identity

Since , $a \in G$ =>there exist $a^{-1} \in G$ such that $a^{-1}a = e = aa^{-1}$

Now , $ab = ac$

$\Rightarrow a^{-1}(ab) = a^{-1}(ac)$ *( multiplying both sides on left by* $a^{-1}$ *)*

$\Rightarrow (a^{-1}a)b = (a^{-1}a)c$ *, by associativity*

$$\Rightarrow eb = ec (\because \mathrm{a}^{-1}\mathrm{a} = \mathrm{e})$$

$\Rightarrow b = c$ $(\because e \text{ is the identity})$

$$\therefore ab = ac \Rightarrow b = c$$

Ex.1 Show that the cancellation laws do not hold in a semi-group.

Solution: Consider the set $M$ of all $2 \times 2$ matrices over integers under matrix multiplication , which forms a semi-group.

If we consider $A = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$ , $B = \begin{bmatrix} 0 & 0 \\ 0 & 2 \end{bmatrix}$ and $C = \begin{bmatrix} 0 & 0 \\ 3 & 0 \end{bmatrix} \in M$

Then , $AB = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} 0 & 0 \\ 0 & 2 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$

$\&AC = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} 0 & 0 \\ 3 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$

$\therefore AB = AC$

But , $B \neq C$

$\therefore$ the cancellation laws do not hold in a semi-group

Ex.2 Prove that if for every element $a$ in a group $G$ , $a^2 = e$ where $e$ is the identity element of $G$ then show that $G$ is abelian.

Space for learners:

Solution: Let $G$ be a group such that $a^2 = e \forall a \in G$ where $e$ is the identity element of $G$.

We are to show that $G$ is abelian.

Let $a, b \in G$ then $ab \in G$ and so

$$(ab)^2 = e$$

$$=> (ab)(ab) = e$$

$=> (ab)(ab)b^{-1} = eb^{-1} \quad [\because b^{-1} \in G]$

$=> (ab)a(bb^{-1})a^{-1} = b^{-1}a^{-1} \quad [\because a^{-1} \in G]$

$$=> (ab)(aea^{-1}) = b^{-1}a^{-1}$$

$$=> abe = b^{-1}a^{-1}$$

$=> ab = b^{-1}a^{-1}$ (i)

Also , $a \in G => a^2 = e$ , by hypothesis

$$=> ae = e$$

$$=> (aa)a^{-1} = ea^{-1}[\because a^{-1} \in G]$$

$$=> a(aa^{-1}) = a^{-1}$$

$\therefore a = a^{-1}$ (ii)

Similarly , $b \in G => b^2 = e$

$=> b = b^{-1}$ (iii)

Using (ii) and (iii) we get from (i)

$$ab = ba$$

Thus , $a , b \in G => ab = ba$

So $G$is abelian.

### 5.3.1Permutation Group:

Let $A$ be a finite set ( may be finite or infinite ) and $f: A \to A$ is bijective mapping then $S$ is called permutation group i.e.

***Space for learners:***

$$P(A) = \{f: A \to A: f \; is \; bijective \; mapping\}$$

Space for learners:

The number of elements in the finite set $S$ is known as the degree of

permutation.

**Cycle or circular permutation:**

Let $\propto \in S_n$ , then $\alpha$ is called a cycle or circular permutation

if there exists

$\{i_1, i_2, \; \ldots \ldots, \; i_r\}$ such that $\alpha(i_1) = i_2$ ,

$\alpha(i_2) = i_3, \ldots \ldots \ldots \ldots \; , \alpha(i_{r-1}) = i_r, \alpha(i_r) = i_1$ then

permutation is represented by $\alpha = (i_1 i_2 \; \ldots \ldots \ldots \; i_r)$and $\alpha(i) = i$
for

all other $i \in S_n$

## 5.4 Subgroup

Sub group**:** Let $< G, *>$ be a group and $H$ be a non-empty

subset of $G$. Then $H$ is called a subgroup $G$. Then $H$ is called

a subgroup $G$ if and only if $H$ itself is a group under the binary

composition '$*$' defined on $G$.

The above definition can be written in full as follows:

Let $< G \;, *>$ be a group and $\subseteq G$ , $H \neq \phi$. Then $H$ is called a

subgroup of $G$ if

(i) $H$ is closed w.r.t. the binary composition '$*$' i.e. $a * b \in H$

$\forall a \;, b \in H$.

(ii) '$*$' is associative in $H$ ( which is obvious since $G$ is group).

(iii) $e \in H$ , where $e$ is the identity element of $G$ .

(iv) $\forall a \in H$ there exist $a^{-1} \in H$ such that $a * a^{-1} = e = a^{-1} * a$ .

Ex.1 Show that a non-empty subset $H$ of a group $G$ is a subgroup

of $G$ if and only $, b \in H => ab^{-1} \in H$ .

Solution: Let $G$ be a group and H be a non empty subset of $G$.

First we assume , $H$ be a subgroup of $G$ then $H$ itself a group w.r.t. the

binary composition defined on $G$. We show that $a , b \in H => ab^{-1} \in H$

Let $, b \in H$ , since $H$ is a group .

$$b \in H => b^{-1} \in H$$

Thus , $a , b \in H => a , b^{-1} \in H$

$=> ab^{-1} \in H$ [ since $H$ is a group , so using closure property in H]

$$\therefore a \ , b \in H => ab^{-1} \in H$$

Conversely let $H$ be such that $a , b \in H => ab^{-1} \in H$ ............. (i)

We show that $H$ is a subgroup of $G$.

(I) Existence of identity:

Let $e$ be the identity element of $G$.

Since , $H$ is nonempty so there exists $a \in H$

Taking $b = a$ in $(i)$ , we see that $aa^{-1} \in H => e \in H$

$\therefore$ This shows the existence of identity in $H$

(II) Existence of inverse:

Let $a \in H$ , by (I) , $e \in H$

$\therefore e \in H , a \in H => ea^{-1} \in H$(using(i))

$$=> a^{-1} \in H$$

Thus $a \in H => a^{-1} \in H$

**Space for learners:**

This shows the existence of the inverse of every element in $H$.

Space for learners:

(III) Closure property:

Let $a , b \in H$. Then by (II) , $b^{-1} \in H$

Thus , a , b$\in$ $H => a(b^{-1})^{-1} \in H$ [by using (i)]

$=> ab \in H$

Thus , $a , b \in H => ab \in H$

Which shows that $H$ is closed with respect to the composition $G$.

(IV) Associativity:

Let $a , b , c \in H$

Since $\subseteq G$ , $a , b , c \in G$

Since , $G$ is a group , we have

$(ab)c = a(bc)$

Hence , associativity holds in $H$.

From (I) to (IV) , it follows that $H$ itself a group with respect to composition $G$.

So , $H$ is a subgroup of $G$ .

Intersections of subgroups:

Theorem1. If $H$ and $K$ are two subgroups of a group $G$ then $H \cap K$ is a subgroup.

Proof: Let $H$ and $K$be two subgroups of a group $G$

We are to show that $H \cap K$ is a subgroup of G.

Since $H$ and $K$ are subgroups of $G$ .

We have , $H \subseteq G$ , $K \subseteq G$ , $e \in H$ , $e \in K$ , where $e$ is the identity element of $G$.

$=> H \cap K \subseteq G$ and $e \in H \cap K$

$=> H \cap K \subseteq G$ ..........(i) $\&H \cap K \neq \phi$ ........(ii)

Finally , let $x , y \in H \cap K$ then

$x , y \in H and x , y \in K$

Since , $H$ is a subgroup of $G$,

$x , y \in H => xy^{-1} \in H$

Similarly , $x , y \in K => xy^{-1} \in K$

$$\therefore xy^{-1} \in H \& xy^{-1} \in K$$

$$=> xy^{-1} \in H \cap K$$

Thus , $x , y \in H \cap K => xy^{-1} \in H \cap K$ ................(iii)

From (i) , (ii) and (iii) we see that

$\therefore H \cap K$ is a subgroup of $G$.

**5.4.1.** Give an example to show that union of two subgroup of a group may not be a subgroup of the group.

Solution: we consider the additive group $< I , + >$ of all integers. Let $H = \{2a / a \in I\}$ and $K = \{3a \;/\; a \in J\}$ then

$H \subseteq I$ , $K \subseteq I$ and $H \neq \phi$ , $K \neq \phi$ .....................(i)

Also , $x , y \in H => x = 2a$ , $y = 2b$ , where $a , b \in I$

$=> x - y = 2(a - b) = 2c$ , where $c = a - b \in I$

$=> x - y \in H$ ............. (ii)

Similarly , $x , y \in K => x - y \in K$ ................(iii)

From (i) , (ii) & (iii) we see that

$\therefore H$ and $K$are subgroups of $I$.

Now , $2 \in H \subseteq H \cup K \& 3 \in K \subseteq H \cup K$

$$=> 2 , 3 \in H \cup K$$

Space for learners:

But $2 + 3 = 5 \notin H \& 2 + 3 = 5 \notin K$

$$=> 2 + 3 = 5 \notin H \cup K$$

Hence , $H \cup K$ is not closed with respect to addition and

consequently $H \cup K$ is not a subgroup of $I$ .

**Definition:**

Coset: Let $H$ be a subgroup of a group $G$. If $a \in G$ then the set

$Ha = \{ha: h \in H\}$ is called the right coset of $H$ in $G$ generated by $a$.

Let $H$ be a subgroup of a group. If $a \in G$ then the set $aH = \{ah: h \in H\}$ is called the left coset of $H$ in $G$ generated by $a$.

Note: Any two right (left) cosets of a subgroup are either disjoint or

Identical.

**Lagrange's Theorem:**

Statement: The order of each subgroup of a finite group

is a divisor of the order of the group.

Proof: Let $G$ be a finite group and $o(G) = n$. Let $H$ be a

subgroup of $G$ then $H$ is obviously finite. Let $o(H) = m$.

We first show that $o(aH) = m \forall a \in G$

We define $f: H \to H$ by $f(h) = ah \forall h \in H$

Then , $h_1, h_2 \in H$ , $f(h_1) = f(h_2) => ah_1 = ah_2$

$=> h_1 = h_2$ , by left cancellation law.

$\therefore f$ is one-one .

Also , for an arbitrary element $ah \in aH$ , we find $h \in H$ such

that $f(h) = ah$ and so $f$ is onto.

Space for learners:

Thus , $f: H \rightarrow aH$ is bijection and consequently $o(aH) =$

$o(H) = m \quad , \forall a \in G \ldots \ldots \ldots \ldots (i)$

Next , let $C = \{aH: a \in G\}$

Since , $G$ is finite, $C$ is clearly a finite family .

Also , two distinct element of $G$ may produce the same left coset . So if $(C) = k$ , then $1 \leq k \leq m$ .

Let $C = \{a_1H , \ a_2H , \ldots \ldots \ldots \ldots \ldots \ldots .. , a_kH\}$ , where

$$a_1 , a_2 , \ldots \ldots \ldots \ldots , a_k \in G$$

Clearly , $a_iH \subseteq G$ for $i = 1 , 2 , \ldots \ldots \ldots , k$

$$=> \cup_{i=1}^{k} a_iH \subseteq G \ldots \ldots \ldots \ldots (ii)$$

Further , $x \in G => xH = a_jH$ , for some $j$ , $1 \leq j \leq k$

$$=> x \in xH = a_jH \subseteq \cup_{i=1}^{k} a_iH$$

$$=> x \in \cup_{i=1}^{k} a_iH$$

$\therefore G \subseteq \cup_{i=1}^{k} a_iH \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots (iii)$

From (ii) & (ii) we get ,

$G = \cup_{i=1}^{k} a_iH \ldots \ldots \ldots \ldots (iv)$

Also , 1$\leq i \leq k$ , $1 \leq j \leq k, i \neq j => a_iH \neq a_jH$

$=> a_iH \cap a_j = \phi$ ..........(v)

Hence , the coset $a_1H , a_2H , \ldots \ldots \ldots \ldots , a_nH$ are mutually disjoint .

From (iv) we get ,

$n = o(G) = o\left(\cup_{i=1}^{k} a_iH\right)$

$= o(a_1H) + o(a_2H) + \ \ldots \ldots \ldots .. + o(a_kH)$

$= m + m + \ \ldots \ldots \ldots \ldots \ldots \ldots$..upto $k$ times (by (i) )

$$= km$$

Space for learners:

$=> \frac{n}{m} = k$

$=> \frac{o(G)}{o(H)} = k$ , where $k$ is positive integers.

$=> o(H)$is a divisor of $o(G)$ .

Cyclic group: Let a group $G$ is said to be cyclic if there exist an element $a \in G$ such that every element $x \in G$ is of the form $= a^n$ , where $n$ is an integer. The element $a$ is then called the generator of $G$ and we can write $G =< a >$.

e.g:- $G = \{1 , -1 , \ i , -i\}$

$= \{i^4 , \quad i^2 , \quad i^1 , i^3\}$

$\therefore G =< i >$ is a cyclic group under multiplication .

Ex 5.4.2. Show that every cyclic group is abelian.

Solution: Let $G =< G >$ be a cyclic group generated by $a \in G$.

Let $x , y \in G$ be any elements .

Then $x = a^r$ , $y = a^s$ for some integer $r\&s$

$\therefore xy = a^r a^s$

$$= a^{r+s}$$

$$= a^{s+r} = a^s a^r = yx$$

$$\therefore xy = yx$$

$\therefore G$ is abelian .

Ex.5.4.3 Show that a subgroup of a cyclic group is cyclic.

Solution: Let $G =< a >$ be a cyclic group generated by $a \in G$ and $H$ is any subgroup of $G$.

If $H = \{e\} =< e >$ then clearly $H$ is a cyclic group

Let $H \neq \{e\}$

Space for learners:

and $x \in H$ be any non-identity element .

$=> x \in G$ , since $H$ is a subgroup of $G$

$=> x = a^n$ for some integer

$$=> x^{-1} = a^{-n}$$

$$\therefore a^n , a^{-n} \in H$$

Let $m$ be the least positive integer such that $a^m \in H$

We shall show that $H =< a^m >$ is a cyclic group generated by $a^m$

Let $x \in H$

$=> x \in G$ , since $H$ is a subgroup of $G$

$=> x = a^n$ , for some integer

By division algorithm there exists integer $q$ and $r$ such that

$n = mq + r$ , where $0 \leq r < |m|$ ............ (i)

$$=> r = n - mq$$

$$=> a^r = a^{n-mq}$$

$$=> a^r = a^n a^{-mq}$$

$$=> a^r = a^n (a^m)^{-q}$$

$$\therefore a^r \in H$$

$\therefore r = 0$ , since $m$ is the least positive integer such that $a^m \in H$

From (i)=>

$$n = mq$$

$=> a^n = a^{mq}$

$=> a^n = (a^m)^q$

$=> x = (a^m)^q$

$\therefore H =< a^m >$ is a cyclic subgroup.

**CHECK YOUR PROGRESS**

1. State whether true or false

(a) A non-empty subset $H$ of a group $G$, which is closed under the binary composition in $G$ is a subgroup of $G$.

(b) If $G$ is a group and $H$ is a non-empty subset of $G$, then $H$ will be

***Space for learners:***

Space for learners:

### 5.5 Ring

Ring: A ring is an order triples $< R , + , \ . >$ where $R$ is a nonempty

set and $+ , \ .$ are two binary operation on $R$ satisfying the following

axioms .

$[R_1]$ Closure property for addition:

$$a , b \in R => a + b \in R \forall a , b \in R$$

$[R_2]$ Associativity for addition:

$(a + b) + c = a + (b + c) \forall a \ , b \ , c \in R$

$[R_3]$ Existence of identity w.r.t. addition:

There exist an element $0 \in R$ , called the zero element of $R$ such that

$a + 0 = a = 0 + a \forall a \in R$

$[R_4]$ Existence of inverse w.r.t. addition:

For all $\in R$ , there exist an element $-a \in R$ such that

$$a + (-a) = 0 = (-a) + a$$

$[R_5]$ Commutative property for addition:

$$a + b = b + a \forall a , b \in R$$

$[R_6]$ Closure property for ( . ):

$$a.b \in R \quad , \qquad \forall a , b \in R$$

$[R_7]$Associative property for ( . ):

$$(a.b).c = a.(b.c) , \forall a , b , c \in R$$

$[R_8]$Distributive laws of (.) and (+):

$$(a + b).c = a.c + b.c$$

$$c.(a + b) = c.a + c.b \forall a , b , c \in R$$

Ex.1 Give two examples of ring.

Solution: (i) We consider the set $R$ of real numbers equipped with two binary composition addition (+) and multiplication (.) then it is easy to verify that $< R , + , \ . >$ is a ring.

(ii) Let $M_2$ denotes the set of all $2 \times 2$ matrices of real numbers . In $M_2$ we consider two binary operations , viz. addition (+) of matrices and multiplication (.) of matrices then it is easy to verify that $(M_2 , + , \ .)$ is a ring.

Ex.2.2 Prove that the set of matrices $M_2$ of order $2 \times 2$ form a ring with respect to addition and multiplication.

Solution: Let $A$ and $B \in M_2$. Then $A$ and $B$ are two $2 \times 2$ matrices and so $A + B$& AB are also $2 \times 2$ matrices.

$$\therefore A + B \in M_2 \& AB \in M_2$$

This shows that $M_2$ is closed with respect to addition and multiplication of matrices ................ (i)

Since , both addition and multiplication of matrices are associative we see that both the composition in $M_2$ are associative. ......(ii)

Also there exists $2 \times 2$ null matrix

**_Space for learners:_**

$O = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$ such that for any $2 \times 2$ matrix $A \in M_2$ ,

$$A + 0 = A = 0 + A$$

This shows that 0 is the zero element of $M_2$ ...............(iii)

Further if $A \in M_2$ where $A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$

Then there exists a matrix

$-A = \begin{bmatrix} -a_{11} & -a_{12} \\ -a_{21} & -a_{22} \end{bmatrix} \in M_2$ such that $A + (-A) = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$

$$+ \begin{bmatrix} -a_{11} & -a_{12} \\ -a_{21} & -a_{22} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} = 0$$

Similarly , $(-A) + A = 0$

Therefore , $A + (-A) = 0 = (-A) + A$

This shows that $-A$ is the inverse of $A$ in $M_2$

Thus , every element of $M_2$ has inverse .........................(iv)

Further addition of matrices is commutative and so the composition

in $M_2$ is commutative ........(v)

Finally , by the distributive property of multiplication of matrices over

addition we have,

$$\left.\begin{aligned} A(B + C) &= AB + AC \\ (B + C)A &= BA + CA \end{aligned}\right\}$$

$\forall A , B , C \in M_2$ ................. (vi)

From (i) to (vi) we see that $M_2$ is a ring with respect to addition

and multiplication of matrices .

Note : The ring $R$ with two binary composition $(+)$and $(\,.\,)$

is sometimes denoted by $(R\ , +\ ,\ .)$.

Space for learners:

**Commutative Ring:**

A ring $(R, +, .)$ is called a commutative ring if and only if for all $a, b \in R$, $a.b = b.a$.

**Ring with unity:**

If in a ring $R$ there exists an element $1 \in R$ such that

$$1.a = a = a.1 \quad \forall a \in R$$

then $R$ is called a ring with unity element.

The element 1 is called the unity element of the ring.

2. Theorem: In a ring, the following results hold

(i) $a.0 = 0 = 0.a \forall a \in R$

(ii) $a.(-b) = (-a).b = -a.b \forall a, b \in R$

(iii) $(-a).(-b) = a.b$

(iv) $a.(b - c) = a.b - a.c$

(v) $(b - c).a = b.a - c.a$

Proof: (i) we have

$a.0 = a(0 + 0) [\therefore \quad 0 = 0 + 0]$

$=> a.0 = a.0 + a.0 \quad [by left distributive law]$

$=> 0 + a.0 = a.0 + a.0$ [ since $0.a \in R$and $0 + a.0 = a.0$]

$=> 0 = a.0$ [ sincr $R$ is a group w.r.t. $+$ ,therefore applying right cancellation law for addition in $R$]

Similarly , we have ,

$0.a = (0 + 0).a = 0.a + 0.a$ [ by right distributive law]

$$=> 0 + 0.a = 0.a + 0.a [\therefore 0 + 0.a = 0.a]$$

$=> 0 = 0.a$ [using R.C.L. w.r.t. $+$ in the group $< R, + >$]

$$\therefore a.0 = 0.a = 0$$

Space for learners:

(ii) From the existence of inverse it follows that

$$b + (-b) = 0$$

$$=> a.[b + (-b)] = a.0$$

$=> a.b + a.(-b) = 0$ [ by the left distributive law and $a.0 = 0$]

$=> -a.b + [a.b + a.(-b)] = -a.b + 0$ [ adding $-a.b$ on the left

of both sides]

$=> [-a.b + a.b] + a.(-b) = -a.b$ [ by associative law ]

$$=> 0 + a.(-b) = -a.b$$

$$=> a.(-b) = -a.b$$

Similarly , we can prove

$$(-a).b = -ab$$

$$\therefore a.(-b) = (-a).b = -a.b$$

(iii) we know that

$$a.(-b) = -.(ab) \ldots\ldots\ldots..(1)$$

Writing $-a$ for $a$ in (1) , we get

$(-a).(-b) = -[(-a).b]$

$= -[-(a.b)]$ , since $(-a).b = -a.b$

$= a.b$ [ $\therefore R$ is a group and inverse of the inverse

of an element is the element itself. $-(-a = a)$ ]

$\therefore (-a).(-b) = a.b$

(iv) we have , $a.(b - c) = a.[b + (-c)]$

$= a.b + a.(-c)$[ by left distributive law]

$= a.b + [-a.c]$[ $\therefore a.(-c) = (-a).c = -a.c$]

$$= a.b - a.c$$

**Space for learners:**

(v) we have , $(b-c).a = [b-c].a$

$= b.a + (-c).a$( by right distributive law)

$$= b.a + (-c.a) \because [(-c).a = -c.a]$$

$$= b.a - c.a$$

Theorem 2.2 A commutative ring is an integral domain if and only if

the cancellation law with respect to multiplication hold on it.

Proof: Let $R$ be a commutative ring . First we assume that $R$ is

an integral domain We are to show that the cancellation laws

hold in $R$.

Let $ab = ac$ , where $a, b, c \in R$ , $a \neq 0$

$$=> ab - ac = 0$$

$$=> a(b-c) = 0$$

$\therefore$either $a = 0$ or $b - c = 0$ [$\because R$is an integral domain ]

But $a \neq 0$ , $b - c = 0$

$$=> b = c$$

$\therefore ab = ac => b = c$ , which shows that left cancellationlaw

with respect to multiplication .

Similarly , $a \neq 0$ , $b$ , $c \in R$ , $ba = ca => b = c$ , which shows

the right cancellation law with respect to multiplication

$\therefore$the cancellation law holds in $R$ .

Conversely , suppose that the cancellation law with respect to

multiplication hold in $R$. We are to show that $R$ is an integral domain.

Let $a, b \in R$ such that $ab = 0$

Space for learners:

Space for learners:

Now , $a \neq 0$ , $ab = 0$

$$=> ab = a0$$

$=> b = 0$ [ by left cancellation law w.r.t. multiplication hold in $R$ ]

Similarly , $b \neq$ , $ab = 0$

$=> ab = 0b$ [ by right cancellation law w.r.t. multiplication]

$$=> a = 0$$

Hence , $a , b \in R$ , $ab = 0 =>$ either $a = 0$ or $b = 0$

So , $R$is an integral domain .

Hence , commutative ring is an integral domain if and only if the cancellation law with respect to multiplication hold on it.

Ex.3 Give an example of ring.

Solution: We consider the set of real number equipped with two binary composition addition (+) and multiplication ( . ) then it is easy to verify that $(R , + , \ .)$ is a ring .

Ex4. If $R$ is a ring such that $a^2 = a \forall a \in R$ , prove that

(i) $a + a = 0 \ \forall a \in R$

(ii) $a + b = 0 => a = b$

(iii) $R$ is commutative.

Solution: (i) Let $a \in R$ such that $a^2 = a$ then by closure property ,

$$a + a \in R$$

$=> (a + a)^2 = a + a$ [ by hypothesis ]

$=> (a + a)(a + a) = a + a$

$=> (a + a)a + (a + a)a = a + a$

$=> (a^2 + a^2) + (a^2 + a^2) = a + a$ [ by right distributive law]

$$=> (a + a) + (a + a) = a + a (\because a^2 = a)$$

$=> (a + a) + (a + a) = (a + a) + 0$

$=> a + a = 0$ [by right distributive law]

(ii) Let $a , b \in R$ such that $a + b = 0$ then using (i)

$a + b = 0 = a + a$

$=> b = a$ [ by left cancellation law]

$\therefore a = b$

$\therefore a + b = 0 => a = b$

(iii) Let $a , b , c \in R$ then $a + b \in R$

and so by hypothesis , $(a + b)^2 = a + b$

$=> (a + b)(a + b) = a + b$

$$=> (a + b)a + (a + b)b = a + b$$

$=> a^2 + ba + ab + b^2 = a + b$ [by right distributive law]

$$=> a + ba + ab + b = a + b$$

$=> ba + ab + b = b$ [by LCL]

$=> ba + ab = 0$ [ by RCL]

$=> ba = ab$[by result (ii)]

$$\therefore ab = ba \forall a , b \in R$$

$\therefore R$ is commutative .

**Zero divisor in a ring:**

A non zero element ‘ $a$’ in a ring $R$ is called a ( proper ) zero divisor

if there exist another non zero elements ‘$b$’ in $R$such that $ab = 0$ .

Ex.5 Give an example of a ring with zero divisor.

Solution: We consider a ring $M_2$ of all $2 \times 2$ matrix over real numbers

Let $A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$, $B = \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix} \in M_2$

Space for learners:

Then $A \neq 0 \quad , B \neq 0$

$$AB = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} = 0$$

$\therefore M_2$is a proper zero divisor

**Integral domain:**

A commutative ring $R$ without proper zero divisor is called an integral

domain i.e. $ab = 0 \;\forall a , b \in R$

$=> a = 0$ or $b = 0$

Ex.6 Give an example of a ring which is not an integral domain.

Solution: Let $M_2$ denote the set of all $2 \times 2$ matrices of realnumbers

then it is easy to verify that $M_2$ is a ring under matrix addition and

multiplication .

Now , let $A = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$ and $B = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$

Then $A \neq 0 \;, \; B \neq 0$

But $AB = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} = 0$

$\therefore M_2$ is not an integral domain

Ex.7 When is a ring said to be an integral domain ?

Solution: A ring $< R \;, + \;, \;. >$is said to be an integral domain

if thefollowing two properties are satisfied in .

(i) Commutative property of multiplicativity

$$ab = ba \forall a , b \in R$$

(ii) Non existence of zero divisor:

$a , b \in R \quad , \; ab = 0 =>$either $a = 0$ or $b = 0$

**5.6 Field**

Space for learners:

Definition: A field is an order triple $< F\ ,+\ ,\ . >$ where $F$ is a set containing atleast two elements and $(+)\ ,(.)$ are two binary composition in $F$ satisfying the following axioms :

(i) Closure property for addition :

$$a + b \in F \forall a\ , b \in F$$

(ii) Associativity for addition:

$$a + (b + c) = (a + b) + c \forall a\ ,\quad b\ , c \in F$$

(iii) Existence of identity:

There exist an element $0 \in F$ called the zero element of $F$ such that

$$a + 0 = a = 0 + a \forall a \in F$$

(iv) Existence of additive inverse:

For all $a \in F$ there exist an element $-a \in F$ such that

$$a + (-a) = (-a) + a$$

(v) Commutativity for addition :

$$a + b = b + a \forall a\ , b \in F$$

(vi) Closure property for multiplication:

$$a.b \in F \forall a\ , b \in F$$

(vii) Associativity for multiplication :

$$a.(b.c) = (a.b).c \forall a\ , b\ , c \in F$$

(viii) Distribution property of multiplication over addition:

$$a.(b + c) = a.b + a.c$$

$$(b + c).a = b.a + b.c \forall a\ , b\ , c \in F$$

(iX) Existence of multiplicative identity:

***Space for learners:***

There exist an element $1 \in F$ , called the unit element of $F$ such that

$$a.1 = a = 1.a \forall a \in F$$

(x) Existence of multiplicative inverse for non zero elements of $F$:

For every non zero element $a \in F$ there exist an element $a^{-1} \in F$ such that $a.a^{-1} = 1 = a^{-1}.a$

(xi) Commutativity for multiplication:

$$a.b = b.a \forall a , b \in F$$

**Alternate definition of Field:**

A commutative ring with unity (having atleast two elements) in which every non zero element has its multiplicative inverse ( i.e. the set of non zero element form a group under multiplication ) is called a field.

Ex. 1 Show that every field is an integral domain but converse is not true ( i.e. integral domain may not be a field).

Solution: Let $F$ be a field i.e. $F$ is a commutative ring with unity '1' in which every non-zero element has its multiplicative inverse .

To show that $F$ is an integral domain.

Let $a , b \in F$ such that $ab = 0$

We first assume that $a \neq 0$ since $F$ is a field , $a^{-1}$ exists i.e $a^{-1} \in F$ such that $aa^{-1} = 1 = a^{-1}a$

So , $ab = 0$

$$=> a^{-1}(ab) = a^{-1}(0) = 0$$

$$=> (a^{-1}a)b = 0$$

Space for learners:

$=> 1\, b = 0$ [ 1 being the unity element in $F$ ]

$$=> b = 0$$

Next , we assume that $b \neq 0$ then $b^{-1}$ exists and so

$$ab = 0$$

$=> (ab)b^{-1} = (0)b^{-1}$

$=> a(bb^{-1}) = (0)b^{-1}$

$=> a(1) = 0$ [ ∴1 being the unit element of $F$]

$=> a = 0$

Thus , $a , b \in F$ , $ab = 0 =>$ either $a = 0$ or $b = 0$

This shows that $F$ is an integral domain

Hence every field is an integral domain.

Ex. 2 Show that every integral domain may not be a field.

Solution: We consider the ring $< I , + , \; . >$ of integers with usual

addition and multiplication is an integral domain which is not a

field. Since , the non-zero elements except $\pm 1$ have no multiplicative

inverse in $I$.

Now , $2 \in I$ and $2 \neq 0$ but there exist no $\frac{1}{2} \in I$ such that $2 . \frac{1}{2} = 1 = \frac{1}{2} . 2$

∴the non-zero element $2 \in I$ has no multiplicative inverse .

So , $< I , + \; , \; . >$is not a field .

**5.7 Homomorphism of a group**

**5.7.1 Homorphism of Group**

A mapping $f : G \rightarrow G'$is said to be a homomorphism of $G$ into $G'$ if

$$f(ab) = f(a)f(b) \;\forall\; a \;, b \;\in G$$

***Space for learners:***

### 5.7.2 Kernel of a Homomorphism

If $f$ is a homomorphism of a group $G$ into a group $G'$, then the set $K$ of all those elements of $G$which are mapped by $f$

## 5.8 Vector Space

Let $(F\ ,+\ ,\ \ .)$ be field. The element of $F$ is called scalars. The element of $V$(any non-empty set) is called vectors. Then $V$ is a epace over the field $F$ if

(I) $(V\ ,+)$is abelian group

(II) External Composition in $V$ over $F$ i.e $\alpha a \in V\ \ ,a \in V\ \ ,\alpha \in F$

(III) The two composition $(+\ ,\ .)$ satisfy the following conditions

(a) $\alpha(a+b) = \alpha a + \alpha b \quad \forall\ a\ ,b \in V$

(b) $(\alpha+\beta)a = \alpha a + \beta a \quad \forall\ \alpha \in F$

(IV) $(\alpha\beta)a = \alpha(\beta a) \quad \forall\, a \in V\ ,\ \alpha\ ,\beta \in V$

(V)$1a = a$ ,1 is unity element of $F$

The $V$ is a vector space over the field .

e.g. $\mathbb{C}$ is a field of complex number and $\mathbb{R}$ is a field of real number.

(i) $\mathbb{C}(\mathbb{R})$ is vector space, since $\mathbb{R} \subseteq \mathbb{C}$ .

(ii) $\mathbb{R}(\mathbb{C})$ is not vector space.

### 5.8.1Linear Dependence and Linear Independence of vectors

Let $V(F)$ be a vector space. A finite set $\{\alpha_1,\ \alpha_2, \dots \dots \dots \dots .., \alpha_n\}$ of vectors of $V$ is said to be linearly dependent if there exist scalars $c_1, c_2\ , \dots \dots \dots ..\ \ , c_n \in F$not all of them zero ( some of them may be zero) such that $c_1\alpha_1 + c_2\alpha_2 + \ \dots \dots \dots \dots \dots . + c_n\alpha_n = 0 \quad i.e$ $\sum_{i=0}^{n} c_i\alpha_i = 0$ atleast one $\alpha_i \neq 0$.

Space for learners:

**Some Important Result:**

1. If two vectors are linearly dependent then one of them is a scalar multiple of other the other

2. Asystem consisting of a single non-zero vector is always linearly dependent.

**5.8.2 Vector Subspaces:**

Let $V$ be a vector space over the field $F$ and let $W \subseteq V$. Then $W$is called a subspace of $V$ if $W$ itself is a vector space over $F$ with respect to the operations of vector addition and scalar multiplication in $V$.

**Result:**

**1.** A subset $W$of a vector space $V(F)$ is a subspace of V , if and only if $\forall\ \alpha , \beta \ \in W$ and $a , b \in F => a\alpha + b\beta \in W$ .

**5.9 Basis and Dimension of a vector space**

Let $V$ be a vector space over a field $F$. Then a subset $B$ of $V$ is called a basis of $V$ if $B$ is linearly independent over $F$ and $V =< B >$

Number of elements in basis is called dimensions of vector space.

**Linear Span**

If $S = \{\alpha_1 , \alpha_2 , \alpha_3 , \dots \dots \dots \dots , \ \alpha_n\}$ from $(F)$ . Set of all liner combination of vectors of $S$ is known as linear span of $S$.

**Finite dimensional vector space**

A vector space $V_F$ is said to be finite dimensional vector space if there exist a finite subset $S$ of $F$ such that $=< B >$ .

If $\mathbb{R}^2(\mathbb{R})$ is finite dimensional vector space. It is generated

Space for learners:

by a finite set $B = \{(0\,,1)\,,(1\,,0)\}$ then $dim\big(\mathbb{R}^2(\mathbb{R})\big) = 2$

Space for learners:

**Example 5.9.1**

Consider the vector space $\mathbb{C}(\mathbb{R})$ , find the basis and dimension.

Solution: Let $\mathbb{C} = \{\alpha + i\beta : \alpha\,,\ \beta \in \mathbb{R}\}$

$$2 + 3i = \alpha.1 + \beta.i(Linear\ combination\ of\ basis)$$

$$= 2.1 + 3.i$$

$$\therefore dim\big(\mathbb{C}(\mathbb{R})\big) = 2$$

**Example 5.9.2**

What is the dimension of $\mathbb{R}(\mathbb{Q})$?

Solution: $Dim\mathbb{R}(\mathbb{Q}) = \infty$

## 5.11 SUMMING UP

- A non empty set , together with a binary composition '$*$' (star) is said to form a group, if it satisfies closure property, Associativity property, existence of identity and existence of inverse properties.
- A commutative group is an ordered pair $(G\,,\,*)$ having the additional Commutativity property in addition to all four properties of a group.
- If G be a group and H be a non-empty subset of $G$. Then $H$ is called a subgroup $G$ if and only if $H$ itself is a group under the binary composition '$*$' defined on $G$.
- A ring is an order triples $< R\,,+\,,\ .>$ where $R$ is a nonempty set and $+\,,\ .$ are two binary operation on $R$ satisfying the closure property for addition, associativity for addition, existence of identity w.r.t. addition, existence of inverse w.r.t. addition, commutative property for addition, closure property for $(\,.\,)$, Associative property for $(\,.)$ and Distributive laws of $(.)$ and $(+)$

- A non zero element ‘ $a$’ in a ring $R$ is called a ( proper ) zero divisor if there exist another non zero elements ‘$b$’ in $R$such that $ab = 0$ .
- A commutative ring $R$ without proper zero divisor is called an integral domain

## 5.12 ANSWERS TO CHECK YOUR PROGRESS

1.(a) False (b) False 2. Cyclic 3. Sub-group

## 5.13 POSSIBLE QUESTIONS

**Short Answer type Questions:**

1. Define semi-group.

2. Give an example of a semi group which is not group.

3. Define zero divisor of a ring.

4. Give an example of a ring which is not an integral domain

5. Define basis of a vector space.

6. Define linearly independent of the vector space.

7. Define order of a permutation group.

8. Under what condition a ring is said to be an integral domain.

**Long Answer type Question:**

1. Show that identity element in a group is Unique.

2. Show that the set of all positive rational numbers forms an abelian

group under the composition defined by $a * b = (ab/2)$

3. Show that every group of prime order is cyclic.

4. Show that union of two subspace may not be a subspace.

5. Show that intersection of two subgroup is again a subgroup.

6. Prove that two vectors are linearly independent then one of them is

Space for learners:

scalar multiple of the other.

7. Show that the vectors $(1,1,0,0)$, $(0,1,-1,0)$, $(0,0,0,3)$ in $\mathbb{R}^4(\mathbb{R})$are linearly independent.

8. Give an example of a ring which is not a field.

9. Show that every transposition is always an odd permutation.

10. Show that every field is an integral domain.

## 5.14 REFERENCES AND SUGGESTED READINGS

Modern Algebra by. Vasistha , A . R ., Krishna Publication, India 2018

***Space for learners:***

# UNIT 6: PROPOSITIONAL CALCULUS-I

Space for learners:

**Unit Structure:**



## 6.1INTRODUCTION

Mathematical logic is the science of reasons. Greek philosopher Aristotle (381-322 BC) first introduced the concept of logical reasoning. The mathematical logic compromise of two branches:(a) Propositional calculus, (b) Predicate calculus. But in this course, our discussion will restrict only propositional calculus. The branch of logic that deals with propositions is called propositional calculus. Propositional calculus is the study of the logical relationship between propositions. Propositional calculus forms the basis of all mathematical reasoning and it has many applications in computer science like design of computing machines, artificial intelligence, data structures for programming language etc.

## 6.2 UNIT OBJECTIVES

Space for learners:

After going through this unit, you will be able to:

- define propositions and examples of propositions
- define truth tables about different propositions
- know about negation of a proposition
- know about conjunction, disjunction, conditional and bi-conditional of two propositions

## 6.3 PROPOSITION

**Definition:**

A proposition (or statement) is a declarative sentence which is either true or false but not both.

The truth or falsity of a proposition is called its truth value.

Notations:

(a) If a proposition is true then its truth value is denoted by T

(b) If a proposition is false then its truth value is denoted by F

We now define Simple propositions and Compound propositions.

A simple proposition is a statement or assertion that must be true or false.

Many statements or propositions are constructed by combining one or more propositions, new propositions called compound propositions are formed existing propositions using logical connectives. A compound proposition is a combination of simple propositions and hence, can be broken down in primitive propositions.

### 6.3.1 Examples of proposition

**ILLUSTRATION 1:** Consider the following sentences

(i) New Delhi is the capital of India.

(ii) 7 is a prime number.

(iii) Every quadrilateral is a rectangle.

(iv) The earth is a planet.

(v) Three plus six is9.

(vi) The sun is a star.

(vii) Delhi is in America.

Each of the sentences (i), (ii), (iv) & (v) is a true declarative sentence and so each of them is a proposition.

Each of the sentences (iii), (vi), (vii) isa false declarative sentence and so each of them is a proposition.

All the above propositions are atomic propositions

**ILLUSTRATION 2:** Consider the following sentences:

i) Go to bed.

ii) Give me a glass of water.

iii) How are you?

iv) Where are you going?

v) May god bless you!

vi) May you livelong!

vii) $x + 2 = 5$

viii) $u + v < w$

Sentences (i) & (ii) are imperative sentences, so they are not propositions. Each of the sentences (iii) & (iv) is interrogative. So, they cannot be propositions. Similarly, (v) & (vi) are also not declarative sentences and hence not propositions. The expression (vii) and (viii) are not propositions, since the variables in these expressions have not been assigned values and hence, they are neither true or false.

***Space for learners:***

Space for learners:

**CHECK YOUR PROGRESS**

1.Define proposition.
2.Which of the following sentences are propositions? What are the truth values of those that are propositions?

(a) $3 + 4 = 7$

(b) $5 + 7 = 10$

(c) There are 35 days in a month.

(d) $x + 3 = 12$

(e)Answer these questions.

3.Write down the truth value of the following propositions:

(a) All the sides of a rhombus are equal in length.

(b) $\sqrt{3}$ is a rational number.

(c) The number 30 has four prime factors.

(d) Every square matrix is non-singular.

(e) $1 + \sqrt{8}$ is an irrational number.

## 6.4 Propositional variables

Now we will abbreviate propositions by using propositional variables. Each proposition will be represented by a propositional variable. Propositional variables are usually represented as lower-case letters, such as *p*, *q*, *r*, *s*, etc. The capital letters *A*, *B*, *C*, ..., *P*, *Q*, ... with the exception of T and F are also used. Each variable can take one of two values: true or false.

***Example 1: Consider the propositions***

***(i)*** Guwahati is in India.

***(ii)*** $6 + 8 = 14$

***(iii)*** The sun is shining.

Now we can assign propositional variables $p$, $q$, $r$ for the propositions (i), (ii) and (iii).

The propositions (i), (ii) and (iii) can be represented by

$p$: Guwahati is in India

$$q: 6 + 8 = 14$$

And $r$: The sun is shining

### 6.5 Truth Tables

A table which gives the truth values of a compound proposition in terms of its component parts is called a 'Truth Table'. A truth table consists of rows and columns. The initial columns are filled with the possible truth values of the component parts and the last column is filled with the truth values of the compound proposition on the basis of the truth values of the component parts written in the initial columns. If the compound proposition is consisting of $n$ component parts, then its truth table will contain $2^n$ rows.

A truth table displays the relationship between the truth values of propositions.

Note: Truth tables are very useful in the determination of propositions constructed from simple propositions.

### 6.6 Logical Connectives

Till now, we have considered simple or primary propositions which are declarative sentences, each of which cannot be expressed as a combination of more than one sentence. We often combine simple(primary) propositions to form compound propositions by using certain connecting words known as logical connectives or logical operators. Primary statements are combined by means of five logical connectives.

Five Basic Logical Connectives

| | Logical Connectives | Name of the Connectives | Symbols of the connectives |
|---|---|---|---|

**Space for learners:**

| | | | |
|---|---|---|---|
| 1 | Not | Negation (or Denial) | $\sim$ |
| 2 | And | Conjunction | $\wedge$ |
| 3 | Or | Disjunction | $\vee$ |
| 4 | If … then | Conditional | $\rightarrow$ |
| 5 | If and only if | Biconditional | $\leftrightarrow$ |

Space for learners:

Now we will discuss in details about these five logical connectives which will allow us to build up compound propositions and their truth values expressed in a tabular form, called Truth Table.

### 6.6.1 Negation

The denial of a proposition$P$ is called its negation and is written as $\sim P$and read as 'not $P$'. Negation of any proposition$P$ is formedbywriting"Itisnotthecasethat''or"Itisfalsethat''before $p$ or inserting in $p$ the word " not''.
Let us consider the proposition

$P$: All integers are rational numbers.

The negation of this statement is:
$\sim P$: It is not the case that all integers are rational numbers.

or

$\sim P$: It is false that all integers are rational numbers.

or

$\sim P$: It is not true that all integers are rational numbers.

Let us consider anotherthe proposition,

$P : 7 > 9$
The negation of this statement is$\sim P$:$\sim (7>9)$ or$\sim p$:$(7<9)$

**Truth Table of Negation:** If the truth value of "$P$'' is T, then the truth value of $\sim P$ is F. Also, if the truth value of "$P$''isF, then the truth value of $\sim P$ is T.

The truth table of for the negation of a proposition

Space for learners:

| $P$ | $\sim P$ |
|---|---|
| T | F |
| F | T |

Example 1: Write the negation of the following propositions:

(i)$\sqrt{7}$is arational

(ii)Every natural number is greater than zero.

(iii)All primes are odd.

(iv)All mathematicians are men.

**Solution:**

(i) Let $p$denote the given proposition i.e.,

$P$: $\sqrt{7}$ is arational.

The negation of this proposition is given by

$\sim P$:It is not the case that $\sqrt{7}$ is arational.

Or

$\sim P$: $\sqrt{7}$ is not arational.

Or

$\sim P$:It is false that $\sqrt{7}$ is arational.

(ii)The negation of the given proposition is

It is false that every natural number is greater than 0.

or

There exists a natural number which is not greater than0.

(iii)The negation of the given proposition is

There exists a prime which is not odd.

or

Some primes are not odd.
or

At least one prime is not odd.

(iv)The negation of the given proposition is:

Some mathematicians are not men.
or

There exists a mathematician who is not man.

or

At least one mathematician is not man.

or

It is false that all mathematicians are me

**CHECK YOUR PROGRESS**

Q4**.** Why Logical Connectives are used?
Q5. Write the negation of the following propositions-

i) Bangalore is the capital of Karnataka.
ii) The Earth is round.
iii) The Sun is cold.
iv) Some even integers are prime.
v) Both the diagonals of a rectangle have the same length.
vi) $4 + 7 = 10$
vii) Today is Monday
viii) If it snows, Samir does not drive the car.

### 6.6.2 Conjunction

The conjunction of two propositions $P$ and $Q$ is the proposition"$P$ *and* $Q$"which is denoted by$P \wedge Q$.$P$,$Q$ are called the components of $P \wedge Q$.

Illustrative Examples:

(i) The conjunction of the propositions:

*Space for learners:*

$P$:It is raining

$Q$: $2+2=4$ is

$P \wedge Q$:It israining and $2+2=4$.

(ii) Consider the proposition

$P$ : The Earth is round and the Sun is cold.

Its components are:

$Q$: The Earth is round.

$R$ : The Sun is cold.

**Truth table:** The statement $P \wedge Q$ has the truth value Twhenever both $P$ and $Q$ have the truth value T; Otherwise, it has the truth value F.

The truth table for conjunction of two propositions

| P | Q | P$\wedge Q$ |
|---|---|---|
| T | T | T |
| T | F | F |
| F | T | F |
| F | F | F |

### 6.6.3 Disjunction

The disjunction of the two propositions $P$ and $Q$ is the statement "$P$ *or* $Q$", denoted by $P \vee Q$. $P$,$Q$are called the components of $P \vee Q$.

Examples:

(i) Consider the compound proposition

$P$: Two lines intersect at a point or they are parallel.

The component propositions of are:

$Q$ : Two lines intersect at a point.

$R$ : Two lines are parallel.

(ii) Consider another proposition

**Space for learners:**

$P$: 45 is a multiple of 4 or 6.
Its component propositions are:
$Q$: 45 is a multiple of 4.

$R$: 45 is a multiple of 6.

**Truth table:** The statement $P \vee Q$has the truth value F only when both $P$ and $Q$ have the truth value F, $P \vee Q$ is true if either$P$ is true or $P$ is true (or both $P$ and $Q$ are true).

Truth table for disjunction of two propositions

| P | Q | P∨Q |
|---|---|---|
| T | T | T |
| T | F | F |
| F | T | F |
| F | F | F |

**Example 2:** Write the component propositions of the following compound propositions and check whether the compound proposition is true or false.

i) 50 is a multiple of both 2and5.

(ii) Mumbai is the capital of Gujrat or Maharashtra.

(iii) A rectangle is a quadrilateral or a 5-sidedpolygon.

**Solution:**

i) The component statements of the given statement are

$P$: 50 is multiple of2
$Q$ : 50 is multiple of 5

***Space for learners:***

We observe that both $P$and $Q$are true statements.

Therefore, the compound statement $P \wedge Q$ is true.

ii) The components proposition of the given proposition are

$P$:Mumbai is the capital of Gujrat.

$Q$: Mumbai is the capital of Maharashtra.

We find that $P$ is false and $Q$ is true. Therefore, the compound statement,$P \vee Q$is true.

iii) The component propositions are

$P$: A rectangle is a quadrilateral.

$Q$: A rectangle is a 5-sided polygon.

We observe that $P$ is true and $Q$ is false. Therefore, the compound proposition$P \vee Q$ is true.

**CHECK YOUR PROGRESS**

Q.6. Write the following propositions in symbolic form:

i) Pavan is rich and Raghav is not happy.

ii) Pavan is not rich and Raghav is happy.

iii) Naveen is poor but happy.

iv) Naveen is rich or unhappy

v) Naveen and Amal are both smart.

vi) It is not true that Naveen and Amal are both smart

vii) Naveen is poor or he is both rich and unhappy

vii) Naveen is neither rich nor happy.

Q.7**.** Write the component statements of the following compound statements and find true values of the compound statements.

***Space for learners:***

i) Delhi is in India and 2 + 2 =4.

ii) Delhi is in England and 2 + 2 =4.

iii) Delhi is in India and 2 + 2 = 5.

iv) Delhi is in England and 2 + 2 =5.

v) Square of an integer is positive or negative.

vi) The sky is blue and the grass is green.

vii) The earth is round or the sun is cold.

viii) All rational numbers are real and all real numbers are complex.

ix) 25 is a multiple of 5 and8.

x) 125 is a multiple of 7or8.

**6.6.4 Conditional Statements**

If P and Q are any two statements, then the statement "if P, then Q", is called a conditional statement. It is denoted by P→Q.

**Example:** Let P:Amulya works hard.
Q:Amulya will pass the examination.

Then P→Q: If Amulya works hard, then he will pass the examination.

The statement P is called the ***antecedent*** and Q is called the ***consequent*** in P→Q. The sign "→" is called the sign of implication. The conditional statement P→Q can also be read as:

i) P only if Q
ii) Q if P
iii) Q provided that P
iv) P is sufficient for Q
v) Q is necessary conditions for P
vi) P implies Q
(vii) Q is implied by P.

Space for learners:

Space for learners:

**Truth table:** If the antecedent P is true and the consequent Q is false, then the conditional statement P→Q is false, otherwise it is true as given in the following table.

The truth table for the conditional P→Q

| P | Q | P→Q |
|---|---|---|
| T | T | T |
| T | F | F |
| F | T | T |
| F | F | T |

Example 3: Write each of the following statements in the form "If–then"

i) You get job implies that your credentials are good.

ii) A quadrilateral is a parallelogram if its diagonals bisect each other.

iii) To get A+ in the class, it is necessary that you do all the exercises of the book.

**Solution:**

(i)The given statement can be written as "If you get a job, then your credentials are good."

(ii) The given statement can be written as-

"If the diagonal of a quadrilateral bisects each other, then it is a parallelogram".

Space for learners:

(iii) The given statement can be written as

"If you are to get A+ in the class, then you are to do all the exercises of the book".

Example 4: Writethefollowingconditionalstatemen tsinsymbolicformand hence, find truthvalues.

i) If 2 + 2 = 4, then Guwahati is inAssam
ii) If 2 + 2 = 4, then Guwahati is inBihar
iii) if 2 + 2 = 5, then Guwahati is inAssam
iv) If2+2=5,thenGuwahatiisinBihar

**Solution:**LetP: 2 + 2 =4

Q: Guwahati is in Assam

R: 2 + 2 =5

S: Guwahati is in Maharashtra

Then i) The given statement is P→Q

As P and Q have truth values T each, so P→Q has truth value T, i.e., the given conditional statement is true.

ii) The given statement isP→S

| P | S | P→S |
|---|---|---|
| T | F | F |

So, the given statement is false.

iii) The given statement isR→Q

| R | Q | R→Q |
|---|---|---|
| F | T | T |

So, the given statement is true.

iv) The given statement is $R \rightarrow S$

| R | S | $R \rightarrow S$ |
|---|---|---|
| F | F | T |

So, the given statement is true.

Space for learners:

**CHECK YOUR PROGRESS**

Q.8. Write down the truth value of each of the following implication.

i) If $3 + 2 = 7$, then Paris is the capital ofIndia.

ii) If $3 + 4 = 7$, then $3 > 7$

iii) If $4 > 5$, then $5 < 6$.

iv) If $7 > 3$, then $6 < 14$

v)If $7 > 3$, then $14 > 9$.

### 6.6.5 Biconditional Statements

If P and Q are any two statements, then the statement 'P if and only if Q' is called a biconditional statement which is denoted by $P \leftrightarrow Q$.'P if and only if Q' is also abbreviated as "P iff Q".

The biconditional 'P if and only if Q' is regarded as having the same meaning as 'if P, then Q and if Q, then P'. So, the biconditional $P \leftrightarrow Q$ is the conjunction of the conditionals $P \rightarrow Q$ and $Q \rightarrow P$ i.e., $(P \rightarrow Q) \wedge (Q \rightarrow P)$ is same as$P \leftrightarrow Q$.

The statement $P \leftrightarrow Q$ can also be read as

a) Q if and only if P

b) P implies Q and Q implies P

c) P is necessary and sufficient condition for Q
d) Q is necessary and sufficient condition for P

The truth table for the biconditional $P \leftrightarrow Q$

| P | Q | $P \leftrightarrow Q$ |
|---|---|---|
| T | T | T |
| T | F | F |
| F | T | F |
| F | F | T |

Thus, the biconditional $P \rightarrow Q$ is true only when both P, Q have identical truth values, otherwise it is false.

Examples:

1.A triangle is equilateral if and only if it is equiangular.

2.8 > 4 if and only if 8 – 4 is positive.

3.2+2=4ifandonlyifitisraining.

4.Two lines are parallel if and only if they have the same slope.

**Example 5:**Write the truth value of each of the following biconditional statements.
i)4 > 2 if and only if 0 < 4 – 2.

Space for learners:

ii)$3 < 2$ if and only if $2 < 1$.

iii)$3 + 5 > 7$ if and only if $4 + 6 < 10$.

iv)$2 + 5 = 7$ if and only if Guwahati is in Assam.

**Solution:**i) L e t P: $4 > 2$

Q: $0 < 4 - 2$

Then, the given statement is $P \rightarrow Q$.

Clearly, P is true and Q is true and therefore, $P \rightarrow Q$ is true.

Hence, the given statement is true, and its truth value is T.

(iv) Let P:
$3 < 2$

Q: $2 < 1$

Then, the given statement is $P \rightarrow Q$.

Clearly, P is false and Q is false and therefore, $P \rightarrow Q$ is true.
Hence, the given statement is true, and its truth value is T.

(v) Let P: $3 + 5 > 7$

Q: $4 + 6 < 10$

Then, the given statement is $P \rightarrow Q$.

Clearly, P is true and Q is false and therefore, $P \rightarrow Q$ is false.
Hence, the given statement is false and therefore, its truth value is F.

(vi) Let P: $2+5=7$

Q: Guwahati is in Assam

Then, the given statement is $P \rightarrow Q$. As P is false, Q is true, the given statement is false.

**CHECK YOUR PROGRESS**

Q.9.Write down the truth value of each of the following:

i) $3 + 5 = 8$ if and only if $4 + 3 = 7$.

ii) 4 is even if and only if 1 is prime.

iii) 6 is odd if and only if 2 is odd.

iv) $2 + 3 = 5$ if and only if $3 > 5$.

**Space for learners:**

v) $4 + 3 = 8$ if and only if $5 + 4 = 10$.

vi) $2 < 3$ if and only if $3 < 4$.

## 6.7 SUMMING UP

- A primary proposition is a declarative sentence which cannot be further broken down or analyzed into simpler sentences.
- New propositions can be formed from primary propositions through the use of sentential connectives. The resulting statements are called compound propositions.
- The sentential connectives are also called logical connectives. These connectives are: NOT (negation), AND (conjunction), OR (disjunction), IF– THEN (conditional), IF AND ONLY IF (Bi-conditional).
- Truth tables have been introduced in the definitions of the connectives.
- The statement P is called the antecedent and Q is called the consequent in $P \rightarrow Q$.

## 6.8 ANSWERS TO CHECK YOUR PROGRESS

**Ans. to Q. No.1:** A proposition is a declarative sentence which is either True or False but not both.

**Ans. to Q. No.2:** (a) Proposition, T
(b) Proposition, F
(c) Proposition, F
(d) Not proposition
(e) Not proposition

**Ans. to Q. No.3:** (a) T (b) F (c) F (d) F (e) T

**Ans. to Q. No.4:** Logical connectives are used to form new propositions or compound
propositions.

**Ans. to Q. No.5:** i) Bangalore is not the capital of Karnataka.

(ii) The earth is not round.

***Space for learners:***

(iii)The sun is not cold.

(iv)No even integer is prime.

(v)There is at least one rectangle whose both diagonals do not have the same length.

(vi)$4 + 7 \neq 10$

(vii)Today is not Monday.

(viii)It snows and Samir drives the car.

**Ans. to Q. No.6:**

i) $P \wedge \sim Q$, where P: Pavan isrich
Q: Raghav is happy

ii) $\sim P \wedge Q$

iii) $\sim R \wedge H$, where R: Naveen isrich
H: Naveen is happy

iv) $R \vee \sim H$

v) $P \wedge Q$, where P: Naveen issmart
Q: Amal is smart

vi) $\sim (P \wedge Q)$

vii) $\sim R \vee (R \vee \sim H)$, where R: Naveen isrich
R: Naveen is happy

viii) $\sim R \wedge \sim H$

**Ans. to Q. No.7:**

i) P: Delhi is inIndia

Q: 2 + 2 = 4

The compound statement is true.

(ii) P: Delhi in England

Q : 2 + 2 =4
The compound statement is false.

(iii) P:
Delhi is
in India

Q: 2 +

Space for learners:

2 =5

The compound statement is false.

(iv)P:
Delhi is in
England

Q: 2 + 2 =5

The compound statement is false.

(v)P: Square of an integer is positive

Q: Square of an integer is negative

The compound statement is true.

(vi)P: The sky is blue

Q: The grass is green

The compound statement is true.

(vii) P: The earth is round

Q: The sun is cold

The compound statement is true.

(viii) P: All rational numbers are real

Q: All real numbers are complex.

The compound statement istrue.

(ix)P: 25 is a multiple of 5

Q: 25 is a multiple of8

The compound statement is false.

(x) P: 125 is a multiple of 7

Q: 125 is a multiple of8

The compound statement is false.

Ans. to Q. No.8: i) True, ii) False, iii) True, iv) False, v)True

Ans. to Q. No.9: i) True, ii) False, iii) True, iv) False, v) True,vi) True.

## 6.9 POSSIBLE QUESTIONS

**Q.1.** Find out which of the following sentences are propositions and which are not .Justify your answer.

Space for learners:

i) The real number x is less than2.

ii) All real numbers are complex numbers.

iii) Listen to me, Ravi!

**Q.2.** Find the component propositions of the following and check whether they are true ornot:

i) The sky is blue and the grass is green.

ii) The earth is round or the sun is cold.

iii) All rational numbers are real and all real numbers are complex

iv) 25 is a multiple of 5 and8.

**Q.3.** Write the component propositions of each of the following statements. Also, check whether the statements are true or not.

i) Sets A and B are equal if and only if ($A \subseteq B$ and $B \subseteq A$).

ii) $|a| < 2$ if and only if ($a > -2$ and $a < 2$)

iii) $\triangle ABC$ is isosceles if and only if $\angle B = \angle C$.

iv) $7 < 5$ if and only if 7 is not a prime number.

v) ABC is a triangle if and only if $AB + BC > AC$.

Q.4.If P is true and Q is false, then find truth values of

i) $P \wedge (\sim Q)$, ii) $\sim P \vee Q$, iii) $\sim P \rightarrow Q$,

iv) $P \rightarrow (\sim Q)$, v) $\sim (P \rightarrow Q)$, iv) $P \wedge Q$

5. What is proposition? Explain with illustration.

6. What do you mean by propositional variables?

7. Discuss truth table with example.

8. What are logical connectives? Explain the logical connectives with their corresponding truth tables.

9. Discuss how conditional statements are implemented using propositions.

Space for learners:

## 6.10 REFERENCES AND SUGGESTED READINGS

Space for learners:


- Rao, G. S. S. Bhishma. *Discrete Structures and Graph Theory*. SciTech Publications (India) PVT Ltd,2003
- Sharma, J.K. *Discrete Mathematics*. Macmillan Publishers India Ltd,2011
- Balaji, G. *Discrete Mathematics*. Balaji Publishers,2015
- Singaravelu, A. &Jeyaraman, M.P. *Discrete Mathematics*. Meenakshi Agency,2016
- Satyanarayana, B. & Prasad, K.S. *Discrete Mathematics and Graph Theory*. PHI Learning Private Limited,2009
- Rao, G.S. *Discrete Mathematical Structure*. New Age international (P) Limited,2009
- Singh, Y.N. *Mathematical Foundation of Computer Science*, New Age international Publishers, 2005
- Lipschutz, S & Lipson, M.L. Schaum's outline of *Theory and Problems of Discrete Mathematics*, McGraw-Hill,2007
- Chandrasekaran, N &Umaparvathi, M. *Discrete Mathematics*, PHI Learning Private Limited,2010
- Tremblay, J.P. & Manohar, *Discrete Mathematical Structures with Applications to Computer Science,* Tata McGraw Hill Education Private Limited,1997.

# UNIT 7: PROPOSITIONAL CALCULUS-II

Space for learners:

**Unit Structure:**



## 7.1 INTRODUCTION

The notion of a proposition has already been introduced in the previous unit. In this unit, we define statement formula. Also, we define tautology and contradiction of statement formulas and discuss equivalence of two statement formulas. In this unit, we will also discuss tautological implication of two statement formulas and will define the concept of two-state devices.

## 7.2 UNIT OBJECTIVES

After going through this unit, you will be able to

- define statement (or proposition) formulas
- define tautology and contradiction
- know about logical equivalence of two different

statement formulas

- know about some important equivalence formulas

### 7.3 STATEMENT FORMULA

Statements which do not contain any connective are called simple or primary or atomic statements.On the other hand, the statements which contain one or more primary statements and at least one connective are called composite or compound statements.

For example, let P and Q be any two simple statements. some of the compound statements formed by P and Q are–

$\sim P$, $P \vee Q$, $(P \vee Q) \wedge (\sim P)$, $P \vee (\sim P)$, $(P \vee \sim Q) \wedge P$.

statement variables P and Q. Therefore, P and Q are called components of the statement formulas.

Definition:

A statement formula is an expression which is a string consisting of propositional variables, parenthesis and connectives. Statement formulas are constructed from simple propositions using logical connectives.

An example of Statement formula is $P \wedge (Q \vee R) \rightarrow S$.

A statement formula alone has no truth value. It has truth value only when the statement variables in the formula are replaced by definite statements and it depends on the truth values of the statements used in replacing the variables.

**The truth table of a statement formula (Proposition):** Truth table has already been introduced in the previous unit. In general, if there are 'n' distinct components in a statement formula. We need to consider $2^n$ possible combinations of truth values in order to obtain the truthtable.

For example, if any statement formula has two component statements namely P and Q, then $2^2$ possible combinations of truthvalues must beconsidered.

**Illustrative Examples:**

Example 1. Construct the truth table for

(a) $\sim(P \wedge Q)$ (b) $(\sim P) \vee (\sim Q)$

Space for learners:

Solution:

(a) Truth table:

| P | Q | $P \wedge Q$ | $\sim(P \wedge Q)$ |
|---|---|---|---|
| T | T | T | F |
| T | F | F | T |
| F | T | F | T |
| F | F | F | T |

(b) Truth table:

| P | Q | $\sim P$ | $\sim Q$ | $(\sim P) \vee (\sim Q)$ |
|---|---|---|---|---|
| T | T | F | F | F |
| T | F | F | T | T |
| F | T | T | F | T |
| F | F | T | T | T |

Example 2. Construct the truth table for

$$\sim P \wedge Q$$

Solution:

Truth table

| P | Q | $\sim P$ | $\sim P \wedge Q$ |
|---|---|---|---|
| T | T | F | F |
| T | F | F | F |
| F | T | T | T |
| F | F | T | F |

Example 3: Construct the truth table for P → (Q →R).

***Space for learners:***

Solution:P, Q and R are the three statement variables that occur in this formula P → (Q → R). There are $2^3 = 8$ different sets of truth value assignments for the variables P, Q and R.

The following table is the truth table for P → (Q → R):

| **P** | **Q** | **R** | **Q → R** | **P → (Q → R)** |
|---|---|---|---|---|
| T | T | T | T | T |
| T | T | F | F | F |
| T | F | T | T | T |
| T | F | F | T | T |
| F | T | T | T | T |
| F | T | F | T | T |
| F | F | T | T | T |
| F | F | F | T | T |

Example 4: Construct the truth table for $P \wedge (P \vee Q)$

Solution:

Truth table

| **P** | **Q** | $\boldsymbol{P \vee Q}$ | $\boldsymbol{P \wedge (P \vee Q)}$ |
|---|---|---|---|
| T | T | T | T |
| T | F | T | T |
| F | T | T | F |
| F | F | F | F |

Example 5: Construct the truth table for $(P \vee Q) \vee \sim P$
Solution:

| **P** | **Q** | $\boldsymbol{P \vee Q}$ | $\boldsymbol{\sim P}$ | $\boldsymbol{(P \vee Q) \vee \sim P}$ |
|---|---|---|---|---|
| T | T | T | F | T |
| T | F | T | F | T |
| F | T | T | T | T |
| F | F | F | T | T |

**CHECK YOUR PROGRESS**

***Space for learners:***

**Q.1.** Construct the truth tables for the followingformulas

a) $\sim(\sim P \wedge \sim Q)$

b) $(\sim P \vee Q) \wedge (\sim Q \vee P)$

c) $(P \wedge Q) \rightarrow (P \vee Q)$

d) $(Q \wedge (P \rightarrow Q)) \rightarrow P$

Space for learners:

**7.4 TAUTOLOGY**

We have already defined truth table of a statement formula. In general, the final column of a given formula contains both T and F. There aresomeformulaswhosetruthvaluesarealwaysToralwaysFregardless of the truth value assignments to the variables. This situation occurs because of the special construction of theseformulas.

**Definition:** A statement formula which is true regardless of the truth values of the statements which replace the variables in it is called a universally valid formula or a tautology or a logicaltruth.

A straight forward method to determine whether a given formula is a tautology is to construct its truth table. In the table, if the column below the statement formula contains T only, then it is a tautology. The conjunction of two tautologies is also a tautology. Let us denote by A and B two statement formulas which are tautologies. If we assign any truth values of the variables of A and B, then the truth values of both A and B will be T. Thus, the truth value of A ∧ B will be T, so that A ∧ B will be atautology.

Example 6: Verify whether P v (~P) is a tautology.

Solution:

| **P** | $\sim P$ | $P \vee \sim P$ |
|---|---|---|
| T | F | T |
| F | T | T |

As the entries in the last column are T, the given formula is a tautology.

Example 7: Show that the proposition $(P \vee Q) \leftrightarrow (Q \vee P)$is atautology.

Solution:

| P | Q | $P \vee Q$ | $Q \vee P$ | $(P \vee Q) \leftrightarrow (Q \vee P)$ |
|---|---|---|---|---|
| T | T | T | T | T |
| T | F | T | T | T |
| F | T | T | T | T |
| F | F | F | F | T |

The last column entries are T. Therefore, given formula is a tautology.

Example 8: Verify whether $(P \rightarrow Q) \wedge (Q \rightarrow P)$is atautology.
Solution:

| P | Q | $P \rightarrow Q$ | $Q \rightarrow P$ | $(P \rightarrow Q) \wedge (Q \rightarrow P)$ |
|---|---|---|---|---|
| T | T | T | T | T |
| T | F | F | T | F |
| F | T | T | F | F |

All the entries in the resulting column are not T, hence the given proposition is not a tautology.
Example 9: Show that the proposition $(P \wedge \sim Q) \vee \sim(P \wedge \sim Q)$ is a tautology.
Solution:

| P | Q | ~Q | $P \wedge \sim Q$ | $\sim(P \wedge \sim Q)$ | $(P \wedge \sim Q) \vee \sim(P \wedge \sim Q)$ |
|---|---|---|---|---|---|
| T | T | F | F | T | T |
| T | F | T | T | F | T |
| F | T | F | F | T | T |
| F | F | T | F | T | T |

As the entries in the last column are T, the given proposition is a tautology.

Space for learners:

Example 10: Verify that the proposition $P \vee \sim(P \wedge Q)$ is a tautology.

Solution:

| **P** | **Q** | $\boldsymbol{P \wedge Q}$ | $\boldsymbol{\sim(P \wedge Q)}$ | $\boldsymbol{P \vee \sim(P \wedge Q)}$ |
|---|---|---|---|---|
| T | T | T | F | T |
| T | F | F | T | T |
| F | T | F | T | T |
| F | F | F | T | T |

As the entries in the last column are T, the given proposition is a tautology.

**CHECK YOUR PROGRESS**

**Q.2.** Prove that the following are tautologies (using truthtables):

a) $Q \vee (P \wedge \sim Q) \vee (\sim P \wedge \sim Q)$

b) $(P \rightarrow Q) \leftrightarrow (\sim P \vee Q)$

c) $\sim (P \vee Q) \vee (\sim P \wedge Q) \vee P$

**Q.3.** Show that $((\sim P) \vee (\sim Q)) \vee P$is a tautology.

## 7.5 CONTRADICTION

**Definition:** A statement formula which is false regardless of in the truth values of the statements which replace the variables in it is called a contradiction., if each entry in the final column of the truth table of a statement formula is F only then it is called as contradiction.

Clearly, the negation of a contradiction is a tautology. We may call a statement formula which is a contradiction as identically false.

Example 11: Verifythat $P \wedge (\sim P)$ is acontradiction.

Solution:

| **P** | **~P** | **P ∧ (~P)** |
|---|---|---|
| T | F | F |

Space for learners:

| | | |
|---|---|---|
| F | T | F |

Since the last column has F only, the statement formula is a contradiction.

Space for learners:

Example 12: Verify the statement (P ∧ Q) ∧~ (P vQ).

Solution:

| P | Q | P ∧ Q | P vQ | ~ (P vQ) | (P ∧ Q) ∧ ~ (P vQ) |
|---|---|---|---|---|---|
| T | T | T | T | F | F |
| T | F | F | T | F | F |
| F | T | F | T | F | F |
| F | F | F | F | T | F |

Since the truth value of (P ∧ Q) ∧ ~ (P vQ) is F, for all values of P and Q, the proposition is a contradiction.

Example 13: Provethat, if A (p, q, ⋯) is a tautology, then ~A (p, q,⋯) is a contradiction andconversely.

**Solution:**Since a tautology is always true, the negation of a tautology is always false i.e. is a contradiction andvice-versa.

**7.6 LOGICAL EQUIVALENCE**

Two statement formulas A (P, Q, ...) and B (P, Q, ...) are said to be logically equivalent or simply equivalent if they have identical truth tables. In other words, corresponding to identical truth values of P, Q, ... the truth valuesofA&Bmustbesame.IfAandBareequivalent,weshallwriteA≡ B or A ⟺B.

Example 14: Show that P is equivalent to the following formulae.

(i)$\sim\sim P$ (ii) $P \wedge P$ (iii) $P \vee P$ (iv) $P \vee (P \wedge Q)$ (v) $P \wedge (P \vee Q)$

Solution:

| P | Q | $\sim P$ | $\sim\sim P$ | $P \wedge P$ | $P \vee P$ | $P \wedge Q$ | $P \vee (P \wedge Q)$ | $P \vee Q$ | $P \wedge (P \vee Q)$ |
|---|---|---|---|---|---|---|---|---|---|
| T | T | F | T | T | T | T | T | T | T |
| T | F | F | T | T | T | F | T | T | T |
| F | T | T | F | F | F | F | F | T | F |
| F | F | T | F | F | F | F | F | F | F |

Here the 4th ,5th ,6th ,7th ,8th ,10th columns give the truth values of the formulas. The columns 1,4,5,6,8,10 have the identical truth values. Hence P is equivalent to all given formulas.

Example15:ProvethatP∨Q⇔~ (~P∧~Q)

Solution:

| P | Q | P∨Q | ~P | ~Q | ~P∧~Q | ~ (~P∧~Q) |
|---|---|---|---|---|---|---|
| T | T | T | F | F | F | T |
| T | F | T | F | T | F | T |
| F | T | T | T | F | F | T |
| F | F | F | T | T | T | F |

The truth table shows that P∨Qand ~ (~P∧~Q) have identical truth value column. So, P∨Q⇔~ (~P∧~Q).

Example16: Prove that P → Q ⇔(~P ∨ Q)
Solution:

| P | Q | P → Q | ~P | ~P ∨ Q |
|---|---|---|---|---|
| T | T | T | F | T |
| T | F | F | F | F |
| F | T | T | T | T |
| F | F | T | T | T |

Here,columnsofP→Qand~P ∨ Qareidentical.

Space for learners:

Hence, $P \rightarrow Q \Leftrightarrow (\sim P \vee Q)$.

Space for learners:

**CHECK YOUR PROGRESS**

Q.4.Explain the equivalence of propositions.

Q.5.Show the following equivalences using truth tablemethod:

a) $\sim(P \rightarrow Q) \Leftrightarrow P \wedge \sim Q$

b) $P \leftrightarrow Q \Leftrightarrow (P \rightarrow Q) \wedge (Q \rightarrow P)$

c) $P \rightarrow Q \Leftrightarrow \sim Q \rightarrow \sim P$

### 7.6.1 EQUIVALENT FORMULAS

Using respective truth tables, we can prove the following equivalence:

| | |
|---|---|
| Idempotent Laws | (i) $P \vee P \Leftrightarrow P$<br>(ii) $P \wedge P \Leftrightarrow P$ |
| Associative Laws | (i) $(P \wedge Q) \wedge R \Leftrightarrow P \wedge (Q \wedge R)$<br>(ii) $(P \vee Q) \vee R \Leftrightarrow P \vee (Q \vee R)$ |
| Commutative Laws | (i) $P \vee Q \Leftrightarrow Q \vee P$<br>(ii) $P \wedge Q \Leftrightarrow Q \wedge P$ |
| De Morgan's Laws | (i) $\sim(P \wedge Q) \Leftrightarrow \sim P \vee \sim Q$<br>(ii) $\sim(P \vee Q) \Leftrightarrow \sim P \wedge \sim Q$ |
| Distributive Laws | (i) $P \wedge (Q \vee R) \Leftrightarrow (P \wedge Q) \vee (P \wedge R)$ |

| | |
|---|---|
| | (ii) $P \vee (Q \wedge R) \Leftrightarrow (P \vee Q) \wedge (P \vee R)$ |
| Complement Laws | (i) $P \wedge \sim P \Leftrightarrow F$<br>(ii) $P \vee \sim P \Leftrightarrow T$ |
| Dominance Laws | (i) $P \vee T \Leftrightarrow T$<br>(ii) $P \wedge F \Leftrightarrow F$ |
| Identity Laws | (i) $P \wedge T \Leftrightarrow P$<br>(ii) $P \vee F \Leftrightarrow P$ |
| Absorption Laws | (i) $P \vee (P \wedge Q) \Leftrightarrow P$<br>(ii) $P \wedge (P \vee Q) \Leftrightarrow P$ |
| Double negation Law | $\sim(\sim P) = P$ |
| Contra positive Law | $P \to Q \Leftrightarrow \sim Q \to \sim P$ |
| Conditional as disjunction | $P \to Q \Leftrightarrow \sim P \vee Q$ |
| Biconditional as conditional | $P \leftrightarrow Q \Leftrightarrow (P \to Q) \wedge (Q \to P)$ |
| Exportation Laws | $P \to (Q \to R) \Leftrightarrow (P \wedge Q) \to R$ |

Check yourself the above formulas as an exercise by truth table technique. Here, T and F respectively stands true statement and false statement.

**Replacement Process:** Consider the formula A: P → (Q→R).The formula Q→R is a part of the formula A. If we replace Q→R by an equivalent formula ~Q∨R in A, we get another formula B:P→(~Q∨R). We can easily verify that the formulas A and B are equivalent to each other. This process of obtaining B from A is known as the replacement process. Using the laws stated in **7.6.1**,

***Space for learners:***

we can also establish equivalence of statement formulas without using truth tables.

**Illustrative Examples:**

Example 17: Provethat,$(P\rightarrow Q)\wedge(R\rightarrow Q)\Leftrightarrow(P\wedge R)\rightarrow Q$

**Solution:** $(P \rightarrow Q) \wedge (R \rightarrow Q)$

$\Leftrightarrow(\sim P \vee Q) \wedge (\sim R \vee Q)$
$\Leftrightarrow(\sim P\wedge\sim R)\vee Q$ [Distributive Law]
$\Leftrightarrow\sim (P \vee Q)\vee Q$ [DeMorgan's Law]
$\Leftrightarrow(P\wedge R)\rightarrow Q$

Example18: Provethat,$(\sim P\wedge(\sim Q\wedge R))\vee(Q\wedge R)\vee(P\wedge R)\Leftrightarrow R$

**Solution:**

$(\sim P\wedge(\sim Q\wedge R))\vee(Q\wedge R)\vee(P\wedge R)$

$\Leftrightarrow((\sim P \wedge \sim Q)\wedge R) \vee ((Q \vee P)\wedge R)$ (Associative Law & distributiveLaw)

$\Leftrightarrow(\sim (P\vee Q)\wedge R)\vee((Q\vee P)\wedge R)$
[DeMorgan's Law]

$\Leftrightarrow(\sim (P\vee Q)\vee(P\vee Q))\wedge R$
(Distributive Law)

$\Leftrightarrow T \wedge R$Since $\sim S \vee S \Leftrightarrow T$

$\Leftrightarrow R$ as $T \wedge R \Leftrightarrow R$

Example 19: Show that $P \rightarrow (Q \rightarrow R) \Leftrightarrow P \rightarrow (\sim Q \vee R) \Leftrightarrow (\sim P \wedge Q) \vee R$

Solution:$P \rightarrow (Q \rightarrow R)$

$\Leftrightarrow P \rightarrow (\sim Q \vee R)$ [$\because P \rightarrow Q \Leftrightarrow \sim P \vee Q$]

$\Leftrightarrow \sim P \vee (\sim Q \vee R)$ [$\because P \rightarrow Q \Leftrightarrow \sim P \vee Q$]

$\Leftrightarrow (\sim P \vee \sim Q) \vee R$ [ by Associative Law]

$\Leftrightarrow \sim(P \wedge Q) \vee R$ [ By De Morgan's Law]

Hence $P \rightarrow (Q \rightarrow R) \Leftrightarrow P \rightarrow (\sim Q \vee R) \Leftrightarrow (\sim P \wedge Q) \vee R$

***Space for learners:***

Space for learners:

**CHECK YOUR PROGRESS**

Q.6. Provethat:

a) $(P \to Q) \wedge (R \to Q) \Leftrightarrow (P \vee R) \to Q$

b) $(P \vee Q) \wedge \sim(\sim P \wedge Q) \Leftrightarrow P$

c) $(P \to Q) \to Q \Leftrightarrow (P \vee Q)$

## 7.7 TAUTOLOGICAL OR LOGICAL IMPLICATIONS

**Definition:** A statement A is said to tautologically or logically imply a statement B if and only if A→B is a tautology. In this case, we write A→B, read as “A tautologically implies B” or “A logically impliesB”. We shall denote this idea by A⟹B which is read as “A implies B”’.

**Note:** Learner should be very cautious with the following four notations:

(1) → means the connective conditional

(2) ↔ means the connective Biconditional

(3) ⟺ means equivalent

(4) ⟹ means tautological implications.

**Let us Know**

i) ⟹ is not connective,A⟹B is not a statement formula.
ii) A⟹B states that A→B is a tautology or A logically implies B.
iii) A⟹B guarantees that B has the truth value T whenever A has the truth value T.
iv) By constructing the truth table, we can determine A⟹B.
v) A⟺B if and only if A⟹B and B⟹A i.e., if each of two formulas A and B tautologically or logically implies the other, then A and B are equivalent.

**Illustrative Examples:**

Example 20: Show that $(P \wedge Q) \Longrightarrow (P \to \mathrm{Q})$

Solution: To prove the given proposition, it is enough to prove that $(P \wedge Q) \to (P \to \mathrm{Q})$is a tautology

| **P** | **Q** | $\boldsymbol{P \wedge Q}$ | $\boldsymbol{P \to \mathrm{Q}}$ | $\boldsymbol{(P \wedge Q) \to (P \to \mathrm{Q})}$ |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| T | T | T | T | T |
| T | F | F | F | T |
| F | T | F | T | T |
| F | F | F | T | T |

Space for learners:

Since the last column of the truth table of $(P \wedge Q) \rightarrow (P \rightarrow Q)$ contains only T's, so $(P \wedge Q) \rightarrow (P \rightarrow Q)$ is a tautology.

Hence $(P \wedge Q) \Longrightarrow (P \rightarrow Q)$

Example 21: Prove that $(P \rightarrow (Q \rightarrow R)) \Longrightarrow ((P \rightarrow Q) \rightarrow (P \rightarrow R))$ by constructing the truth table.

Solution:

To prove the given proposition, it is enough to prove that $(P \rightarrow (Q \rightarrow R)) \rightarrow ((P \rightarrow Q) \rightarrow (P \rightarrow R))$ is a tautology.

| **P** | **Q** | **R** | $Q \rightarrow R$ | $P \rightarrow Q$ | $P \rightarrow R$ | $(P \rightarrow (Q \rightarrow R))$ | $(P \rightarrow Q) \rightarrow (P \rightarrow R)$ | $(P \rightarrow (Q \rightarrow R)) \rightarrow ((P \rightarrow Q) \rightarrow (P \rightarrow R))$ |
|---|---|---|---|---|---|---|---|---|
| T | T | T | T | T | T | T | T | T |
| T | T | F | F | T | F | F | F | T |
| T | F | T | T | F | T | T | T | T |
| T | F | F | T | F | F | T | T | T |
| F | T | T | T | T | T | T | T | T |
| F | T | F | F | T | T | T | T | T |
| F | F | T | T | T | T | T | T | T |
| F | F | F | T | T | T | T | T | T |

Since the last column of the truth table of $(P \rightarrow (Q \rightarrow R)) \rightarrow ((P \rightarrow Q) \rightarrow (P \rightarrow R))$ contains only T's, so $(P \rightarrow (Q \rightarrow R)) \rightarrow ((P \rightarrow Q) \rightarrow (P \rightarrow R))$ is a tautology.

Hence, $(P \rightarrow (Q \rightarrow R)) \Longrightarrow ((P \rightarrow Q) \rightarrow (P \rightarrow R))$

**Some Important Logical Implications:**

1. $P \wedge Q \Longrightarrow P$
2. $P \wedge Q \Longrightarrow Q$
3. $P \Longrightarrow P \vee Q$
4. $\sim P \Longrightarrow P \rightarrow Q$
5. $Q \Longrightarrow P \rightarrow Q$
6. $\sim (P \rightarrow Q) \Longrightarrow P$
7. $\sim (P \rightarrow Q) \Longrightarrow \sim Q$
8. $P \wedge (P \rightarrow Q) \Longrightarrow Q$

9.$(P \rightarrow Q) \wedge (Q \rightarrow R) \Rightarrow P \rightarrow R$

10. $(P \vee Q) \wedge (P \rightarrow R) \wedge (Q \rightarrow R) \Rightarrow R$

Check yourself the above logical implications by using the truth table.

Example 22: Show that $P \wedge (P \rightarrow Q) \Rightarrow Q$ without constructing the truth table.

Solution:We have to prove that $[P \wedge (P \rightarrow Q)] \rightarrow Q$ is a tautology.

$[P \wedge (P \rightarrow Q)] \rightarrow Q$

$\Leftrightarrow [P \wedge (\sim P \vee Q)] \rightarrow Q$ $[\because P \rightarrow Q \Leftrightarrow \sim P \vee Q]$

$\Leftrightarrow \sim [P \wedge (\sim P \vee Q)] \vee Q$ $[\because P \rightarrow Q \Leftrightarrow \sim P \vee Q]$

$\Leftrightarrow [\sim P \vee \sim(\sim P \vee Q)] \vee Q$ [DeMorgan's Law]

$\Leftrightarrow [\sim P \vee (P \wedge \sim Q)] \vee Q$ [DeMorgan's Law]

$\Leftrightarrow [(\sim P \vee P) \wedge (\sim P \vee \sim Q)] \vee Q$ [Distributive Law]

$\Leftrightarrow [T \wedge (\sim P \vee \sim Q)] \vee Q$ [Complement Law]

$\Leftrightarrow [\sim P \vee \sim Q] \vee Q$ [Identity Law]

$\Leftrightarrow [\sim P \vee (\sim Q \vee Q)]$ [Associative Law]

$\Leftrightarrow \sim P \vee T$ [Complement Law]

$\Leftrightarrow T$ [Identity Law]

Hence, $P \wedge (P \rightarrow Q) \Rightarrow Q$

**CHECK YOUR PROGRESS**

Q 7.Show the following logical implications using the truth table:

a) $Q \Rightarrow P \rightarrow Q$

b) $\sim(P \rightarrow Q) \Rightarrow \sim Q$

Q.8.Show the following logical implications without constructing the truth tables:

a) $P \wedge Q \Rightarrow P \vee Q$

b) $P \wedge Q \Rightarrow P \rightarrow Q$

### 7.8 Two-State Devices

Let us consider the example of an electric switch which is used for turning "On" and "Off "an electric light. It has wo states "On" and

***Space for learners:***

"Off ". So, it is a two-statedevice. Let us consider another example of a magnetic core which is used in computer. In magnetic core, there lies a doughnut-shaped metal disc with a wire coil wrapped around it. It may be magnetized in one direction, if current is passed through the coil in one way and may be magnetized in the opposite direction, it the current is reversed. So, the magnetic core is a two-state device.

Space for learners:

## 7.9 SUMMING UP

- A statement formula is an expression which is a string consisting of (capital letters with or without subscripts), parentheses and connective symbols (∨, ∧, →, ↔, ~), which produces a statement when the variables are replaced by statements.
- A statement formula which is true regardless of the truth values of the statements which replace the variables in it is called a universally valid formula or a tautology or a logical truth.
- A statement formula which is false regardless of the truth values of the statements which replaces the variables in it a contradiction.
- The statement formulas A and B are equivalent provided A ↔B is a tautology; and conversely, if A↔B is a tautology, then A and B are equivalent. We shall represent the equivalence of A and B by writing "A ⇔B" which is read as "A is equivalent to B."
- A statement A is said is to tautologically imply a statement B if and only if A → B is a tautology. We shall denote this idea by A ⟹ B which is read as "A logically implies B".

## 7.10 ANSWERS TO CHECK YOUR PROGRESS

Ans. to Q. No. 1: a) The variable that occur in the formula are P and Q, so we have to consider 4 possible combinations of truth values of two statements P and Q.

| P | Q | $\sim P$ | $\sim Q$ | $\sim P \wedge \sim Q$ | $\sim(\sim P \wedge \sim Q)$ |
|---|---|---|---|---|---|
| T | T | F | F | F | T |

| | | | | | |
|---|---|---|---|---|---|
| T | F | F | T | F | T |
| F | T | T | F | F | T |
| F | F | T | T | T | F |

Space for learners:

(b) The variable are P and Q,
clearly there are $2^2$ rows in the truth table of this formula.

| P | Q | ~P | ~Q | ~P ∨ Q | ~Q ∨ P | (~P ∨ Q) ∧(~Q ∨ P) |
|---|---|---|---|---|---|---|
| T | T | F | F | T | T | T |
| T | F | F | T | F | T | F |
| F | T | T | F | T | F | F |
| F | F | T | T | T | T | T |

(c)

| P | Q | $P \wedge Q$ | $P \vee Q$ | $(P \wedge Q) \rightarrow (P \vee Q)$ |
|---|---|---|---|---|
| T | T | T | T | T |
| T | F | F | T | T |
| F | T | F | T | T |
| F | F | F | F | T |

(d)

| P | Q | $P \rightarrow Q$ | $Q \wedge (P \rightarrow Q)$ | $\big(Q \wedge (P \rightarrow Q)\big) \rightarrow P$ |
|---|---|---|---|---|
| T | T | T | T | T |
| T | F | F | F | T |
| F | T | T | T | F |
| F | F | T | F | T |

Answer to Q2:

(a)

| $P$ | $Q$ | $\sim P$ | $\sim Q$ | $P \wedge \sim Q$ | $Q \vee (P \wedge \sim Q)$ | $\sim P \wedge \sim Q$ | $Q \vee (P \wedge \sim Q) \vee (\sim P \wedge \sim Q)$ |
|---|---|---|---|---|---|---|---|
| T | T | F | F | F | T | F | T |
| T | F | F | T | T | T | F | T |
| F | T | T | F | F | T | F | T |
| F | F | T | T | F | F | T | T |

All the entries in the last column are T, the given formula is a tautology.

(b)

| P | Q | ~P | ~P ∨ Q | P → Q | (P → Q) ↔(~P ∨ Q) |
|---|---|---|---|---|---|
| T | T | F | T | T | T |
| T | F | F | T | F | T |
| F | T | T | T | T | T |
| F | F | T | F | F | T |

All the entries in the last column are T, the given formula is a tautology.

Similarly, for (c) construct truth tables.

Answer to Q 3:

Truth table for $((\sim P) \vee (\sim Q)) \vee P$

| $P$ | $Q$ | $\sim P$ | $\sim Q$ | $(\sim P) \vee (\sim Q)$ | $((\sim P) \vee (\sim Q)) \vee P$ |
|---|---|---|---|---|---|
| T | T | F | F | F | T |
| T | F | F | T | T | T |
| F | T | T | F | T | T |
| F | F | T | T | T | T |

The last column contains only T.

$\therefore$ $((\sim P) \vee (\sim Q)) \vee P$ is a tautology.

Ans. To Q.No.4: Two propositions are logically equivalent or simply equivalent if they have exactly the same truth values under all circumstances.

***Space for learners:***

Ans. To Q.No.5:(a)

| **P** | **Q** | **~Q** | **$P \rightarrow Q$** | **$\sim(P \rightarrow Q)$** | **$P \wedge \sim Q$** |
|---|---|---|---|---|---|
| T | T | F | T | F | F |
| T | F | T | F | T | T |
| F | T | F | T | F | F |
| F | F | T | T | F | F |

As $\sim(P \rightarrow Q)$ and $P \wedge \sim Q$ have identical truth columns, so $\sim(P \rightarrow Q) \Leftrightarrow P \wedge \sim Q$.

(b)

| **P** | **Q** | **$P \rightarrow Q$** | **$Q \rightarrow P$** | **$P \leftrightarrow Q$** | **$(P \rightarrow Q) \wedge (Q \rightarrow P)$** |
|---|---|---|---|---|---|
| T | T | T | T | T | T |
| T | F | F | T | F | F |
| F | T | T | F | F | F |
| F | F | T | T | T | T |

As $P \leftrightarrow Q$ and $(P \rightarrow Q) \wedge (Q \rightarrow P)$ have identical truth columns,
so, $P \leftrightarrow Q \Leftrightarrow (P \rightarrow Q) \wedge (Q \rightarrow P)$.

(c)

| **P** | **Q** | **~P** | **~Q** | **$P \rightarrow Q$** | **$\sim Q \rightarrow \sim P$** |
|---|---|---|---|---|---|
| T | T | F | F | T | T |
| T | F | F | T | F | F |
| F | T | T | F | T | T |
| F | F | T | T | T | T |

As $P \rightarrow Q$ and $\sim Q \rightarrow \sim P$ have identical truth columns, so $P \rightarrow Q \Leftrightarrow \sim Q \rightarrow \sim P$.

Ans.toQ.No.6:a)WeknowthatP$\rightarrow$Q$\Leftrightarrow$~P$\vee$Q

Similarly, $R \rightarrow Q \Leftrightarrow \sim R \vee Q$

Now,$(P \rightarrow Q) \wedge (R \rightarrow Q) \Leftrightarrow (\sim P \vee Q) \wedge (\sim R \vee Q)$
$\Leftrightarrow (\sim P \wedge \sim R) \vee Q$ (Bydistributivelaw)
$\Leftrightarrow (\sim(P \vee R)) \vee Q$ (ByDeMorgan'slaw)

Space for learners:

$\Leftrightarrow (P \vee R) \rightarrow Q.$

(b) $(P \vee Q) \wedge \sim(\sim P \wedge Q)$

$\Leftrightarrow (P \vee Q) \wedge (\sim\sim P \vee \sim Q)$ [De Morgan's Law]

$\Leftrightarrow (P \vee Q) \wedge (P \vee \sim Q)$ [ Law of Double Negation]

$\Leftrightarrow P \vee (Q \wedge \sim Q)$ [Distributive Law]

$\Leftrightarrow P \vee F$ [Inverse Law]

$\Leftrightarrow P$ [Identity Law]

(c) $(P \rightarrow Q) \rightarrow Q$

$\Leftrightarrow (\sim P \vee Q) \rightarrow Q$ [Conditional as disjunction]

$\Leftrightarrow \sim(\sim P \vee Q) \vee Q$ [Conditional as disjunction]

$\Leftrightarrow (\sim\sim P \wedge \sim Q) \vee Q$ [De Morgan's Law]

$\Leftrightarrow (P \wedge \sim Q) \vee Q$ [ Law of Double Negation]

$\Leftrightarrow (P \vee Q) \wedge (\sim Q \vee Q)$[Distributive Law]

$\Leftrightarrow (P \vee Q) \wedge T$ [$\sim Q \vee Q = T$]

$\Leftrightarrow P \vee Q$ [$\because P \wedge T = P$]

Hence, $(P \rightarrow Q) \rightarrow Q \Leftrightarrow P \vee Q$

Ans. to Q. No.7: a)

| P | Q | $P \rightarrow Q$ | $Q \rightarrow (P \rightarrow Q)$ |
|---|---|---|---|
| T | T | T | T |
| T | F | F | T |
| F | T | T | T |
| F | F | T | T |

Since $Q \rightarrow (P \rightarrow Q)$ is a tautology, Therefore $Q \Longrightarrow P \rightarrow Q$.

(b)

| P | Q | $\sim Q$ | $P \rightarrow Q$ | $\sim (P \rightarrow Q)$ | $\sim (P \rightarrow Q) \rightarrow \sim Q$ |
|---|---|---|---|---|---|
| T | T | F | T | F | T |
| T | F | T | F | T | T |
| F | T | F | T | F | T |
| F | F | T | T | F | T |

Space for learners:

Since ~(P→Q)→ ~Q is a tautology, therefore~(P→Q)⟹ ~Q.

.

Ans. to Q. No.8: a) We have to prove that $P \wedge Q \rightarrow P \vee Q$ is a tautology.

$$P \wedge Q \rightarrow P \vee Q$$

$\Leftrightarrow$ $\sim(P \wedge Q) \vee (P \vee Q)$ [ Conditional as disjunction]

$\Leftrightarrow$ $(\sim P \vee \sim Q) \vee (P \vee Q)$ [ De Morgan's Law]

$\Leftrightarrow$ $(P \vee \sim P) \vee (Q \vee \sim Q)$ [ Associative and Commutative law]

$\Leftrightarrow$ $T \vee T$ [ $P \vee \sim P = T$ ,$Q \vee \sim Q = T$]

$\Leftrightarrow$ $T$ [ Idempotent Law]

Hence, $P \wedge Q \Longrightarrow P \vee Q$

(b) We have to prove that $P \wedge Q \rightarrow (P \rightarrow Q)$is a tautology.

$$P \wedge Q \rightarrow (P \rightarrow Q)$$

$\Leftrightarrow$ $(P \wedge Q) \rightarrow (\sim P \vee Q)$ [ Conditional as disjunction]

$\Leftrightarrow$ $\sim(P \wedge Q) \vee (\sim P \vee Q)$ [ Conditional as disjunction]

$\Leftrightarrow$ $(\sim P \vee \sim Q) \vee (\sim P \vee Q)$ [ De Morgan's Law]

$\Leftrightarrow$ $\sim P \vee (\sim Q \vee Q)$ [ Associative Law]

$\Leftrightarrow$ $\sim P \vee T$[~Q∨ Q = T]

$$\Leftrightarrow T$$

Hence, $P \wedge Q \rightarrow P \vee Q$ is a tautology.

$$\therefore \; P \wedge Q \Longrightarrow P \vee Q$$

## 7.11 POSSIBLE QUESTIONS

Q1. Construct the truth table for each of the following:

(a) (P ∧ Q) → (P ∨Q)

(b) (P ∧ Q) →~P

(c) (P → Q) ∧ (~P ∨ Q)

Q2. With the help of truth tables, prove the following:

(a) (P → Q) ⟺(~P ∨ Q)

(b) (P → Q) ⟺(~Q → ~P)

(c) (P∧Q)⟺(P→Q)∧(Q→P)

**Space for learners:**

Q3. Use the truth table to determine whether the proposition $((\sim P) \vee Q) \vee (P \wedge (\sim Q))$ is a tautology.

Q4. Show the implications without constructing the truth tables.

(a) $\sim (P \rightarrow Q) \Rightarrow P$

(b) $P \wedge (P \rightarrow Q) \Rightarrow Q$

(c) $\sim Q \wedge (P \rightarrow Q) \Rightarrow \sim P$

(d) $(P \vee Q) \wedge (\sim P) \Rightarrow Q$

(e) $(P \rightarrow Q) \rightarrow Q \Rightarrow P \vee Q$

(f) $(P \wedge Q) \Rightarrow P \rightarrow Q$

Q5. Discuss the different types of statements with examples.

Q6. What do you mean by tautology? Explain with example

Q7. What is contradiction? Discuss.

Q8. Give a detailed discussion on logical equivalence.

Q9. What do you mean by tautological implications? Explain.

## 7.12 REFERENCES AND SUGGESTED READINGS


- Rao, G.S. S. Bhishma. *Discrete Structures and Graph Theory*. SciTech Publications (India) PVT Ltd,2003
- Sharma, J.K. *Discrete Mathematics*. Macmillan Publishers India Ltd,2011
- Balaji, G.*Discrete Mathematics*. Balaji Publishers,2015
- Singaravelu, A.&Jeyaraman, M.P. *Discrete Mathematics*. Meenakshi Agency,2016
- Satyanarayana, B.&Prasad, K.S. *Discrete Mathematics and Graph Theory*. PHI Learning Private Limited,2009
- Rao, G.S. *Discrete Mathematical Structure*. New Age international (P) Limited,2009
- Singh, Y.N. *Mathematical Foundation of Computer Science*, New Age international Publishers, 2005
- Lipschutz, S&Lipson, M.L.Schaum's outline of*Theory and Problems of Discrete Mathematics*, McGraw-Hill,2007


**Space for learners:**

- Chandrasekaran, N & Umaparvathi, M. *Discrete Mathematics*, PHI Learning Private Limited,201
- Tremblay, J.P. & Manohar, *Discrete Mathematical Structures with Applications to Computer Science,* Tata McGraw Hill Education Private Limited,1997


***Space for learners:***

# UNIT 8: PREDICATED CALCULUS

Space for learners:

**Unit Structure:**



## 8.1 INTRODUCTION

In this unit, we shall discuss about simple statements and their validity through predicates and quantifiers. Also, we apply predicate formulas to determine the truth or falsity of the statements. The soundness, completeness and compactness of the statements are also discussed in this unit.

## 8.2 UNIT OBJECTIVES

After going through this unit, you will be able to

- understand the logic of a computer program

- develop write programs in various computer languages.
- get ideas behind mathematical logic and inference theory
- understand mathematical logic associated with various reasoning and mathematical proofs
- understand predicates, quantifiers, free and bound variables
- know the inference theory of predicate calculus.

Space for learners:

## 8.3 PREDICATES

Let us consider a mathematical relation $x > 10$.

The Statement "$x$ is greater than 10" has two parts. The first part, the variable $x$, is the subject of the statement. The second part, "is greater than 10" which refers to a property that the subject can have, is called the predicate.

We can denote the statement "$x$ is greater than 10" by the notation $P(x)$, where $P$ denotes the predicate "isgreaterthan10"and $x$ is the variable.

$P(x)$ is called propositional function of $x$.

Once a value has been assigned to the variable $x$, the statement $P(x)$ becomes a proposition and has a truth value.

## 8.4 QUANTIFIERS

Many mathematical statements assert that a property is true for all values of a variable or for some values of the variable, in a particular domain, called the universe of discourse. Mostly it is denoted by $D$.

The universal quantification of $P(x)$ is the statement:

$P(x)$ is true for all values of $x$ in the universe of discourse" and is denoted by the notation,

$$(x)P(x) \text{ or } \forall x P(x).$$

Theproposition$(x)P(x)$or$\forall xP(x)$isreadas"forall$x$,$P(x)$"or"forevery$x$,$P(x)$".The symbol$\forall$iscalledtheuniversalquantifier.

Theexistentialquantificationof$P(x)$istheproposition.

Thereexistsatleastone$x$(oran$x$)suchthat$P(x)$istrue"andisdenotedbythenotati on

$$\exists x P(x).$$

The symbol $\exists$ is called the existential quantifier.

**Example 1**

Express the following statement using quantifiers:

"Every Computer SciencestudentneedsacourseinMathematics"

**Solution.**

Let$D$={Students in Computer Science}($D$isuniverseofdiscourseordomain).

Let$P(x)$:$x$needsacourseinMathematics.

Wecanrewritetheaboveexpressionas"Forall$x$,$x$needsacourseinMathe matics".

Then$\forall xP(x)$.

**Example 2**

Express the following statement using quantifiers:

"Every Computer SciencestudentneedsacourseinMathematics".

**Solution.**

Let$D$={Students}

Let$P(x)$:$x$is Computer Sciencestudent.

$Q(x)$:$x$needs a course inMathematics.

Wecanrewritetheaboveexpressionas

"Forall$x$,if$x$isa Computer Science student then $x$ needs a course inMathematics".

Space for learners:

$$\text{Then} \forall x[P(x) \rightarrow Q(x)].$$

Space for learners:

**Example 3**

Express the following statement using quantifiers:

"Thereisastudentinthisclass,whoownsapersonalcomputer".

**Solution.**

Let$D$={Students inthisclass}.

Let $P(x)$:$x$owns a personal computer.

Wecanrewritetheaboveexpressionas:

"Thereexistsan$x$,suchthat,$x$ownsapersonalcomputer".

$$\text{Then} \exists x P(x).$$

**Example 4**

Express the following statement using quantifiers:

"EveryonewhoknowshowtowriteprogramsinJAVAcangetahighpayingjob".

**Solution.**

Let$D$={Students inthisclass}.

$P(x)$:$x$knowshowtowriteprogramsinJAVA.

$Q(x)$:$x$getsanhighpayingjob.

We can rewrite the above expression as

"For every $x$, if $x$knowshowtowriteprograms inJAVAthenhegetsahighpayingjob".

$$\text{Then} \forall x[P(x) \rightarrow Q(x)].$$

**Example 5**

Express the following statement using quantifiers:

"Someonewhopassedthefirstexaminationhasnotreadthebook".

**Solution**.

Let$D$={Students inthisclass}.

Let $P(x)$: $x$ has passed the first examination.

$Q(x)$:$x$hasnotreadthebook.

Wecanrewritetheaboveexpressionas“Thereexistsan$x$,such that $x$ has passed the first examination and $x$ has notreadthebook”.

Then$\exists x[P(x) \wedge Q(x)]$.

### 8.4.1 Negation of a Quantified Expression

**Example 6**

Findthenegationoffollowingexpression:

“Everystudentintheclasshasstudiedcomputer programming”.

**Solution**.

Let$D$={Students ina Class}.

Let$P(x)$:$x$hasstudiedcomputerprogramming.

Thenthegivenexpressionis$\forall x P(x)$.

Tofindthenegationof$\forall x P(x)$:

Negation of the above expression is “It is not the case that,everystudentintheclasshasstudiedcomputerprogramming”. Hence it is represented as $\neg[\forall x P(x)]$.

It also meansthat,“thereisastudentintheclasswhohasnotstudiedcomputerprogramming”,

i.e.,thereisastudent$x$intheclass,suchthat,$x$hasnotstudiedcomputerprogramming.

Henceitisrepresentedas$\exists x[\neg P(x)]$.

$\therefore \neg[\forall x P(x)] \equiv \exists x[\neg P(x)]$.

**Example 7**

Findthenegationoffollowingexpression:

“Thereisastudentintheclasswhohasstudiedcomputerprogramming”.

**Solution**.

Space for learners:

Let $P(x)$: $x$ has studied computer programming.

Let$D$={Students inaClass} ($D$isuniverseofdiscourse ordomain).

Thenthegivenexpressionis$\exists x P(x)$.

Tofindthenegationof$\exists x P(x)$:

Negationoftheaboveexpressionis“Itisnotthecasethat,there is a student in the class who has studied computerprogramming”. Hence it is represented as $\neg[\exists x P(x)]$.

It also meansthat,“Nostudentisfoundintheclass,whohavestudiedcomputerprogramming”,

i.e.,“Everystudentintheclasshasnotstudiedcomputer programming”,

i.e.,“Forevery$x$intheclass,$x$hasnotstudiedcomputer programming”.

Henceitisrepresentedas$\forall x[\neg P(x)]$.

$\therefore \neg[\exists x P(x)] \equiv \forall x[\neg P(x)]$.

## 8.5 Predicate Formulas

We denote by $P(x_1, x_2, ..., x_n)$, an $n$-place predicate formula in which the letter $P$ is an $n$-place predicate and $x_1$, $x_2$,…,$x_n$ are individual variables. In general, $P(x_1, x_2, ..., x_n)$ is called an **atomic formula** of predicate calculus.

The following are some examples of atomic formulas.

$A(x)$, $B(x, y)$ and $C(x, d, z)$.

A **well-formed formula** (*wf.*) of predicate calculus is defined by

(i) Every atomic formula is a well-formed formula.

(ii) If $A$ is a well-formed formula, so is $\neg A$.

Space for learners:

(iii) If $A$ and $B$ are well-formed formulas, so are $(A \vee B)$, $(A \wedge B)$, $(A \rightarrow B)$ and $(A € \quad B)$.

(iv) If $A$ is a well-formed formula and $x$ is any variable, so are $(x)A$ and $(\exists x)A$.

(v) Only the formulas obtained by applying rules (i)-(iv) are well-formed formulas.

Space for learners:

**Check Your Progress**

1. A property that the subject can have, is called the ________________
2. The symbol _______ is called the existential quantifier.
3. The symbol _______ is called the universal quantifier.
4. The negation of a statement is denoted by the symbol _______.

## 8.6 Free and Bound Variables

Consider the following statement:

All students are intelligent. This can be written in symbolic form as

$$(x)(S(x) \rightarrow I(x)),$$

where $S(x): x$ is a student and $I(x): x$ is intelligent.

In the above statement, if we restrict the class as the class of students, then the symbolic representation will be $(x)I(x)$. Such a restricted class is also called "Universe of Discourse".

- In any symbolic formula, the part containing $(x)A(x)$ or $\exists xA(x)$, such part is called the "$x$-bound" part of the formula.
- Any variable appearing in an "$x$- bound" part of the formula is called as a bound variable.

- Otherwise it is called as a free variable.
- Any formula immediately following $(x)$ or $(\exists x)$ is called the scope of the quantifier.

**Example 8**

Consider the symbolic form of a statement: $(y)A(y) \vee B(y)$.

In this notation, all $y$ in $A(y)$ is bound whereas the $y$ in $B(y)$ is free. The scope of $(y)$ is $A(y)$.

## 8.7 Inference Theory of Predicate Calculus

**Rules of Inference**

1. **Rule P**: A premise can be introduced at any point of derivation.
2. **Rule T**: A formula can be introduced provided it is tautologically implied by previously introduced formulas in the derivation.
3. **Rule CP**: If $S$ can be derived from $R$ and a set of premises, then$R \rightarrow S$ can be derived from the set of premises alone.

**Rule US[Universal Specification]**

It is the rule of inference, which states that one can conclude that $A(k)$ is true, if $\forall y \ A(y)$ is true, where $'k'$ is an arbitrary member of the universe of discourse .This rule is also called the Universal Instantiation.

In other words, Universal Specification is the rule of inference which says we can conclude $A(k)$ is true for a particular element $k$ of the universe of discourse if $\forall y \ A(y)$ is true. This $k$ can be chose arbitrarily.

For example, if $y^2 > 0, \ \forall y \neq 0,$ then $3^2 > 0,$ for the particular value $3.$ It is true for any $k \neq 0, \ k^2 > 0.$

Space for learners:

**Rule ES[ExistentialSpecification]**

Itistherulewhichallowsustoconcludethat $A(k)$ istrue,if $\exists y \quad A(y)$ istrue,where $'k'$isnotanarbitrarymemberof theuniverse, butone for which $A(k)$ istrue.Usuallywe will notknow,what$'k'$ is,butknowthatitexists.Sinceitexists,wemaycallit $'k'$. ThisruleisalsocalledtheExistentialInstantiation.

In other words, Existential Specification is the rule of inference which says that there is an element $k$ in the universe of discourse for which $A(k)$ is true if $\exists y \quad A(y)$ is true. Here $'k'$ is not arbitrary, but it is specific. In practice, we may not know what $'k'$ is, but it exists. Since it exists, we give a name $'k'$ and proceed with our argument.

**RuleUG[UniversalGeneralization]**

Itistherulewhichstatesthat $\forall y \quad A(y)$ istrue,if $A(k)$ istrue,where$'k'$ isanarbitrary member (notaspecificmember)oftheuniverse ofdiscourse.

In other words, Universal Generalization is the rule of inference which says that $\forall y \quad A(y)$ is true if $A(k)$ is true for an arbitrary element $'k'$ of the universe of discourse. This rule is used when we need to prove $\forall y \quad A(y)$ is true.

**RuleEG[ExistentialGeneralization]**

Itistherulethatisusedtoconcludethat, $\exists y \quad A(y)$ when $A(k)$ istrue,where $'k'$isaparticularmemberoftheuniverseofdiscourse.

In other words, Existential Generalization is the rule of inference which says that for particular element $'k'$ of the universe of discourse if $A(k)$ is true, then $\exists y \quad A(y)$ is true.

We summarize the above rules in the following table.

| Rule | Inference |
|---|---|
| US | $\forall y A(y)$ |

Space for learners:

| | |
|---|---|
| | $\therefore A(k)$ for an arbitrary $k$ |
| ES | $\exists y \ \ A(y)$<br>$\therefore A(k)$ for a particular $k$ |
| UG | $A(k)$ for an arbitrary $k$<br>$\therefore \forall y \ \ A(y)$ |
| EG | $A(k)$ for some $k$<br>$\therefore \exists y \ \ A(y)$ |

Space for learners:

**Remark:**

We have seen rules of inference for proposition and rules of inference for quantified propositions. Sometimes, we have to use a combination of the above rules. Two such combinations of rules of inference quite often used are the Universal Modus Ponens and Universal Modus Tollens.

- **Universal Modus Ponens (MP)** says that $\forall y,$ if $A(y)$ is true then $B(y)$ is true and if $A(k)$ is true for a particular element $'k'$ in the universe of discourse then $B(k)$ must also be true.
  Thus,

  $$\forall y \ \ (A(y) \to B(y))$$

  $$A(k)$$

  $$B(k),$$

  where $'k'$ is particular element in the domain.

- **Universal Modus Tollens (MT)**says that $\forall y,$ if $A(y)$ then $B(y)$ and if for a particular element $'k'$ in the universe of discourse $\neg B(k)$ is true then $\neg A(k)$ is true.
  Thus,

  $$\forall y \ (A(y) \to B(y))$$

  $$\neg B(k)$$

  $$\neg A(k)$$

  for a particular $'k'$.

- **Universal Transitivity (UT)** says that if $\forall y(A(y) \to B(y))$ and $\forall y(B(y) \to C(y))$ are true, then $\forall y(A(y) \to C(y))$ is true, where the domains of all the quantifiers are the same.

Space for learners:

**Example 8**

Letusverifytheargumentbyinferencetheory.

Allmenaremortal.

Socratesisaman.

Therefore, Socrates is mortal.

**Solution.**

Let$D$={Humanbeing}

Let$P(x)$: $x$ isaman;$Q(x)$: $x$ ismortaland $s$ :Socrates.

Aboveproblembecomes$\forall x[P(x) \to Q(x)], P(s) \Rightarrow Q(s)$

| S.No | Statement | Reason |
|---|---|---|
| 1 | $\forall x[P(x) \to Q(x)]$ | RuleP (Given Premise) |
| 2 | $P(s) \to Q(s)$ | RuleUS, 1 |
| 3 | $P(s)$ | RuleP (Given Premise) |
| 4 | $Q(s)$ | MP, 2, 3 |

**Solved Problems**

1.

Provetheimplication:$\forall x(P(x) \to Q(x)), \forall x(R(x) \to \neg Q(x)) \Rightarrow \forall x(R(x) \to \neg P(x))$.

**Solution.**

Given premisesare$\forall x(P(x)\rightarrow Q(x)),\forall x(R(x)\rightarrow\neg Q(x))$.

Conclusionis$\forall x(R(x)\rightarrow\neg P(x))$.

**Space for learners:**

| S.No | Statement | Reason |
|---|---|---|
| 1 | $\forall x(P(x)\rightarrow Q(x))$ | RuleP |
| 2 | $P(a)\rightarrow Q(a)$ | RuleUS,1,forall$a$ |
| 3 | $\forall x(R(x)\rightarrow\neg Q(x))$ | RuleP |
| 4 | $R(a)\rightarrow\neg Q(a)$ | RuleUS,3,forall$a$ |
| 5 | $\neg Q(a)\rightarrow\neg P(a)$ | RuleT,2,contrapositive,forall$a$ |
| 6 | $R(a)\rightarrow\neg P(a)$ | RuleT,4,5, for all$a$ |
| 7 | $\forall x(R(x)\rightarrow\neg P(x))$ | RuleUG,6 |

2. Provethat$\forall x(P(x)\rightarrow(Q(y)\wedge R(x))),\exists xP(x)\Rightarrow Q(y)\wedge\exists x(P(x)\wedge R(x))$.

**Solution.**

Premisesare$\forall x(P(x)\rightarrow(Q(y)\wedge R(x))),\exists xP(x)$.

Conclusionis$Q(y)\wedge\exists x(P(x)\wedge R(x))$.

| S.No | Statement | Reason |
|---|---|---|
| 1 | $\forall x(P(x)\rightarrow(Q(y)\wedge R(x)))$ | RuleP |
| 2 | $P(a)\rightarrow(Q(y)\wedge R(a))$ | RuleUS,1,forall$a$Rul |
| 3 | $\exists xP(x)$ | eP |
| 4 | $P(a)$ | RuleES,3,forsome $a$ |
| 5 | $Q(y)\wedge R(a)$ | MP,2,4,forsome$a$ |
| 6 | $Q(y)$ | RuleT,5 |
| 7 | $R(a)$ | RuleT,5,forsome$a$ |
| 8 | $P(a)\wedge R(a)$ | Rule T, 4,7, for some |
| 9 | $\exists x(P(x)\wedge R(x))$ | $a$RuleEG,8 |

| | | | |
|---|---|---|---|
| 10 | $Q(y) \wedge \exists x(P(x) \wedge R(x))$ | RuleT,6,9 | |

**Space for learners:**

3. Showby indirect methodofproof,that$\forall x(P(x) \vee Q(x)) \Rightarrow (\forall x P(x)) \vee (\exists x Q(x))$.

**Solution**.

Byindirectmethod,letusassumethat$\neg[(\forall x P(x)) \vee (\exists x Q(x))]$asanadditionalpremiseandarrive ata contradiction.

| S.No | Statement | Reason |
|---|---|---|
| 1 | $\neg[(\forall x P(x)) \vee (\exists x Q(x))]$ | RuleP(additionalpremise) |
| 2 | $\neg(\forall x P(x)) \wedge \neg(\exists x Q(x))$ | RuleT,DeMorgan'slaw,1 |
| 3 | $\neg(\forall x P(x))$ | RuleT,2 |
| 4 | $\neg(\exists x Q(x))$ | RuleT,2 |
| 5 | $\exists x \neg P(x)$ | RuleT,3 |
| 6 | $\forall x \neg Q(x)$ | RuleT,4 |
| 7 | $\neg P(a)$ | Rule ES, 5, for some |
| 8 | $\neg Q(a)$ | *a*Rule US, 6, for all *a* |
| 9 | $\neg P(a) \wedge \neg Q(a)$ | RuleT,7,8,forsome*a* |
| 10 | $\neg(P(a) \vee Q(a))$ | RuleT,DeMorgan'slaw,9,forsome*a* |
| 11 | $\forall x(P(x) \vee Q(x))$ | RuleP |
| 12 | $P(a) \vee Q(a)$ | Rule US, 11, for all *a* |
| 13 | $\neg(P(a) \vee Q(a)) \wedge (P(a) \vee Q(a))$ | RuleT,10,12,forsome*a* |
| 14 | $F$ | RuleT,13 |

4. Provetheimplication$\forall x(P(x) \to Q(x)), \forall x(R(x) \to \neg Q(x)) \Rightarrow \forall x(R(x) \to \neg P(x))$.

**Solution**.

Premisesare$\forall x(P(x) \to Q(x))$and$\forall x(R(x) \to \neg Q(x))$.

Conclusionis$\forall x(R(x)\rightarrow\neg P(x))$.

Space for learners:

| S.No | Statement | Reason |
|---|---|---|
| 1 | $\forall x(P(x)\rightarrow Q(x))$ | RuleP |
| 2 | $P(a)\rightarrow Q(a)$ | RuleUS,1,forall$a$ |
| 3 | $\forall x(R(x)\rightarrow\neg Q(x))$ | RuleP |
| 4 | $R(a)\rightarrow\neg Q(a)$ | RuleUS,3,forall$a$ |
| 5 | $Q(a)\rightarrow\neg R(a)$ | RuleT,Contrapositive,4, forall$a$ |
| 6 | $P(a)\rightarrow\neg R(a)$ | Rule T, 2, 5, for all $a$ |
| 7 | $R(a)\rightarrow\neg P(a)$ | RuleT,Contrapositive,6, forall$a$ |
| 8 | $\forall x(R(x)\rightarrow\neg P(x))$ | RuleUG,7 |

5. Show that the premises "One student in this class knows howtowriteprogramsinJAVA"and"Everyonewhoknowshowtowrite programsinJAVAcangetahigh-payingjob"implytheconclusion "Someone in this class can get a high-paying job".

**Solution**.

Let$D=\{\text{Student}\}$.

Let$C(x)$:$x$isinthisclass.

$J(x)$:$x$knowsJAVAprogramming.

$H(x)$:$x$ cangetahigh-payingjob.

Thenpremisesare$\exists x(C(x)\wedge J(x))$and$\forall x(J(x)\rightarrow H(x))$.

Conclusionis$\exists x(C(x)\wedge H(x))$.

| S.No | Statement | Reason |
|---|---|---|

| | | |
|---|---|---|
| 1 | $\exists x(C(x)\wedge J(x))$ | RuleP |
| 2 | $C(a)\wedge J(a)$ | RuleES,1,forsome$a$Ru |
| 3 | $C(a)$ | le T, 2, for some |
| 4 | $J(a)$ | $a$Rule T, 2, for some |
| 5 | $\forall x(J(x)\rightarrow H(x))$ | $a$RuleP |
| 6 | $J(a)\rightarrow H(a)$ | RuleUS,5,forall$a$ |
| 7 | $H(a)$ | Rule T, MP,4,6,forsome$a$ |
| 8 | $C(a)\wedge H(a)$ | Rule T, 3, 7, for some $a$ |
| 9 | $\exists x(C(x)\wedge H(x))$ | RuleEG,8 |

***Space for learners:***

6. Show that the premises "A student in this class has not readthe book" and "Everyone in this class passed the firstexamination"implytheconclusion"Someonewhopassedthefirstexa minationhasnotreadthebook".

**Solution.**

Let$D$={Student}.

Let$C(x)$:$x$isinthisclass.

$R(x)$:$x$hasnotreadthebook.

$F(x)$:$x$haspassedthefirstexamination.

Thenthepremisesare$\exists x(C(x)\wedge R(x)),\forall x(C(x)\rightarrow F(x))$.

Conclusionis$\exists x(F(x)\wedge R(x))$.

| S.No | Statement | Reason |
|---|---|---|
| 1 | $\exists x(C(x)\wedge R(x))$ | RuleP |
| 2 | $C(a)\wedge R(a)$ | RuleES,1,forsome$a$Ru |
| 3 | $C(a)$ | le T, 2, for some |
| 4 | $R(a)$ | $a$Rule T, 2, for some |
| 5 | $\forall x(C(x)\rightarrow F(x))$ | $a$RuleP |
| 6 | | RuleUS,5,forall$a$ |

| | | |
|---|---|---|
| 7 | $C(a) \rightarrow F(a)$ | RuleT,MP,3,6,forsome$a$Rul |
| 8 | $F(a)$ | e T, 4, 7, for some |
| 9 | $F(a) \wedge R(a)$ | $a$RuleEG,8 |
| | $\exists x(F(x) \wedge R(x))$ | |

***Space for learners:***

## 8.8 Validity

In the practical life, the validity of a statement made by a person is important. Suppose a person makes a validity of a statement which may be true depending on the nature of the statement. For example, if the statement is "Daily it is raining or it is raining on some days".

A predicate formula is said to have validity if every assignment in every structure satisfies it.

**Examples**

1. $\exists y\, P \rightarrow \exists \forall y \neg P$
2. $\forall y\, P \rightarrow \neg \exists y \neg P$
3. $\exists y(P \vee Q) \leftrightarrow \exists y\, P \vee \exists y\, Q$
4. $\forall y\,(P \wedge Q) \leftrightarrow \forall y\, P \wedge \forall y\, Q$
5. $\forall y\; y = y$

## 8.9. Soundness, completeness and compactness

There are distinct concepts of "truth" (⊨) and "provability" (⊢). We'd like them to be the same, in the sense that we should only be able to prove things that are true, and if they are true, we should be able to prove them. These two properties are known as **soundness** and **completeness**.

A proof system is **sound** if everything that is provable is true. In other words, if $A_1,\ldots,A_n \vdash S$ then $A_1,\ldots,A_n \models S$.

A proof system is **complete** if everything that is true has a proof. In other words, if $A_1,\ldots,A_n \vDash S$ then $A_1,\ldots,A_n \vdash S$.

A set $W$ of well-formed formulasis called satisfiable if and only if there is a truth assignment that satisfies every member of $W$.

**CHECK YOUR PROGRESS**

5. In the statement $(x)(S(x) \rightarrow I(x))$, _______ is free and ______ is bound variables.
6. A proof system is _______if everything that is provable is true.
7. A proof system is _________if everything that is true has a proof.
8. A set of well-formed formulas is ____________if and only if every finite subset is satisfiable (**Compactness Theorem**).

Space for learners:

## 8.10 SUMMING UP

- Propositions have truth values.
- A property is true for all values of a variable or for some values of the variable, in a particular domain. It is called the universe of discourse.
- Conjunction, disjunction or negation operations can be applied on propositions.
- Every atomic formula is a well-formed formula.
- Any variable appearing in an "$x$ - bound" part of the formula is called as a bound variable. Otherwise, it is called as a free variable.
- Any formula immediately following $(x)$ or $(\exists x)$ is called the scope of the quantifier.
- A predicate formula is said to have validity if every assignment in every structure satisfies it.
- A proof system is **sound** if everything that is provable is true. A proof system is **complete** if everything that is true has a proof.

- A set $W$ of well-formed formulas is called satisfiable if and only if there is a truth assignment that satisfies every member of $W$.

Space for learners:

## 8.11 ANSWERS TO CHECK YOUR PROGRESS

1. Predicate
2. $\exists$
3. $\forall$
4. $\neg$
5. *I(x), S(x)*
6. Sound
7. Complete
8. Satisfiable

## 8.12 POSSIBLE QUESTIONS

1. Express the following statements using predicates and both quantifiers.

   (i) All men are mortal.

   (ii) Every apple is red.

   (iii) All birds can fly.

   (iv) There is an integer which is odd and prime.

   (v) Every student of this class visited either Mumbai or New Delhi".

2. For the following statements, write the symbolic form using predicates and quantifiers, and then their negation forms.

   (i) Everybody who is healthy can do all types of works.

   (ii) Some people are not admired by everyone.

   (iii) Everyone should help his neighbours or his neighbours will not help him.

3. Show that the premises "Everyone in the Computer Science branch has studied Discrete Mathematics" and "John is in Computer Science branch" imply the conclusion "John has studied Discrete Mathematics".

4. Verify the validity of the following statement.

   Every living thing is a plant or an animal. John's gold fish is alive and it is not a plant. All animals have hearts. Therefore John's gold fish has a heart.

5. Find the free and bound variables in the following:

   (i) $\forall y\ (A(y) \wedge B(y)) \rightarrow \forall y\ (A(y)) \wedge C(y)$.

   (ii) $(\forall y\, A(y)\ \text{€}\quad B(y) \wedge \exists y\ C(y)) \wedge D(y)$.

***Space for learners:***

## 8.13 REFERENCES AND SUGGESTED READINGS


- Bernad Kolman, Robert C. Busby and Sharon Cutler Ross, *Discrete Mathematical Structures*, Pearson, USA. 2014.
- Kenneth Rosen, *Discrete Mathematics and Its Applications*, McGraw-Hill Higher Education, New York, 2019.
- Senthil Kumar B. V. and Hemen Dutta, *Discrete Mathematical Structures: A Succinct Foundation*, CRC Press, Boca Raton FL, 2020.
- Seymour Lipschutz and Marc Lipson, *Schaums Outline of Discrete Mathematics*, MCGraw-Hill, New York, 2007.
- G. Shankar Rao, *Discrete Mathematics Structures*, New Age International Pvt. Ltd., India, 2007.
- Sriraman Sridharan and R. Balakrishnan, *Discrete Mathematics: Graph Algorithms*, Algebraic Structures, Coding Theory and Cryptography, CRC Press, Boca Raton, FL, 2020.
- Susanna S. Epp, *Discrete Mathematics with Applications*, Cengage Learning, Boston USA, 2019.
- J. P. Tremblay and R. Manohar, *Discrete Mathematical Structures with Applications to Computer Science*, Tata

McGraw-Hill, New Delhi, India, 2008.

- Iqbal H. Jebril, Hemen Dutta and Ilwoo Cho, *Concise Introduction to Logic and Set Theory*, CRC Press, Boca Raton FL, 2021.

**Space for learners:**

# BLOCK II:
# GRAPH THEORY

# UNIT 1: INTRODUCTION TO GRAPH

Space for learners:

**Unit Structure:**



## 1.1 INTRODUCTION

In this unit, you will learn the fundamental aspects of graph theory. You will also learn about finite and infinite graph, directed and undirected graphs, incidence and degree, isolated and pendant vertices, null graph. You will also learn the history of graph theory in this unit. Graph theory is an area of mathematics which is realistic in its nature. The purpose of graph theory is to solve day to day problems of human beings. In this unit, you will also learn several properties related to finite and infinite graphs. This unit tries to simplify the ideas related to directed and undirected graphs. Incidence and degree of a vertex are two of the main building blocks of graph theory. You will learn some fundamental properties related to degree. Moreover, various examples will be discussed in this unit. These examples will help your knowledge to grow. Applications of graph theory can be found in various areas of mathematics, computer science, biology, theoretical chemistry, social networks, etc. since the scopes of applications are limited in this unit, thus we will skip applications and we will mainly focus on theoretical foundations only.

## 1.2 UNIT OBJECTIVES

After going through this unit, you will be able to:

- know the history of development of graph theory in concise manner
- understand the fundamental concepts and notions of graph theory
- define graph and its different types viz. finite and infinite graphs, directed and undirected graphs. Incidence and degree, isolated and pendant vertices, null graphs etc. will be discussed.
- solve problems related to above graphs.

## 1.3 BRIEF HISTORY ON DEVELOPMENT OF GRAPH THEORY

The subject "graph theory" was initiated from a real problem. The problem is known as "Konigsberg bridge problem". It was one of the unsolved problems of $18^{th}$ century. But, mathematician Leonhard Euler (1707-1782) solved this famous problem in 1736. The problem is discussed below.

***Space for learners:***

ace for learners:

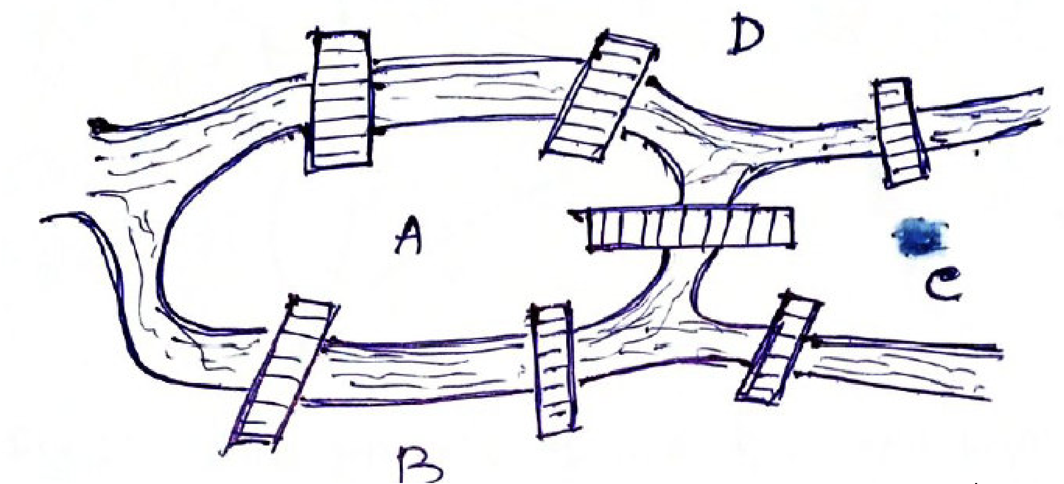



Figure 1: Konigsberg bridge problem

Two islands, A and C, of the Pregel River in Konigsberg were linked to each other and to the banks, B and D, of the Pregel River by seven bridges as shown in Fig1. The problem was to start at any of the four land areas A,B,C or D, walk across each of the seven bridges exactly once, and return to the starting point. Euler proved that there is no solution of the Konigsberg bridge problem. To give proof, Euler simplified the problem. He represented each land area by a point and each bridge by a line joining the corresponding points. Thus, this simplified representation produces a graph. Euler's representation of the Konigsberg bridge problem is shown in Fig2.

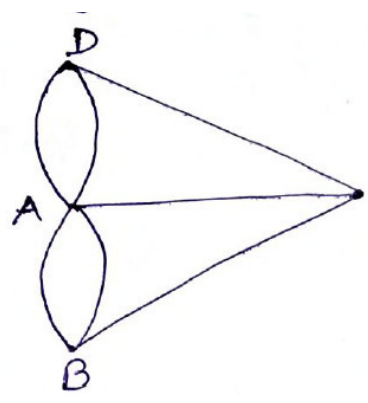
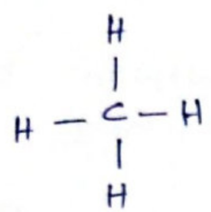



Figure 2: The graph of the Konigsberg bridge problem

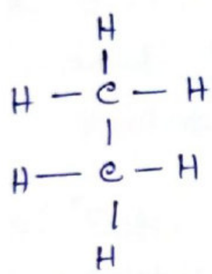
Space for learners:

Since the development of graph theory by Euler, it has been using in many areas. Physicist Gustav Kirchhoff used theory of tree in 1845 to solve the system of simultaneous linear equations representing the current in each branch and around each circuit of an electric network. Kirchhoff used simple representations of the electric networks of the circuits using only points and lines without indicating electrical elements of the circuits. Similarly Caylay discovered trees and his ideas of trees were applied to enumerate the isomers of the saturated hydrocarbons $C_nH_{2n+2}$, where n represents the number of carbons atoms. Some of the saturated hydrocarbons are given below.

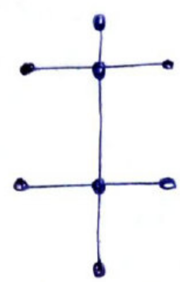


Methane ($CH_4$)

Graphical representation of methane



Ethane ($C_2H_6$)



Graphical representation of ethane

*ce for learners:*

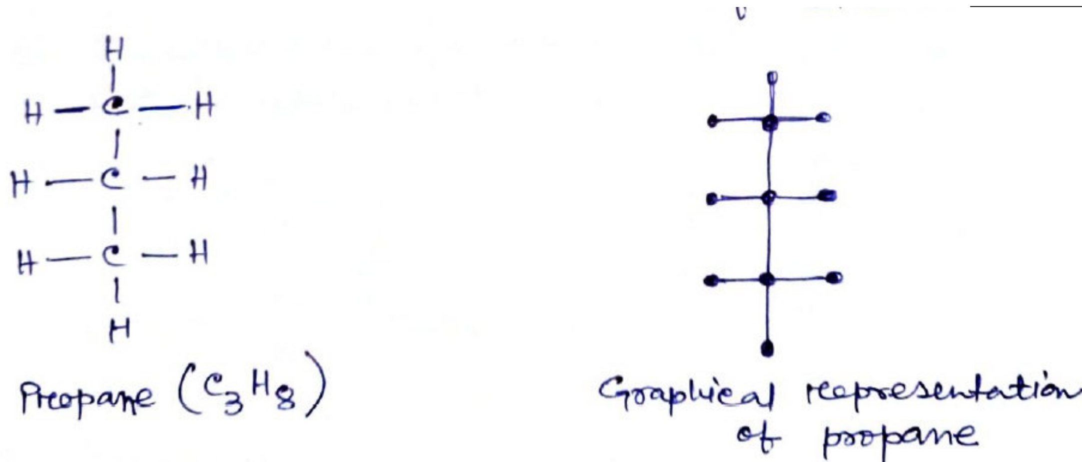



Figure 3: Some saturated hydrocarbons and their graphical representations.

These are some of the fields where graph theory has been applied. But, the reach of graph theory to any area of science and social science can be found easily. Computer networking, an area of computer science, has its foundation based on graph theory.

**CHECK YOUR PROGRESS**

1. Graph theory was initiated from a real problem entitled ……………………… .

2. ………………. Solved Konigsberg bridge problem in ………………. .

3. There were ……………. Islands, ……………... banks and ……………. Bridges in Konigsberg bridge problem.

4. “Euler proved that Konigsberg bridge problem has no solution” …………..is the statement true?

## 1.4 BASIC CONCEPTS

In this section, we study the basic concepts of graph theory. It is important to note that graph theory does not require any other sophisticated area of mathematics other than basic set theory to introduce fundamental notions and definitions of graph theory. Thus, it is expected that our readers are familiar to basic set theory.

Space for learners:

### 1.4.1 Definition of A Graph

A graph (or simply a linear graph) G=(V,E) consists of a set of points $V=\{v_1, v_2, v_3, \ldots\}$ and a set $E=\{e_1, e_2, e_3, \ldots\}$ of unordered pairs of points of V.

The points $v_1, v_2, v_3, \ldots$ are called vertices and $e_1, e_2, e_3, \ldots$ are called edges. Several books use the words 'nodes' in lieu of vertices and 'lines' in lieu of edges.

Each edges $e_k$ is represented by an unordered pair $(v_i, v_j)$. It is also denoted as $\{v_i, v_j\}$. Thus, $(v_i, v_j)$ and $\{v_i, v_j\}$ are not different in graph theory. $e_k = (v_i, v_j)$ is said to join vertex $v_i$ and vertex $v_j$. We write $e_k = v_iv_j$ and say that $e_i$ and $e_j$ are adjacent points. It is sometimes denoted as $e_i adj e_j$. The points $v_i$ and line $e_k$ are incident with each other. Similarly, $v_j$ and the line $e_k$ are incident with each other. Two lines $e_k$ and $e_m$ are adjacent lines if they are incident with a common point.

Example1. If G=(V,E) be a graph, where V={x,y,z,p,q} and E={(x,y), (y,z), (x,q), (x,p), (y,p)}, then the graph G=(V,E) can be represented as given below.

Space for learners:

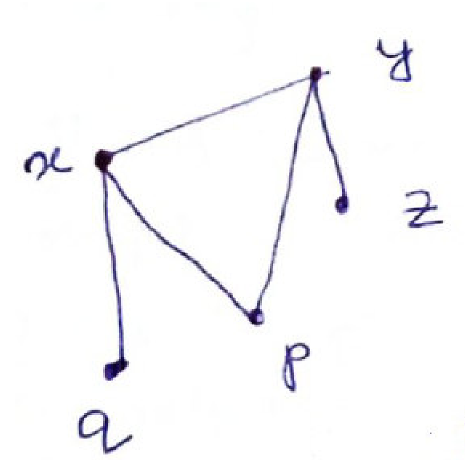



**Figure 4: Graph of G=(V,E)**

If we represent $e_1=(x,y)$, $e_2=(y,z)$, $e_3=(x,q)$, $e_4=(x,p)$ and $e_5=(y,p)$, then the graph can be represented as given below

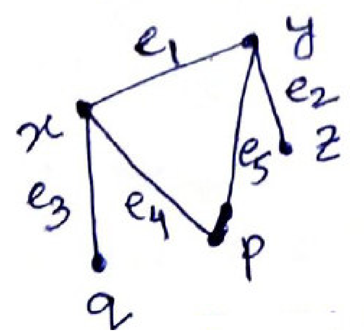



**Figure 5: Graph of G = (V, E)**

## Check your progress

5. A graph G=(V,E) consist of ……… and ……………

6. The elements of the set V of a graph G=(V,E) are called………. .

7. The elements of the set E of a graph G=(V,E) are called ……….. .

### 1.4.2 Basic Terminologies

In this section, we study basic terminologies of graph theory.

Definition1: A graph with p-vertics and q- edges is called a (p,q) graph.

Example 2 : A (3,3) graph is represented as given below

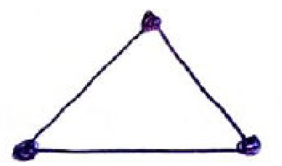


Figure 6: (3,3)-graph

Definition2 : The (1,0)-graph is called trivial graph

Example 3 :The trivial graph is shown below

Definition3 : In graph G=(V,E); an edge to be associated with a vertex pair ((()))) is permissible. Such an edge is called a loop (or self-loop).

Example 4 :Let G=(V,E) be a graph, where V={x,y,z} and E= {(x,x), (x,y), (x,z), (y,z), (z,z)}.

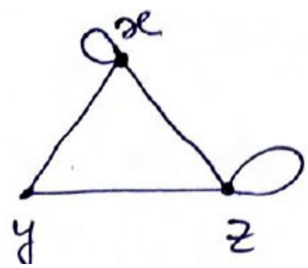



Figure 7: Graph of G=(V,E).

If we denote $e_1=(x_1,x)$, $e_2=(x,y)$, $e_3=(x,z)$, $e_4=(y,z)$ and $e_5=(z,z)$, then figure 7 can be re-drawn as below.

***Space for learners:***

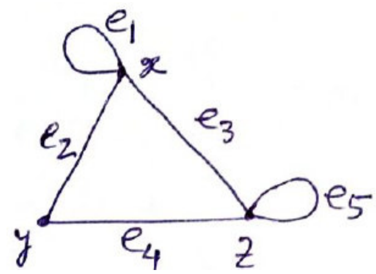



Figure 8: Graph of G=(V,E) with distinct edges.

Defination 4: In a multigraph, no loops are allowed but more than one edge can join two vertices. These edges are called multiple edges (or multiple lines).

Example 5: If G=(V,E) beagraph, where V= {x,y,z,p} and E={(x,y), (x,z), (x,p), (y,z), (y,z)} then G=(V,E) is a multigraph. This graph is shown below.

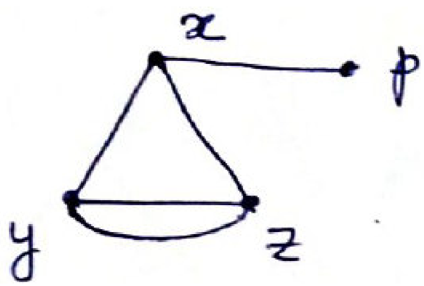
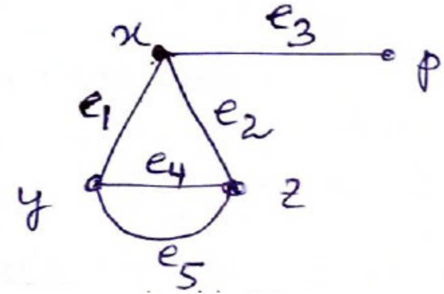



Figure 9: Multigraph G=(V,E)

Here, (y,z) is considered twice in the set e. If we represent G=(x,y), $e_2$=(x,z), $e_3$=(x,p), $e_4$=(y,z) and $e_5$=(y,z), then fig 9 can be represented as given below.




Space for learners:

Figure 10: Multigraph G=(V,E)

Space for learners:

Since $e_4$ and $e_5$ are distinct edges, thus it is not problematic for our readers to write (y,z) twice in the set E. Such edges are known as parallel edges.

Defination 5: A multigraph having loop(s) is called a pseudo graph.

Example 6: If G=(V,E) be a graph, where V={x,y,z,p} and E={(x,x), (x,y), (x,z), (x,p), (y,z), (y,z)}, then G=(V,E) is a pseudo graph.

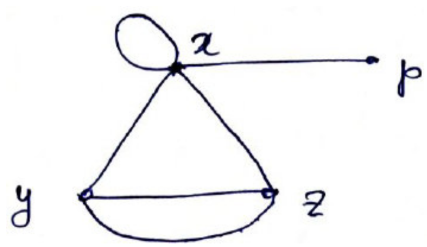



Figure 11: Pseudo graph G=(V,E)

Defination 6: A graph which has neither loops nor multiple edges is called a simple graph.

Example7: Fig5, fig6 are example of simple graph.

**CHECK YOUR PROGRESS**

8. Euler's graph representing Konigsberg bridge problem is a ……… graph.
9. In a multigraph, ……….. are allowed.
10. Is loop allowed in a multigraph?
11. Is loop allowed in a simple graph?
12. Are multiedges allowed in a simple graph?
13. "Every pseudo graph is a simple graph"- true or false?

## 1.4.3 Finite and Infinite Graphs

In this section, we study finite and infinite graph.

Definition7: A graph with a finite number of vertices as well as a finite number of edges is called a finite graph; otherwise, it is an infinite graph.

The graphs, which we considered earlier, are all finite graphs.

Example 8:

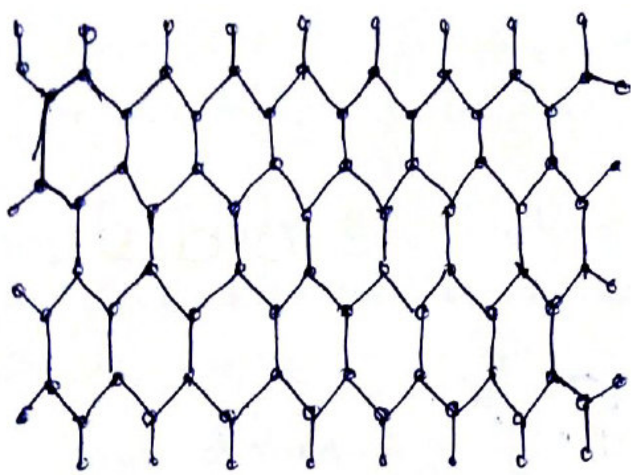


Figure 12: Portion of an infinite graph

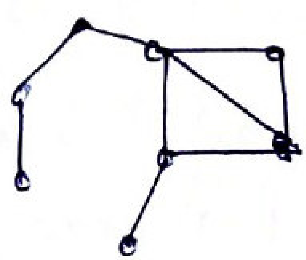


Figure 13: a finite graph

In reality, Fig12 is the graphical representation of Graphene. Graphene is one of the very important atomic-scale hexagonal lattices made of

Space for learners:

carbon-atoms. The discoverers of Graphene were awarded Nobel prize in physics in 2010.

***Space for learners:***

### 1.4.4 Directed and Undirected Graphs

In this section, we discuss directed and undirected graphs. These graphs are used in several real-life problems.

Definition8: A directed graph (or digraph) G=(V,E) consists of a set of vertices V={$v_1$, $v_2$, $v_3$, .....}, a set of edges E={$e_1$,$e_2$,$e_3$,....} and a mapping of that maps every edges onto some ordered pair of vertices ($v_{i,}$, $v_j$). A diagraph is also referred to as oriented graph.

We often make a distinction between the terms "oriented graph" and "directed graph" by considering only these digraphs which have at most one directed edge between a pair of vertices (for digraphs).

The elements of E are called directed edges (or directed lines or areas).

In digraphs, a vertex is represented by a point and an edge by a line segment between $v_i$ and $v_j$ with an arrow directed from vertex $v_i$ to vertex $v_j$.

Example: 9

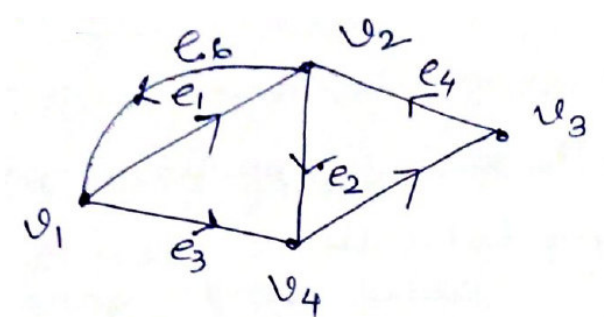



Fig 14: A digraph with four vertices and six edges.

**Space for learners:**

Suppose $e_k=(v_i, v_j)$ is a directed edge in a digraph G= (V,E); then $v_i$ is called the initial vertex of $e_k$ and $v_j$ is called the terminal vertex of $e_k$. In this case, $e_k$ is said to be incident from $v_i$ and to be incident to $v_j$. Also, $v_i$ is adjacent to $v_j$; and $v_j$ is adjacent from $v_i$.

Defination 9: An undirected graph G=(V,E) consist of a set V={$v_1$, $v_2$, $v_3$, .....} of vertices and a set E= {$e_1$,$e_2$, $e_3$,……} of edges such that each edges $e_k \in E$ is associated with an unordered pair of vertices.

Example 10:

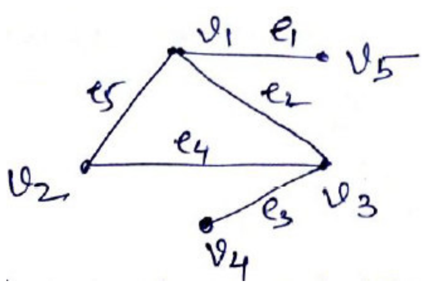



Fig 15: An undirected graph with five vertices and five edges.

## 1.4.5 Different Types of Digraphs

There are various types of digraphs available in literature. Here we discuss some of them.

Definition 10: A digraph that has no parallel edges or self-loops is called a simple digraph.

Example 11:

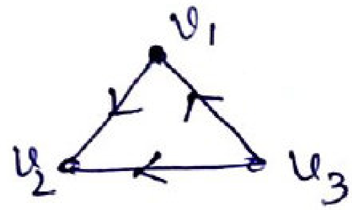



Fig 16: A simple digraph

Definition 11: A digraph that has at most one directed edge between a pair of vertices, but is allowed to have loops, is called an asymmetric graph or anti symmetric digraph.

Example 12:

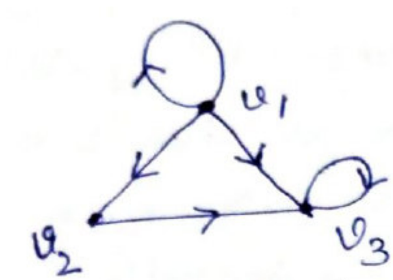



Fig17. An asymmetric digraph

Definition 12: A digraph that is both simple and symmetric is called a simple symmetric digraph.

Definition 13: A digraph that is both simple and asymmetric is called simple asymmetric is called simple asymmetric digraph.

Definition 14: A simple digraph in which there is exactly one edge directed from every vertex to other vertex is said to be a complete symmetric digraph.

## 1.4.6 Incidence and Degree

In this section, we discuss incidence and degree.

Space for learners:

Definition 15: A vertex $v_i$ and an edge $e_k$ are said to be incident with (on or to) each other, if $v_i$ is an end vertex of the edge $e_k$.

Example 13:

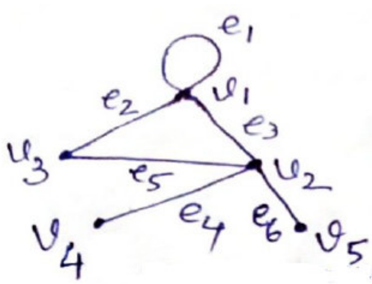



Fig 18: Vertex incident

Here, Vertex $v_4$ is incident with edge $e_4$. The edges $e_5$, $e_3$, $e_4$ and $e_6$ are incident with vertex $v_2$.

Definition 16: Two non-parallel edges are said to be adjacent if they are incident on a common vertex.

Example 14: In example 13, the edges $e_3$, $e_4$, $e_5$ and $e_6$ are adjacent.

Definition 17: Two vertices are said to be adjacent if there is an edge between them.

Example 15:

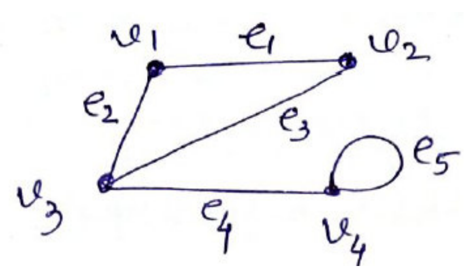



Fig 19: $v_1$ and $v_2$ are adjacent

Here, vertices $v_1$ and $v_2$ are adjacent because there is an edge $e_1$ between them.

Space for learners:

Definition 18: Degree of a vertex $v_i$ is the number of edges incident on a vertex $v_i$, with loops counted twice. It is denoted as $d(v_i)$

Example 16: In example 15, $d(v_1)=2$, $d(v_2)=2$, $d(v_3)=3$ and $d(v_4)=3$

Definition 19: A regular graph is a graph in which all vertices are of equal degree.

Example 17:

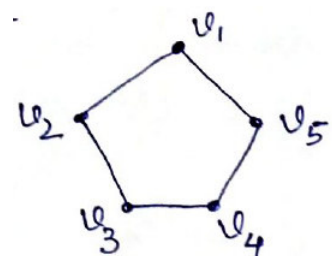



Fig20: A regular graph G=(V,E) of degree two.

Here, $d(v_i)=2 \;\; \forall \; i=1,2,3,4,5$. Thus, G=(V,E) is a regular graph of degree two.

## 1.4.7 Out-Degree and in-Degree in Directed Graph

Definition 20: In a digraph, out-degree of a vertex $v_i$ is the number of edges incident out of a vertex $v_i$. It is denoted by $d^+(v_i)$.

Example 18:

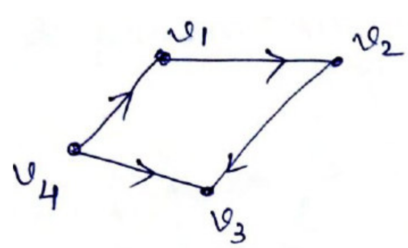



**Fig 21: Out-Degree of G=(V,E)**

Here, in the graph G=(V,E); $d^+(v_1)=1$, $d^+(v_2)=1$, $d^+(v_3)=0$ and $d^+(v_4)=2$.

***Space for learners:***

Definition 21: In a digraph, indegree of a vertex $v_i$ is the number of edges incident into $v_i$. It is denoted by $d^-(v_i)$.

Example 19: In example 18, $d^+(v_1)=1$, $d^+(v_2)=1$, $d^+(v_3)=2$ and $d^+(v_4)=0$.

Space for learners:

## 1.4.8 Isolated Vertex, Pendant Vertex and Null Graph

Definition 22: A vertex is said to be an isolated vertex if it has degree zero.

Definition 23: A vertex having degree one is called a pendant vertex (or an end vertex)

Example 20:

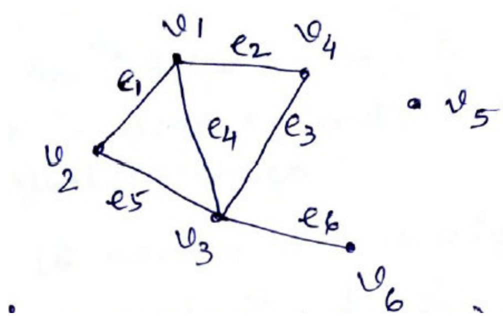



Fig 22: A graph G=(V,E) with six vertices and six edges.

In the above graph, $v_5$ is isolated vertex and $v_6$ is the pendant vertex.

Definition 24: If two adjacent edges have their common vertex of degree two then the two edges are said to be in series.

Example 21: In example 20, the edges $e_1$ and $e_5$ are in series.

Definition 25: A graph is said to be a null graph if every vertex of it has degree zero.

Example 22:

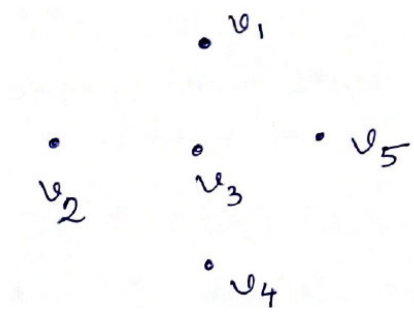



Fig 23: Null graph of five vertices

Definition 26: An isolated vertex is a vertex in which the in-degree and the out-degree are both equal to zero.

Definition 27: A vertex $v_i$ in a digraph is said to be pendant if $d^+(v_i) + d^-(v_i)=1$.

### 1.4.9 Some Results

In this section, we discuss some theorems, problems, etc. related to previous sections.

Theorem 1: If G=(V,E) be an undirected graph, then

$$\sum_{v \in V} deg(v) = 2|E|$$

Or,

The sum of the degrees of all vertices in an undirected graph G=(V,E) is twice the number of edges in G.

Proof: Let G=(V,E) be an undirected graph. In G, every edge is incident with exactly two vertices. Thus, each edge gets counted twice, once at each end. Moreover, degree of a vertex is the number of edges incident with that vertex. Thus, sum of the degrees of all vertices counts the total number of times an edge is incident with a vertex. Thus,

$$\sum_{v \in V} deg(v) = 2|E|$$

.

Space for learners:

**_Space for learners:_**

Theorem 2: The number of vertices of odd degree in a graph is always even.

Proof: Let G=(V,E) be a graph. We write $V=V_1 \cup V_2$, where $V_1$ and $V_2$ are the sets of vertices with odd and even degrees respectively.

Then,
$$\sum_{v \in V} d(v) = \sum_{v_i \in V_1} d(v_i) + \sum_{v_j \in V_2} d(v_j)$$

$$\Rightarrow 2|E| = \sum_{v_i \in V_1} d(v_i) + \text{an even number}$$

$$\Rightarrow \sum_{v_i \in V_1} d(v_i) = 2|E| - \text{an even number}$$

$$\Rightarrow \sum_{v_i \in V_1} d(v_i) = \text{an even number.}$$

Thus, the number of vertices of odd degree in G=(V,E) is even.

Theorem 3: In a directed graph G=(V,E);

$$\sum_{v_i \in V} d^+(v_i) = \sum_{v_i \in V} d^-(v_i) = \sum_{v \in V} d(v_i)$$

Theorem 4: If Gis a directed graph, then

$$\sum_{v_i \in V} d^+(v_i) = \sum_{v_i \in V} d^-(v_i) = |E|$$

Proof: Let G=(V,E) be a directed graph

then
$$\sum_{v_i \in V} d(v_i) = \sum_{v_i \in V} d^+(v_i) + \sum_{v_i \in V} d^-(v_i)$$

$$\Rightarrow 2|E| = 2\sum_{v_i \in V} d^+(v_i) = 2\sum_{v_i \in V} d^-(v_i)$$

$$\Rightarrow \sum_{v_i \in V} d^+(v_i) = \sum_{v_i \in V} d^-(v_i) = |E|. \quad //$$

Problem 1: Determine the number of edges in a graph with 5 vertices, 2 vertices of degree 4, 2 vertices of degree 3 and 1 vertex of degree2.

Solution: Let G=(V,E) be a graph where $|v| = 5$. Let $v_1, v_2, v_3, v_5$ and $v_5$ are five vertices of G=(V,E).

Now, $\sum_{v_i \in V} d(v_i) = 2|E|$

$\Rightarrow (4+4)+(3+3)+2 = 2|E|$

$\Rightarrow |E| = \frac{16}{2} = 8$

Thus, the number of edges of G=(V,E) is 8.

Problem 2: How many vertices are required to draw a graph with 7 edges in which each vertex is of degree2.

Solution: Let there are 'x' number of vertices in the graph.

then $\sum_{v_i \in V} d(v_i) = 2|E|$

$\Rightarrow 2x = 2 \times 7$

$\Rightarrow x = 7$

So, 7 vertices are required.

Problem 3: Show that the maximum number of edges in a simple graph with n vertices is $\frac{n(n-1)}{2}$.

Solution let G=(V,E) be a simple graph then $\sum_{v_i \in V} d(v_i) = 2|E|$.

Given, $|v| = n$ also, the maximum degree of each vertex in a simple graph can be (n-1)

Therefore; $\sum_{i=1}^{n} d(v_i) = 2|E|$

$\Rightarrow n(n-1) = 2|E|$

$\Rightarrow |E| = \frac{n(n-1)}{2}$.

**Space for learners:**

Hence, the maximum number of edges in a simple graph with n vertices is $\frac{n(n-1)}{2}$.

Space for learners:

## 1.5 SUMMING UP

- "Konigsberg bridge problem" has two islands, two banks and seven bridges.
- Leonhard Euler initially represented "Konigsberg bridge problem" using a graph.
- A linear graph G=(V,E) consists of a set of vertices and edges. A graph with p-vertics and q- edges is called a (p,q) graph. The (1,0)-graph is called trivial graph.
- In a multigraph, no loops are allowed but more than one edge can join two vertices. These edges are called multiple edges (or multiple lines). A multigraph having loop(s) is called a pseudo graph. A simple graph which has neither loops nor multiple edges.
- A graph with a finite number of vertices as well as a finite number of edges is called a finite graph; otherwise, it is an infinite graph.
- A simple digraph has no parallel edges or self-loops.
- A simple digraph in which there is exactly one edge directed from every vertex to other vertex is said to be a complete symmetric digraph. A digraph that has at most one directed edge between a pair of vertices, but is allowed to have loops, is called an asymmetric graph.
- In a digraph, indegree of a vertex $v_i$ is the number of edges incident into $v_i$. In a digraph, out-degree of a vertex $v_i$ is the number of edges incident out of a vertex $v_i$.
- An isolated vertex if it has degree zero. A pendant vertex has degree one. A graph is said to be a null graph if every vertex of it has degree zero.

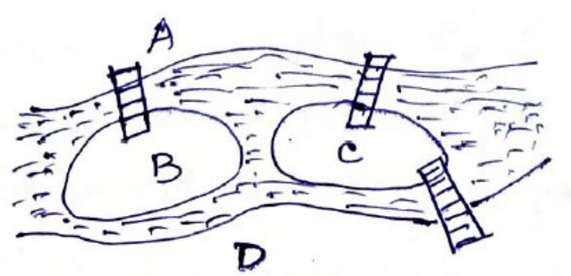
## 1.6 ANSWERS TO CHECK YOUR PROGRESS

Space for learners:

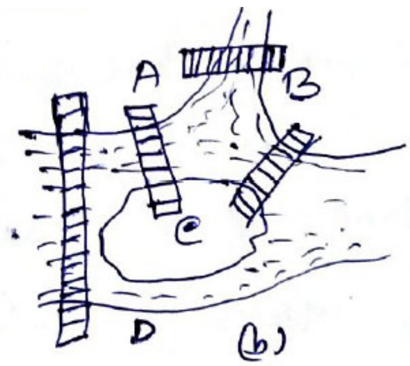
1. Konigsberg bridge problem
2. Leonhard Euler, 1736
3. Two, two, seven
4. True
5. Vertices, edges
6. Vertices
7. Edges
8. Eulerian
9. multiple edges
10. No
11. No
12. No
13. False

## 1.7 POSSIBLE QUESTIONS

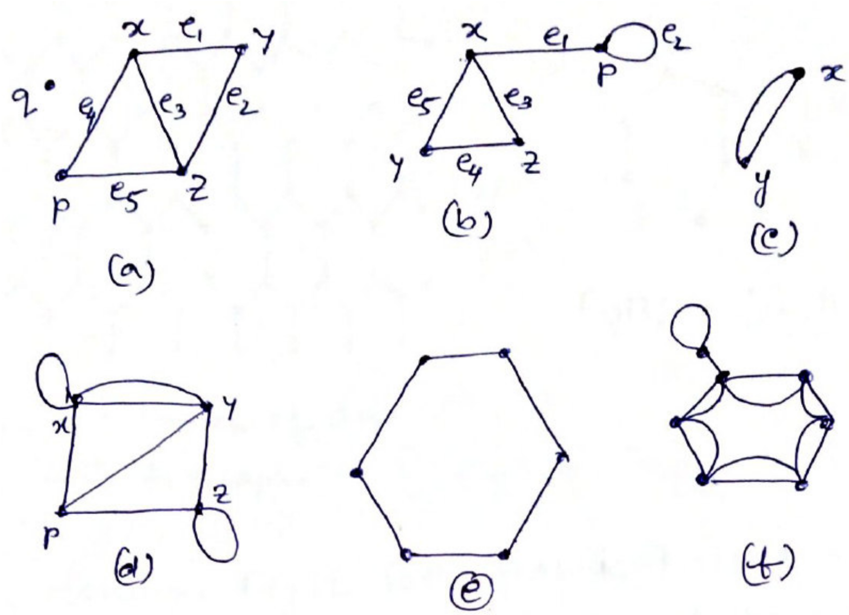
1. Represent the following figures as of Euler's representation process.




(a)



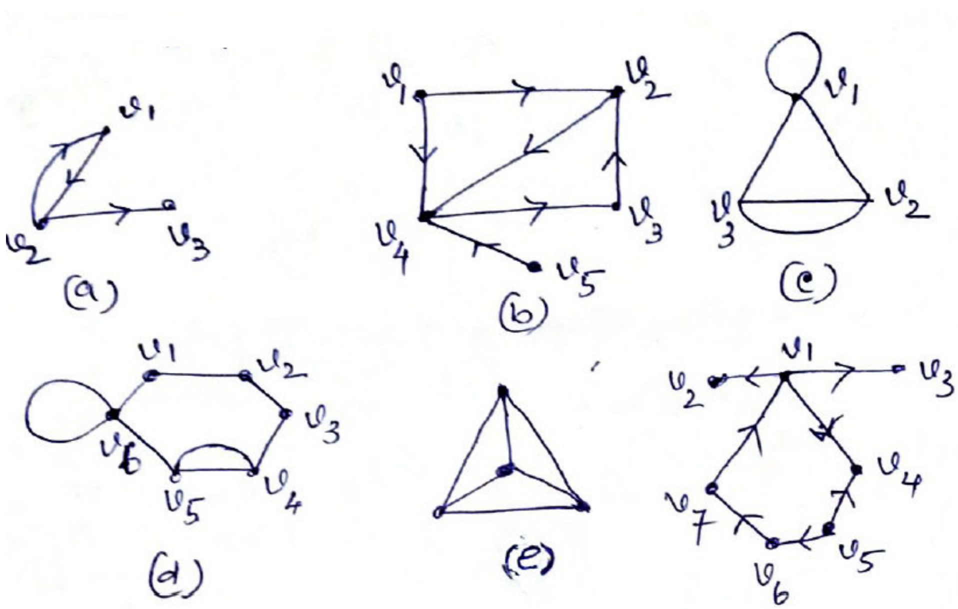

2. Represent graphically $C_4H_{10}$.
3. Draw the graph G=(V,E), where V={x,y,z,p} and E= {(x,y),(x,p), (x,z) (y,z)}.
4. What is the size of an r-regular (m,n)-graph?
5. Prove that the degree of a vertex of a simple graph G on n vertices cannot exceed (n-1).
6. Is it possible to draw a simple graph with 5 vertices and 13 edges? Justify your answer.
7. Identify simple graphs, multigraph, pseudo graphs from the figures given below.




8. The graphical representation of $C_2H_6$ is a ………… graph.

**Space for learners:**

9. Draw a portion of an infinite graph.
10. Draw a finite graph.
11. In a finite graph, number of vertices and number of edges are both ………. .
12. In an infinite graph, number of vertices and number of edges are both…….. .
13. Define digraph.
14. Define undirected graph.
15. Is every graph is a digraph?
16. Choose directed graphs and undirected graphs from below.



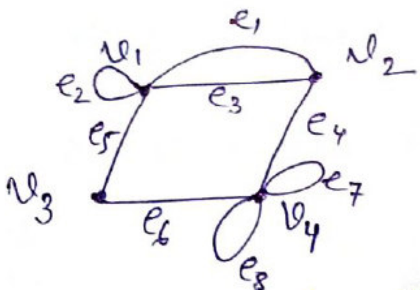

17. Define a simple digraph.
18. Define an asymmetric digraph.
19. Define a simple symmetric digraph.
20. Define a simple asymmetric digraph.
21. Define a complete symmetric digraph.

***Space for learners:***

22. Draw a simple digraph, asymmetric digraph, a simple symmetric digraph, a simple asymmetric digraph, a complete symmetric digraph.

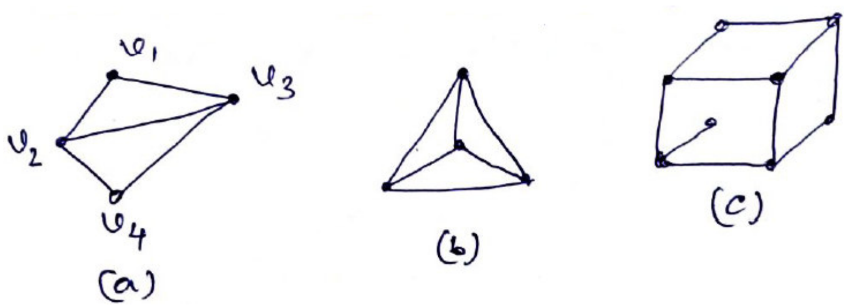
23. Define degree of a vertex.

24. Find degree of vertex ((()))))) of the following graph.

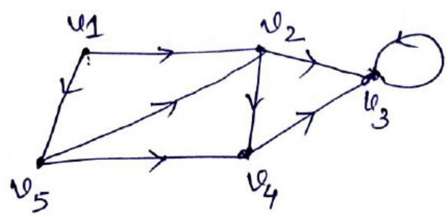



25. Degree of the vertex in (1,0) graph is …………. .

26. Identify the regular graph.

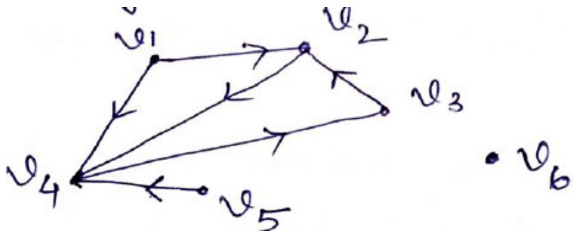



27. Find out-degree and in-degree of each vertex of the following graph.




28. Define isolated vertex.
29. Define pendant vertex
30. Find $d^+(v_i)$ and $d^-(v_i)$ $\forall i = 1, 2, 3$ of the following graph.

**Space for learners:**




31. Prove that in a diagraph,

i) If $v_i$ is an isolated vertex, then

$$d^+(v_i) = 0 \text{ and } d^-(v_i) = 0.$$

ii) If $v_i$ is a pendant vertex, then

$$d^+(v_i) + d^-(v_i) = 1$$

## 1.8 REFERENCES AND SUGGESTED READINGS


- Harary, F. Graph theory. Narosa Publishing House, 2001
- Deo, Narsingh, Graph theory with applications to engineering and computer science, PHI Learning Pvt. Ltd., 2013
- Diestel, Reinhard, Graph theory, Springer, 2006
- Vasudev, C. Graph theory with applications, New age international Publishers, 2018.


Space for learners:

# UNIT 2: PATHS AND CIRCUITS-I

Space for learners:

**Unit Structure:**



## 2.1. INTRODUCTION

Many tangible real-world issues may be successfully analyzed using graphs as mathematical models. Graph theory may be used to formulate issues in physics, chemistry, communication science, computer technology, genetics, psychology, sociology, and linguistics. Graph theory also has strong ties to several disciplines of mathematics, including group theory, matrix theory, probability, and topology. The development of different subjects in graph theory has been aided by several puzzles and issues of a practical character. The classic Konigsberg bridge issue served as a model for the creation of Eulerian graph theory. The Hamiltonian graph theory was derived from Sir William Hamilton's "Around the World" game. The study of "trees" was created to enumerate isomers of chemical compounds, and the idea of acyclic graphs was developed to solve difficulties with electrical networks. In this unit, we present some fundamental concepts of graph theory which include graph isomorphism, various types of subgraphs, walks, trails, paths and circuit.

**Space for learners:**

## 2.2. UNIT OBJECTIVES

On completion of this unit students will be able to:

- Explain the definition, concept, and properties of graph isomorphism.
- Explain and differentiate various types of subgraphs.
- Define walks, trails, paths and circuits.
- Differentiate between connected and disconnected graph.

## 2.3. ISOMORPHISM IN GRAPHS

In graph theory, a graph $G$ can be called as equivalent to another graph $G'$ if both the graphs are identical in terms of their vertices and edges. This concept is called as graph ***isomorphism***. Two ***isomorphic*** graphs may use different labels for the vertices and may have drawn differently, but they have exactly the same number of vertices and same sets of edges. The formal definition on graph isomorphism is presented below.

**Definition 2.3.1:**Graph ***Isomorphism*** is a concept in graph theory which states that any two graphs, $G$ and $G'$ are called as ***Isomorphic*** if there is a bijection between the vertex sets of $G$ and $G'$. Formally, a graph $G(V, E)$ is isomorphic to another graph $G'(V', E')$ if there exists a bijective function $\Phi: V \rightarrow V'$ such that if any vertices, $u, v \in V$ are there is an edge from $u$ to $v$ in $G$ then there must be an edge from $\Phi(u)$ to $\Phi(v)$ in $G'$. Mathematically two isomorphic graphs $G$ and $G'$ are denoted as $G \simeq G'$. The map $\Phi$ is termed as an isomorphism from $G$ to $G'$.

**Example 2.3.1:** The graphs shown in figure 2.1 are examples of isomorphic graphs. Both the graphs have equal number of vertices and edges. The graph on left has the vertices,$< u_1, u_2, \dots \dots \dots, u_n >$ and the graph on right has the vertices $<v_1, v_2, \dots \dots \dots, v_n >$. In this example,

$\Phi(u_i) = v_i$. The adjacency matrices of both the graphs are presented in Table 2.1. The adjacency matrices of both the graphs are identical as they are isomorphic.

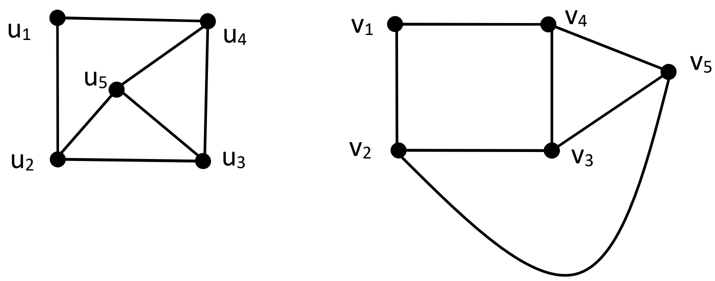



Fig. 2.1 Isomorphic Graphs

Table 2.1 Adjacency matrices of the graphs shown in Fig. 2.1

| | ***u1*** | ***u2*** | ***u3*** | ***u4*** | ***u5*** |
|---|---|---|---|---|---|
| ***u1*** | 0 | 1 | 0 | 1 | 0 |
| ***u2*** | 1 | 0 | 1 | 0 | 1 |
| ***u3*** | 0 | 1 | 0 | 1 | 1 |
| ***u4*** | 0 | 1 | 1 | 0 | 1 |
| ***u5*** | 0 | 1 | 1 | 1 | 0 |

| | ***v1*** | ***v2*** | ***v3*** | ***v4*** | ***v5*** |
|---|---|---|---|---|---|
| ***v1*** | 0 | 1 | 0 | 1 | 0 |
| ***v2*** | 1 | 0 | 1 | 0 | 1 |
| ***v3*** | 0 | 1 | 0 | 1 | 1 |
| ***v4*** | 0 | 1 | 1 | 0 | 1 |
| ***v5*** | 0 | 1 | 1 | 1 | 0 |

**Theorem 2.3.1:** Consider $\phi$ be an isomorphism of the graph $G = (V, E)$ to the graph $G' = (V', E')$. Consider a vertex $v \in V$. Then $degree(v) = degree(\phi(v))$. i.e., the degree of vertices is preserved by isomorphism.

*Proof*: Consider two vertices $u, v \in V$. If $u$ is adjacent to $v$in graph $G$, then $\phi(u)$ must be adjacent to $\phi(v)$ in graph $G'$. So, the number adjacent vertices of $v$in $G$ is equal to the number of adjacent vertices of $\phi(v)$ in $G'$. Hence, $degree(v) = degree(\phi(v))$.

*Properties*: If a graph $G$ is isomorphic to another graph $G'$ by a bijection $\phi$, then the following properties hold true.

- Number of vertices in $G$ is same as the number of vertices in $G'$.
- Number of edges in $G$ is same as the number of edges in $G'$.

***Space for learners:***

- Both in-degree and out-degree of a vertex $v$is same as the in-degree and out-degree of $\phi(v)$.

**Definition 2.3.2:** An ***automorphism*** of a graph is a type of symmetry in graph theory in which the graph is mapped onto itself while retaining the edge–vertex connection. In other words, a graph $G$ is isomorphic onto itself.

## 2.4. SUBGRAPHS

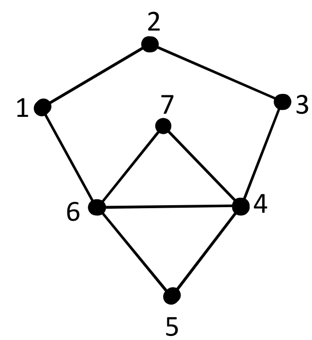
A graph $G'$is a subgraph of another graph $G$ if all the vertices and edges of $G'$ belong to the $G$ and each edge in $G'$ has the same source and destination in $G$ as $G'$. A subgraph can be called as a subpart of another graph. The formal definition of subgraph is presented in Definition 2.4.1.

**Definition 2.4.1:** A graph $G'(V', E')$ is called as a subgraph of another graph $G\ (V, E)$if $V' \subseteq V$,$E' \subseteq E$ and $R_{G'}$ is the restriction of$R_G$ to$E'$. The graph $G'$is a ***proper subgraph*** of $G$ if $V' \subset V$ or $E' \subset E$. The graph $G$ can be called as ***supergraph*** of $G'$ if $G'$is a subgraph of $G$. A graph $G'$ is called as an ***induced subgraph*** of $G$ if a vertex $v \in V$ is adjacent to another vertex $u$ in $G$ and $u, v \in V'$ then $v$ must be adjacent to $v$ in $G'$as well. If $G'$is an induced subgraph of $G$ and vertex set of $G'$, $V' \subseteq V$ then $G'$ is called as the subgraph of $G$ induced by $V'$ and is denoted by $G[V']$. If $G'(V', E')$ is an induced subgraph of $G\ (V, E)$ and $E' \subseteq E$ then $G'$ is called as the subgraph of $G$ induced by $E'$ and is denoted by $G[E']$. A subgraph $G'(V', E')$of $G\ (V, E)$ is called as a***spanning subgraph*** of $G$ if $V' = V$.

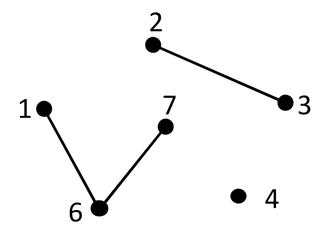
**Example 2.4.1:** Figure 2.2 (b)-(d) shows various types of subgraphs of the graph $G$. The graph $G$ has the vertex set, $V = \{1,2,3,4,5,6,7\}$. The graph in (b) is a subgraph of $G$ but not an induced subgraph because in $G$ the vertex 1 is adjacent to 2 and vertex 4 is adjacent to 3 and 7; but the same is not true in this subgraph. Also, this is not a spanning subgraph as it does not include all the vertices of $G$. The graph shown in (c) is an induced subgraph of $G$ as this subgraph doesn't include any pair of vertices which are adjacent in $G$ but not in this subgraph. The subgraph in (d) includes all the vertices of $G$ but not all the edges. So, this is a spanning subgraph of $G$.

Space for learners:

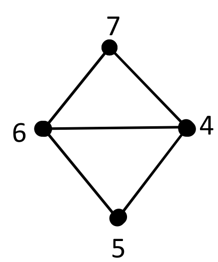
**Definition 2.4.2:** A ***clique*** is a subgraph of a graph $G$ whose vertex set is a subset of the vertex set of $G$ and any two vertices of the clique are adjacent. Informally, a clique is a complete subgraph of another graph. That means, all the vertices of the clique are adjacent to each other. A clique is called as ***maximal clique*** if no adjacent vertex can be added to expand the clique. A ***maximum clique*** is a clique which contains maximum possible vertex.



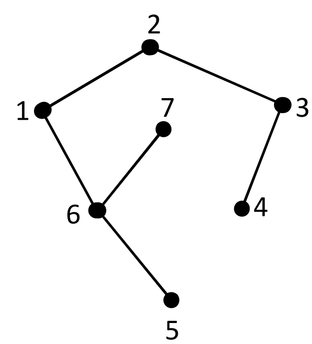

(a) Graph G




(b) Subgraph of G




(c) Induced Subgraph of G




(d) Spanning subgraph of G

Fig. 2.2 Various types of subgraphs

**Example 2.4.2:** Figure 2.3 presents some examples of cliques of the graph shown in the subfigure (a). The subgraph shown in subfigure (b) is a clique as there is an edge between any two vertices in the subgraph. Similarly, the subgraphs shown in subfigures (c) and (d) are also cliques. While the subgraph in subfigure (e) is not a clique as the vertex 1 is adjacent to vertex 2 only. The cliques in (c) and (d) are maximal cliques as if we add any other vertex to them, then they will no longer

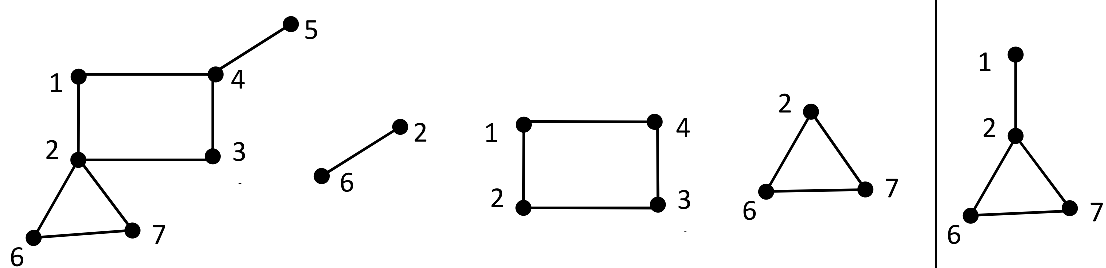
***<u>Space for learners:</u>***

be cliques. The clique in (b) is not maximal as there is a possibility of adding another vertex (vertex 7) to expand the clique.

Space for learners:

## 2.5. WALKS, TRAILS, PATHS AND CIRCUIT

A ***walk*** is a finite alternating series of vertices and edges that starts and ends with vertices, with each edge connecting the vertices before and after it. A walk may have repeated vertices but not edges. A walk is called as a ***closed*** walk if the starting vertex and the ending vertex is the same,otherwise, it is called as ***open***. An open walk with no repeated edges is called as a ***trail***. The vertices may repeat in a trail. A trail with non-repeated vertices is called as a ***path***. A non-empty trail in which the starting and ending vertices are the only vertices that are repeated is called as a ***circuit*** or a ***cycle***. Definition 2.5.1 gives the formal definition of walk, trail, path and circuit.




(a) Main Graph, $G$(b) clique of $G$(c) clique of $G$ (d) clique of $G$ (e) subgraph of $G$

Fig. 2.3 Various subgraphs of the graph shown in (a). The subgraphs (b)-(d) are cliques but the one in (e) is not a clique. The cliques (c) and (d) are maximal cliques but not the one in (b).

**Definition 2.5.1:** Consider a graph $G = (V, E)$ with the vertex set, $V = \{v_0, v_1, v_2, \dots \dots \dots, v_n\}$ and the set of edges, $E = \{e_0, e_1, e_2, \dots \dots \dots, e_m\}$. A walk, $W$ of the graph $G$ can be defined as an alternating sequence of vertices and edges,$W =< v_0, e_1, v_1, e_2 v_2, \dots \dots \dots, e_p, v_p >$such that $e_i$ is the edge from the vertex $v_{i-1}$ to $v_i$. The vertex $v_0$ is called as the ***origin*** and the $v_p$ is called as the ***terminal*** of $W$. The walk, $W$ joins the vertex $v_0$ to $v_p$ and the walk is called as $v_0 - v_p$ walk. The walk which terminates at origin, i.e., $v_0 = v_p$ is termed as a ***closed*** walk otherwise termed as ***open***. When the edges of a walk are distinct, the walk is

Space for learners:

called as a ***trail*** and when the vertices are distinct then it is called a ***path***. A closed trail with distinct vertices is called as a ***circuit***. The number of edges present in a walk can be referred as the ***length*** of the walk.

**Definition 2.5.2:** A cycle which has $n$ vertices and $n \geq 3$has a length of $n$. A graph having a cycle of length $n$ is denoted as $C_n$. A cycle with length 3 i.e., $C_3$ is called as a ***triangle***, $C_4$ is termed as a ***square*** and $C_5$ as ***pentagon***.

**Lemma 2.5.1:** Every $u - v$ walk of a graph $G$ contains a $u - v$ path.

*Proof*: We prove this lemma by the method of induction on length of the walk $u - v$. Let $l$ be the length of the walk $u - v, W$.

Base case: $l = 0$.The walk contains a single vertex $u = v$ with no edge. Then there is obviously a $u - v$ path of length 0.

Induction step: $l \geq 1$.Let us try to prove the lemma for walks of length less than $l$. If $W$ is a $u - v$ walk with length $l$ contains no repeated vertex then the walk is already a path. If not, then suppose there exist some vertex $i$ in $W$which occur more than once in the walk then removing the all the occurrences of $i$(and the corresponding edges) leaving one then we will get a walk $W'$ of length less than $l$. By induction hypothesis there exists a $u - v$ path in $W'$ and as $W'$ is contained in $W$ hence there is a $u - v$path in $W$.

**Lemma 2.5.2:** A closed walk of odd length contains a cycle.

*Proof*: Let $W$ be a closed walk with odd length $l$. Using the method of induction, we can prove that $W$ contains a cycle.

Base case: $l = 1$. If the length of the walk is 1 then there is a self- loop and $W$ contains only a single vertex, hence there is a cycle.

Induction step:$l \geq 3$. Consider the walk, $W$ consists of a vertex set $V = < v_0, v_1, v_2, \dots \dots, v_n >$ and $v_0 = v_n$. If each vertex$v_i (0 \leq i \leq n)$ is distinct, then the walk itself is a cycle. If not, then there exist two positiveterms $i, j$ such that $i < j$, $v_i, v_j \in V - \{v_0, v_n\}$and $v_i = v_j$. Now we can split this walk $W$ into two closed walks $W_1$ and $W_2$ at $v_i$ such that $W_1$ includes the vertices $v_i, v_{i+1}, v_{i+2}, \dots \dots, v_j$ $(v_i = v_j)$ and $v_0, v_1, .. v_i, v_{j+1} \dots \dots, v_n$. So, sum of the lengths of $W_1$ and $W_2$ will be equal to $l$. Since the length $l$ is odd, one of these closed walks will be odd and by induction hypothesis, it has a cycle.

Space for learners:

**Example 2.5.1:** The graphs in figure 2.4 illustrates various subtypes of walk. The subfigure (b) presents an open walk (3→2→1→3→4→1) of the graph shown in subfigure (a). In this walk, 3 is the origin and 2 is the terminal. This is called as an open walk as the origin and the terminal is not the same.

The walk shown in (c)(1→2→3→4→5→3→1) is a closed walk as the origin and the terminal vertices are the same which is 1. This cannot be called as a circuit or cycle as the vertex 3is repeated twice in the walk.

The walk shown in (d) is (3→2→1→3→4→5→6) represents a trail of the graph in (a). Here, the vertex 3 repeated twice but all the edges are distinct which satisfies the properties of a trail.

The subfigure (e) presents a path of graph in (a). All the vertices and edges along the path 3→2→1→4→5→6→7 are distinct which satisfies the properties of a path.

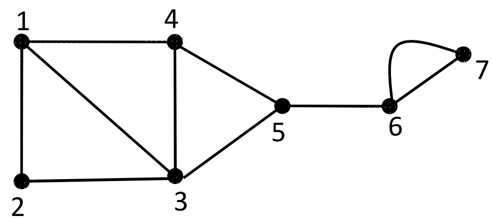
The example shown in (f) represent a circuit of the graph in (a). A circuit must have distinct edges and distinct vertices except for the starting and ending vertices. The walk 3→2→1→4→5→3 satisfies the properties of a circuit, so we can term this walk as a circuit.

## 2.6. CONNECTED AND DISCONNECTED GRAPHS

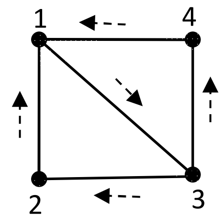
A graph is called as connected if each vertex of the graph is reachable from all other vertices. Otherwise, the graph is called as a disconnected graph. A disconnected graph contains more than one connected subgraph. Such subgraphs are called as components of a graph. Formal definition of connected graphs, disconnected graphs and components are given by Definition 2.6.1.

**Definition 2.6.1:** Consider a graph, $G(V, E)$. If there exist a $u - v$ path in $G$ such that $u, v \in V$, then $u$ is said to ***connected*** to $v$. The relation ***connected*** is an equivalence relation on the vertex set $V$ of graph $G$. Suppose $V_1, V_2, \dots, V_k$ are equivalence classes of $V$. Then a subgraph with vertex set $V_i$, $1 \leq i \leq k$ is a ***component*** of G. If $k = 1$, then the graph $G$ is ***connected graph*** and the graph $G$ will be called as

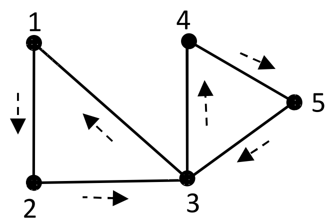
***disconnected graph*** if $k \geq 2$. In simple words, a ***connected graph*** can have at most one component. In case of a ***connected graph***, $G$ there will be a path $u - v$ for any pair of vertices $u, v \in V$.



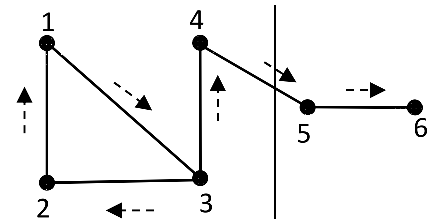

(a) A sample graph $G$



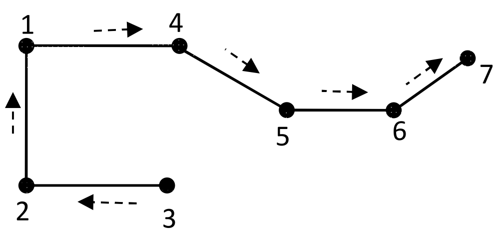

(b) An open walk of $G$: 3→2→1→3→4→1



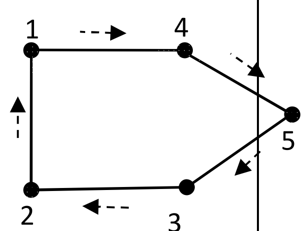

(c) A close walk of $G$: 1→2→3→4→5→3→1




(d) A trail of $G$: 3→2→1→3→4→5→6




(e) A path of $G$: 3→2→1→4→5→6→7




(f) A circuit of $G$: 3→2→1→4→5→3

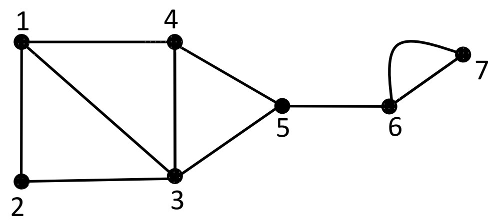
Fig. 2.4 Example to illustrate walk, trail, path and circuit

**Definition 2.6.2:** Let $G'(V', E')$ be a subgraph of a graph $G(V, E)$. The subgraph $G'$ will be a ***maximally connected component*** of $G$ if $G'$ is connected and for any vertex $v$ such that $v \in V$ and $v \notin V'$ there is no vertex $u \in V$ which is adjacent to $v$. Informally, if there exist no vertex in $G$ which can be added to $G'$ and $G'$ still be connected.

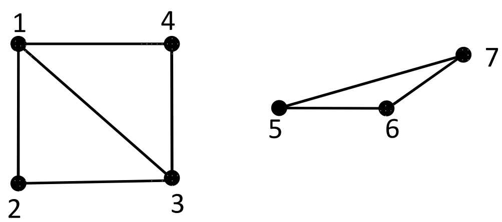
**Example 2.6.1:** The figures shown in figure 2.4 illustrates the graph connectedness. The graph shown in (a) represents a connected a graph. This graph has 7 vertices and each vertex is reachable from all remaining 6 vertices. The graph in (b) also has 7 vertices, but the

***<u>Space for learners:</u>***

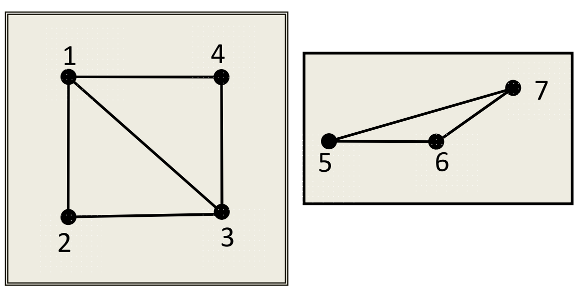
vertices 1-4 are not reachable from the vertices 5-7. Thus, this graph is disconnected. The subfigure (c) shows the components of the graph in (b). The components are enclosed within the rectangular boxes. One component has the vertex set {1,2,3,4} and the other has the vertex set {5,6,7}. Both the components are connected graphs individually.




(a) A connected graph




(b) A disconnected graph




(c) Components of the graph shown in (b)

Fig. 2.5 Sample graphs illustrating connected graph, disconnected graph and components of a graph

**Theorem 2.6.1:** A simple graph $G$ with $n$ vertices and minimum degree $\delta \geq \frac{n-1}{2}$ is connected.

*Proof:* We shall prove this theorem by contradiction. Suppose $G$ is not connected and has at least two components, say $G_1$ and $G_2$. Let us consider $v$ be any vertex of $G_1$. The degree of $v$, $d(v) \geq \frac{n-1}{2}$ as $\delta \geq$

***Space for learners:***

$\frac{n-1}{2}$. Hence, $v$ has at least $\frac{n-1}{2}$ adjacent vertices in $G_1$and so, $G_1$contains at least $\frac{n-1}{2}+1=\frac{n+1}{2}$vetrices. Similarly, $G_2$ also contains minimum $\frac{n+1}{2}$ vertices. Hence, the graph, $G$ has a minimum of $\frac{n+1}{2}+\frac{n+1}{2}=n+1$ vertices, which is a contradiction.

Space for learners:

**Theorem 2.6.2:** If a simple graph $G$ is not connected then $\overline{G}$ is connected.

*Proof*: Let $G(V,E)$ has more than one component. Let $u,v$ be any two vertices of $G$ (and of $\overline{G}$). If $u,v$ belongs to two different components of $G$ ($u$ is not adjacent to $v$ in $G$), then they are adjacent in $\overline{G}$. So, $u$ and $v$ are connected in $\overline{G}$. If $u,v$ belongs to the same component of $G$ then let us select a vertex $w$ from a different component. The edges $uw$ and $vw$ do not belong to $G$ but they belong to $\overline{G}$. Then there exists a path $uwv$ in $\overline{G}$, which is nothing but a $u-v$ path. Hence $\overline{G}$ is connected.

**Theorem 2.6.3:** A graph with $n$ vertices and $k$ components can have at most$\frac{(n-k)(n-k+1)}{2}$ edges.

*Proof*: Let $G_1, G_2, \dots G_k$be the components of a graph $G$ and let $n_i$ be the number of vertices of the $i^{th}$ component of $G$ such that $1\le i\le k$ and $e(G_i)$ represents the number of edges present in $G_i$.

Any graph of $n$ vertices can have at most $\frac{n(n-1)}{2}$ vertices (this happens when the graph is a complete graph which mean each vertex is connected with each other).

Thus, for any $G_i, 1\le i\le k$, $e(G_i)\le\frac{n_i(n_i-1)}{2}$, and hence $e(G)\le\sum_{i=1}^{k}\frac{n_i(n_i-1)}{2}$.

Since each component has at most one vertex, for any $G_i$, $n_i>1$ and$n_i=n-$
$(sum\ of\ the\ vertices\ in\ all\ the\ components\ of\ G\ except\ G_i)$

Hence, $n_i\le n-k+1$, so $\sum_{i=1}^{k}\frac{n_i(n_i-1)}{2}\le\sum_{i=1}^{k}\frac{(n-k+1)(n-k)}{2}=\frac{(n-k+1)(n-k)}{2}=\frac{(n-k)(n-k+1)}{2}$,

Hence proved, $e(G)\le\frac{(n-k)(n-k+1)}{2}$.

**Theorem 2.6.3:** A graph $G(V, E)$ is connected if and only if for any partition of vertex set $V$ into subsets $V_1$ and $V_2$, there is an edge from any vertex of $V_1$ to any vertex of $V_2$.

*Proof*: Let a graph $G(V, E)$ is connected and let $V = V_1 \cup V_2$. Let us consider two vertices $u, v$ such that $u \in V_1$ and $v \in V_2$. There exists a $u - v$ path in $G$, say$< u, w_0, w_2, \dots, w_k >(w_k = v)$ as $G$ is connected. Let $i$ be the smallest positive integer such that $w_i \in V_2$, then $w_{i-1} \in V_1$. Since $w_{i-1}$ and $w_i$ are adjacent, thus there is an edge from $w_{i-1} \in V_1$to $w_i \in V_2$. Conversely, let $G$ is not connected. Thus, $G$ has at least two components. Let $V_1$ represents the set of one component and $V_2$ represents the other component. It is obvious that there is no edge from any vertex of $V_1$ to any vertex of $V_2$. Hence it proves the theorem.

**CHECK YOUR PROGRESS**

**i.** Let two groups $G$ and $H$ are two isomorphic graphs. Which of the following is true in terms of $G$ and $H$?

**a.** Number of vertices in $G$ is same as the number of vertices in $H$

**b.** Number of edges in $G$ is same as the number of edges in $H$.

**c.** Both in-degree and out-degree of a vertex $v$is same as the in-degree and out-degree of $\phi(v)$.

**d.** All of the above

**ii.** A subgraph containing all the vertices is called as

**a.** Induced subgraph

**b.** Spanning graph

**c.** Clique

**d.** None of the above

**iii.** The subgraph in which all the vertices are adjacent to each other is called as

**a.** Induced subgraph

**b.** Spanning graph

**c.** Clique

**d.** None of the above

**iv.** What will be the number of edges in a walk with $n$ vertices?

**a.** $n$-1

*Space for learners:*

b. $n$

c. $n + 1$

d. $2n$

**v.** Which of the following is true in terms of a walk?

**a.** All the vertices must be distinct.

**b.** All the edges must be distinct.

**c.** Both (a) and (b)

**d.** None of the above.

**vi.** A walk with same starting and ending vertex is called as a

**a.** Open walk

**b.** Closed walk

**c.** Cycle

**d.** Trail

**vii.** A walk with no repeated vertex is called as a

**a.** Open walk

**b.** Closed walk

**c.** Path

**d.** Trail

**viii.** A closed walk with distinct vertices is called as a

**a.** Cycle

**b.** Path

**c.** Trail

**d.** Clique

**ix.** The maximum number of edges a graph with $n$ vertices and $k$ edges is

**a.** $\frac{(n-k)(n-k+1)}{2}$

**b.** $\frac{(n-k)(n-k-1)}{2}$

**c.** $\frac{(n-k)n}{2}$

**d.** $\frac{(n-k-1)(n-k+1)}{2}$

**x.** The maximum number of components that a connected graph with $n$ vertice scan have is

a. 0

b. $n/2$

c. $n$-1

d. $n$

***Space for learners:***

## 2.7. SUMMING UP

Space for learners:

- ***Graph Isomorphism*** is a concept in graph theory which states that any two graphs, $G$ and $G'$ are called as *Isomorphic* if there is a bijection between the vertex sets of $G$ and $G'$. Two isomorphic graphs have equal number of vertices and edges. The degree of a vertex $v$ in $G$ is same as the degree of its corresponding vertex in $G'$.
- A ***subgraph*** can be called as a subpart of another graph. An ***induced subgraph*** of a graph is another graph generated from a subset of the graph's vertices and all of the edges joining pairs of vertices in that subset. A ***spanning subgraph*** is a subgraph of another graph if the vertex set remains the same in both the subgraph and the original graph. A ***clique*** is a complete subgraph. of another graph.
- A ***walk*** is a finite alternating series of vertices and edges that starts and ends with vertices, with each edge connecting the vertices before and after it. A walk may have repeated vertices but not the edges of another graph. An open walk with no repeated edges is called as a ***trail***. The vertices may repeat in a trail. A trail with non-repeated vertices is called as a ***path***. A non-empty trail in which the starting and ending vertices are the only vertices that are repeated is called as a ***circuit*** or a ***cycle***.
- A graph is called as connected if each vertex of the graph is reachable from all other vertices. Otherwise, the graph is called as a disconnected graph. A disconnected graph contains more than one connected subgraph. Such subgraphs are called as components of a graph. A trail is a walk in which the starting and ending vertices are the only vertices that are repeated is called as a ***circuit*** or a ***cycle***.

## 2.8. ANSWERS TO CHECK YOUR PROGRESS

| | | | | |
|---|---|---|---|---|
| (i)d | (ii)b | (iii)c | (iv)a | (v)b |
| (vi)b | (vii)c | (viii)a | (ix)a | (x)a |

## 2.9. POSSIBLE QUESTIONS

i. What are the properties of graph isomorphism?

ii. What is meant by subgraph? What are the different types of subgraphs available?

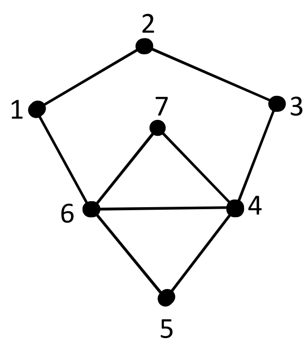
iii. What is meant by an induced subgraph? Explain with an example.

iv. What do mean by a walk of a graph? What is the difference between a trail and a path?

v. What is the difference between a closed walk and a cycle?

vi. What is meant by components of a graph? How is it related to graph connectedness?

vii. Find a trail, a cycle and a path in the graph given below.




viii. Verify if the sequence given below can be considered as a trail. Justify your answer.

$$2 \rightarrow 3 \rightarrow 1 \rightarrow 4 \rightarrow 6 \rightarrow 2 \rightarrow 1$$

ix. Verify if the sequence given below can be considered as a path. Justify your answer.

$$2 \rightarrow 1 \rightarrow 5 \rightarrow 3 \rightarrow 4 \rightarrow 1$$

x. Verify if the sequence given below can be considered as a cycle. Justify your answer.

$$2 \rightarrow 3 \rightarrow 1 \rightarrow 4 \rightarrow 6 \rightarrow 2$$

## 2.10. REFERENCES AND FURTHER READINGS


- Graph Theory with Applications to Engineering and Computer Science by Narsingh Deo, Published by Prentice Hall India Learning Private Limited.


Space for learners:

- Introduction to Graph Theory by Richard J Trudeau, Published by Courier Corporation.
- A First Course in Graph Theory by Gary Chartrand and Ping Zhang, Published by Courier Corporation.
- Graph theory with applications by John Adrian Bondy, Published by Elsevier Publishing Company


***Space for learners:***

# UNIT 3: PATHS AND CIRCUITS-II

Space for learners:

**Unit Structure:**

## 3.1 INTRODUCTION

Space for learners:

This unit focuses on three very important concepts of graph theory- Euler graph, Hamiltonian graph, and Bipartite graph. Almost every real-world problem involving discrete groupings of items, where the focus is on the relationship between them rather than the intrinsic features of the items, may be translated into one of these graphs. Thus, these graphs find a wide range of applications such as in computer science, communication science, economics, computer graphics, electronic circuit design, mapping genomes, operation research, error correction code etc. The goal of this unit is to familiarize the students with the various terms and definitions related to these graphs and to introduce some important theorems.

## 3.2 UNIT OBJECTIVES

After completing the module learners will be able to-

- define the terms Euler graph, Hamiltonian graph, and Bipartite graph.
- determine whether a graph is an Euler graph or not.
- determine if a graph is a Hamiltonian graph.
- list the properties of the Bipartite graph.
- Determine whether there exists a perfect match in a graph or not.

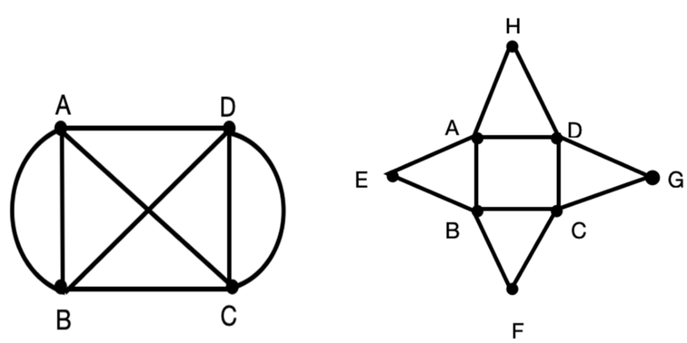
## 3.3 EULER GRAPHS

In 1736, Swiss mathematician Leonhard Euler in his famous paper, where he solved the Königsberg bridge problem, raised an interesting problem. The problem was, given a graph $G$, is it possible to find a walk, with the same staring and the end vertices and includes each edge of $G$ exactly once. In the same paper, he also presented the solution of the problem and introduced the concept of the Euler graph which now is extensively used in many fields- ranging from DNA sequence reconstruction to circuit designs.

## 3.3.1 Definitions

- **Euler line:** in a given graph *G*, if there exists a closed walk (having the same starting and end vertex) such that it contains each edge of *G* exactly once, then the walk is called an Euler line.
- **Euler Graph:** A graph that contains an Euler line is called an Euler Graph.
- **Unicursal Line:** A walk that contains all the edges of the graph exactly once, with different starting and end vertices, is called a unicursal line. A unicursal line is an Euler line with the dropped constraint of the walk being closed. Hence, it is also referred to as the open Euler line.
- **Unicursal Graph:** A graph that contains a unicursal line is called the Unicursal Graph.

**Example:** The graphs in figure 3.1 are examples of Euler graphs. The graph in figure 3.1(a) consists of four vertices and eight edges. If we start from vertex A, the walk (A, B, C, D, B, A, C, D, A) contains all the edges of the graph. Similarly, the graph in figure 3.1 (b) is also an Euler graph and one possible Euler line is-(A, B, C, D, A, H, D, G, C, F, B, E, A) starting and ending with vertex A.




(a) (b)

Fig 3.1: Examples of Euler Graph

However, the graphs in figure 3.2 are not Euler graphs. It is not possible to get any Euler line starting from any vertex of these graphs.

The graphs in figure 3.3 are also not Euler graphs. But they are unicursal graphs. For the graph in figure 3.3 (a), one possible

***Space for learners:***

unicursal line is (A, B, C, A, D, C, E, D, C). Similarly, (B, F, A, B, C, A, D, C, E, D) is a possible unicursal line for the graph in figure 3.3 (b).

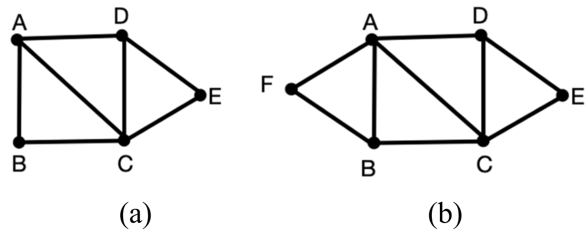



Fig 3.3: Example of Unicursal Graph and Unicursal Line

### 3.3.2 Theorems on Euler Graph

***Theorem 3.1*:** A connected graph $G$ is an Euler graph if and only if each vertex in $G$ is of even degree.

***Proof:***

**Part A- Necessary:**

Assume that $G$ is a Euler graph. By the definition of the Euler graph, there must exist a Euler line in $G$, which is a closed walk containing each edge of $G$ exactly once. This implies that- every time a vertex $v$ is encountered while tracing the Euler line, there must be-

1. A new edge incident on $v$ which serves as the entry edge and
2. Another new edge incident out of $v$ which serves as the exit edge.

This holds true for each intermediate vertex $v$ indicating that the intermediate vertices must be of even degree. As an Euler line is a closed walk, there exists only one terminal vertex which is both the starting and the end vertex. Thus, the walk starts from the terminal vertex and came back to the same vertex at the end. This indicates that the terminal vertex must also be of even degree.

Thus, we may conclude that if a graph $G$ is an Euler graph then each vertex must be of even degree.

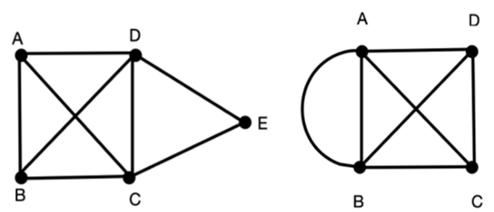
**Part B- Sufficiency:**

Assume that each vertex of $G$ is of even degree. Let's start with any vertex $v$ of $G$and then arbitrarily trace a walk in such a way that an edge is traversed only once. As each vertex of $G$ is of even degree,

*Space for learners:*

we can exit from any vertex *u* that we enter in this walk. The walk may come to end only when we eventually reach *v*. Let this closed walk be termed as *h*. If *h* contains all the edges of *G* then it's an Euler line, otherwise, we will remove *h* from *G*, which results in a subgraph *G'*. As the vertices in both *h* and *G* are of even degree, the vertices in *G'* must also be of even degree. As the graph *G* was connected, *G'* must have at least one common vertex with *G*. Let this common vertex be *w*. Now, starting from *w*, again we trace another arbitrary walk *h'* containing an edge of *G'* only once. As *h'* also has vertices only of even degree, this walk may also come to an end only on encountering *w*. Now we remove *h'* from *G'* and join with *h*, which results in a new walk that starts and ends with *v* but with more number of edges. We can apply this process recursively until we obtain a closed walk that contains all the edges of *G*. ThusG is an Euler graph.

**Example:** If we consider the graphs in figure 3.1(a), the graph has four vertices(A, B, C, D) and eight edges. All the vertices of the graph are of degree 4, which is even. Thus the graph is an Euler graph.




(a) (b)

Fig 3.2: Examples of non-Euler Graph

Similarly, the graph in figure 3.1(b) has eight vertices (A, B, C, D, E, F, G, H). The vertices A, B, C, and D are of degree 4; and the vertices E, E, G, and H are of degree 2. Thus, all the vertices in this graph are also of even degree. Thus, the graph is an Euler graph.

Now, if we consider the graphs in figure 3.2, there exists at least one vertex in each graph which is of odd degree. For example, in the graph of figure 3.2 (a), the degree of vertices A and B is 3. Similarly, in graph 3.2 (b), the vertices C and D are of degree 3. Thus, the graphs are not Euler graphs.

**Königsberg bridge problem:** The famous Königsberg bridge problem stated that whether it is possible to cross the seven bridges, connecting the two islands of the city Königsberg, exactly once in a single traversal. The additional requirement of the problem was that

***Space for learners:***

the traversal must end at the same point from where it started. The problem may be represented graphically as in figure 3.4(a). An equivalent representation of the same problem in terms of a graph is given in figure 3.4(b).

As it can be seen from figure 3.4 (b) that the vertices of the graph are not of even degree. Hence, it is not an Euler graph and a walk that starts and ends at the same point, by crossing each edge of the graph exactly once, is not possible.

Fig 3.4: Graphical representation of Königsberg bridge problem

***Theorem 3.2:*** There exist exactly $k$ edge-disjoint subgraphs, in a graph $G$ with exactly $2k$ odd degree vertices, such that each subgraph is a unicursal graph and all the subgraphs together include all the edges of $G$.

***Proof:*** Let the odd degree vertices in $G$ be ($v_1, v_2, v_3, \ldots\ldots, v_k; u_1, u_2, u_3\ldots\ldots, u_k$). Now, let's add $k$ edges ($e_1, e_2, e_3, \ldots\ldots e_k$) in between a pair of vertices ($v_1, u_1$), ($v_2, u_2$), ($v_3, u_3$)….. and ($v_k, u_k$). This results in a new graph $G'$ where each vertex is of even degree. Thus, $G'$ is an Euler graph.

Let, **ρ** be an Euler line in $G'$. If we now deleting $e_1$from **ρ** will result in a unicursal line **γ**. Deleting $e_2$ from that from **γ**will split it into two unicursal lines **γ**1 and **γ**'; removal of $e_3$ from whichever unicursal line that it belongs to, will split that line again into two more unicursal lines resulting in a total 3 unicursal lines. Continuing this process, until we delete all other remaining extra $k$-$3$ edges, i.e., $e_4, e_5, \ldots\ldots e_k$ will finally result in $k$ unicursal lines. As we removed

***Space for learners:***

only the extra edges that we had added to *G*, all the unicursal lines together will still contain all the original edges of *G*.

***Theorem 3.3:*** For a connected graph to be an Euler Graph, if and only if it contains edge-disjoint circuits.

***Proof:***

***<u>If Part:</u>***

Let's *G* is a connected graph containing circuits. All the circuits in *G* are edge-disjoint and thus *G* can be decomposed into circuits. As in a circuit, all the vertices are of degree 2, it can be concluded that all the vertices in *G* have even degree. Hence, *G* is an Euler graph.

***<u>Only if path:</u>***

Let *G* is a Euler graph. Now let's randomly take any vertex $v_1$ from *G*, it must be involved with at least two edges as it is of even degree. Let one of these edges be ($v_1$, $v_2$) incident on the vertex $v_2$. Due to the same reason, $v_2$ must also be part of at least two edges. Let ($v_2$, $v_3$) be an edge connecting the vertices $v_2$ and $v_3$. If we continue the process, it will end only when we reach the starting vertex $v_1$ resulting in a circuit *C*. Now, removing *C* from *G* will result in a subgraph *G'* where all the vertices are of even degree. Thus, we can repeat the same process in *G'* and remove another circuit from it. This we can continue until we get a Null graph.

***<u>Space for learners:</u>***

## 3.3.3 Arbitrarily Traceable Graphs

***Space for learners:***

In an Euler graph, starting from any vertex *v*, if we start tracing the edges in such a way that no edge is repeated, it may not always result in an Euler line. For example, consider the graph in figure 3.5. If we now start from vertex A and start tracing the edges in the sequence (A, C, B, A); we will get back to the vertex A ,after which we don't have any option to visit a new edge. However, the sequence (A, B, C, A) is a circuit, not an Euler line as it does not cover all the edges of the graph. On the contrary, if we choose the starting vertex as C, and take a walk by visiting a new edge every time, we are guaranteed to get an Euler line, does not matter in what sequence we visit the edges. One such sequence is (C, E, D, C, B, A, C). From this example, it is clear that in an Euler graph, starting from any random vertex *v,* if we take an arbitrary walk by simply visiting a new edge every time, we may not get an Euler line.

For any vertex *u* in an Euler Graph *G*, if it is always possible to start from that vertex and then take an arbitrary walk by randomly selecting a new edge every time, and get back to *u* by traversing all the edges in *G*, then *G* is said to be ***arbitrarily traceable*** with respect to *u*.

***Theorem 3.4:*** A Euler graph *G* is arbitrarily traceable with respect to a vertex *v*, if and only if *v* is a part of every circuit in *G*.

***Proof*:** Let the Eulerian graph *G* can be traced arbitrarily from a vertex *v*. Assume that circuit *C* does not pass through*v*. Let *H* be a

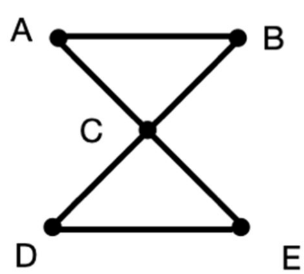



Fig 3.5: Arbitrarily Traceable Graph with respect to Vertex 'C'

subgraph of *G* that does not contain the edges of *C*. As *G* is Euler graph all its vertices are of even degree and $C$ being a circuit all its vertices are also of degree 2. Therefore, all the vertices in $H$ also have an even degree meaning that it's an Euler graph. So, in $H$ if we start from $v$, then it is possible to traverse all the edges of $H$ exactly once and then come back to $v$. Now, according to our initial assumption, as $C$ does not contain $v$, this walk cannot be extended to contain the edges of $C$.

**Example:** The graph in figure 3.5 contains two circuits $C1$(A, B, C, A) and $C2$( C, D, E, C). As we can see that C is the only vertex that is common in both the circuits, the graph is arbitrarily traceable with respect to C only. For the other vertices namely- A, B, D, and E we may not always get an Euler line by randomly walking through a new edge every time.

**CHECK YOUR PROGRESS**

1. Fill in the blanks
   a. A connected graph is an Euler graph if and only if all its vertices are of ______ degree.
   b. In a graph $G$, with 6 odd degree vertices, there exist at least _____ subgraphs such that each subgraph is a unicursal graph and all the subgraphs together include all the edges of $G$.
2. State true or false:
   a. An Euler graph is arbitrarily traceable with respect to any vertex in the graph.
   b. In a graph $G$, the sum of the degrees of vertices is18. $G$ is an Euler graph.
   c. A graph where $G$ all the vertices are of degree 6. The graph is an Euler graph.

## 3.4 HAMILTONIAN GRAPHS

Sir William Hamilton, an Irish mathematician (1805-1865), created the Icosian game, wherea dodecahedron was used with each of the 20 vertices labelled with the name of a different capital city across the world. The objective of the game was to create a closed walkacross all the cities using the edges of the dodecahedron that

***Space for learners:***

visited each city precisely once, beginning and finishing in the same city. The term -"Hamiltonian Graph" originated from this problem and became one of the most important and interesting concepts in Graph Theory.

Space for learners:

### 3.4.1 Definitions

- **HAMILTONIAN CIRCUITS**: In a graph *G*, if there exists a circuit that passes through all the vertices of *G* exactly once, then the circuit is called a Hamiltonian Circuit. If *G* contains *n* vertices, then a Hamiltonian circuit of *G* will always contain exactly *n* edges.
- **Hamiltonian Graph:** A graph that possesses a Hamiltonian circuit is called a Hamiltonian graph.
- **HAMILTONIAN PATHS :** If there exists apathin a graph *G*, such that it starts with a vertex *v* and ends with vertex *u*; containing all the vertices of *G* exactly once, then that path is called Hamiltonian path. Dropping an edge from a Hamiltonian circuit results in a Hamiltonian path. In a Hamiltonian graph *G*, each Hamiltonian path contains exactly *n-1* edges.

**Example:** Consider the graph in figure 3.6(a). It contains five vertices and 8 edges. The graph is Hamiltonian and a possible Hamiltonian circuit is shown in figure 3.6 (b). The figure in 3.6(c)

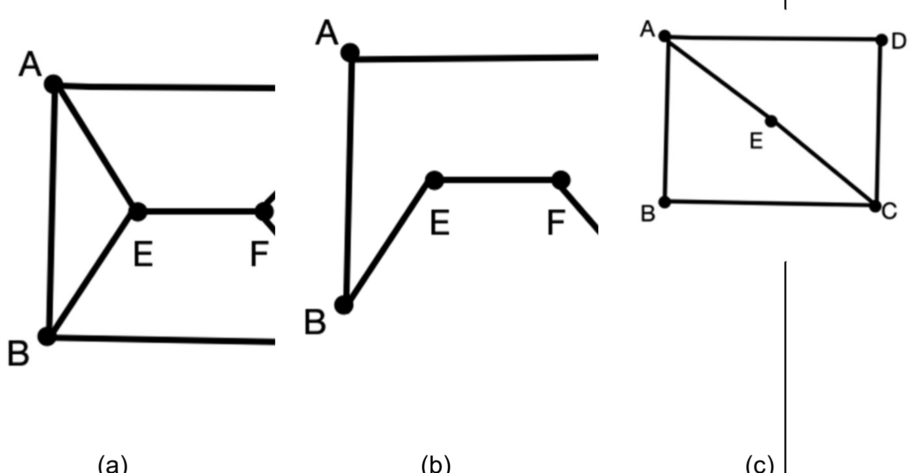



Fig 3.7: Example of Hamiltonian graph and non-Hamiltonian Graph

presents a possible Hamiltonian path for the graph.

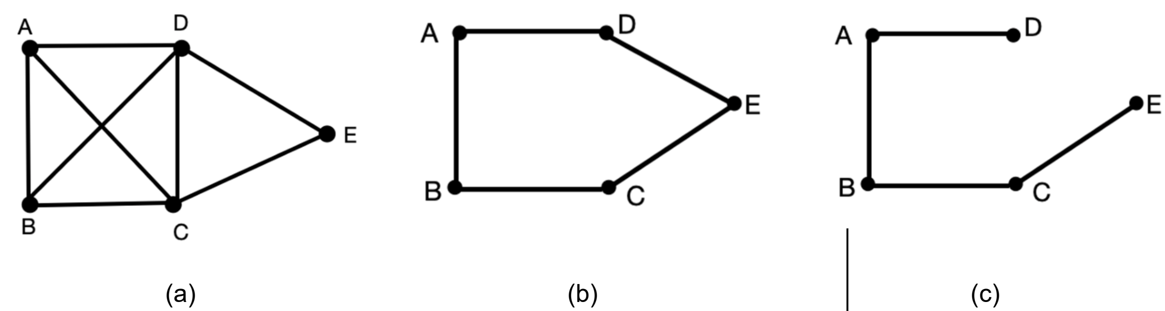



Fig 3.6: Examples of Hamiltonian graph, Hamiltonian circuit and Hamiltonian path

We may observe that the Hamiltonian Circuit contains all the 5 vertices of the original graph and has exactly 5 edges. The Hamiltonian path on the other hand contains exactly 4 edges and 5 vertices.

## 3.4.2 Theorems on Hamiltonian Circuits

***Theorem 3.5 (Dirac's Theorem):*** In a simple graph $G$, with $n$ vertices ($n\geq 3$), if the degree of each vertex is greater than or equal to $n/2$, then $G$ is a Hamiltonian graph.

**Example:** The graph in figure 3.7(a) has 6 vertices. Each vertex in the graph has degree 3≥ (6/2). Thus the graph is Hamiltonian. Figure 3.7(b) presents a Hamiltonian circuit for the same. Now, if we consider the graph in figure 3.7 (c) it has 5 vertices. The degree of the vertex E is 2 which is less than 5/2. Thus this graph is not Hamiltonian.

***Theorem 3.6 (Oreo's theorem):*** If in a simple Graph $G$ with $n$ vertices, where $n \geq 2$, for each pair of non-adjacent vertices $u$ and $v$, degree($u$)+degree($v$)$\geq n$, then the graph $G$ is a Hamiltonian graph.

**Example:** Consider the graph in figure 3.6 (a). The pairs of non-adjacent vertices in this graph are- (A, C), (A, F), (B, F), (B, D), (C, E), and (D, E). All the pair of vertices has a sum of degrees equal to 6≥6. Thus the graph is Hamiltonian.

For the graph in figure 3.7(c), (E, D) is a non-adjacent pair of vertices. The sump of degrees of the vertices E and D is 4 which is less than 5. Thus we can claim that the graph is not a Hamiltonian graph.

**CHECK YOUR PROGRESS**

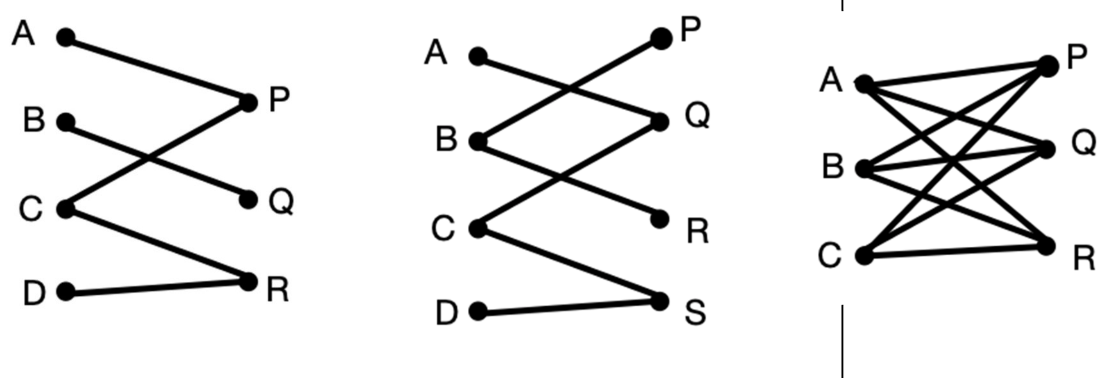
3. Fill in the blanks
   a. A Hamiltonian path traverse each vertex of the graph exactly______ .
   b. A Hamiltonian path for a Hamiltonian graph with 6 vertices has exactly _____ edges.
4. State true or false:
   a. A graph *G* has 6 vertices with degrees 2, 2, 4, 1, 3, 3 and 3. The graph is a Hamiltonian graph.
   b. A Hamiltonian circuit contains all the edges of the graph.

## 3.5 BIPARTITE GRAPHS

In graph theory, a graph *G*=(V, E) is said to be a bipartite graph if, the set of vertices V can be divided into two disjoint sets V1 and V2 such that each edge *e* belonging to E, connects a pair of vertices (*u, v*) such that $u \in$ V1 and $v \in$ V2. In other words, there does not exist any edge in *G* that connects vertices of the same set.

***Space for learners:***

***Space for learners:***




(a) (b) (c)

Fig 3.8: Examples of (a) Bipartite graph, (b) Balanced Bipartite graph and (c) Complete Bipartite Graph

**Example:** The graphs in figure 3.8 are bipartite. In all the graphs the set of vertices can be divided into two disjoint sets and none of the edges connects vertices from the same set. In graph 3.8(a) , the bipartition of the vertex set is V1={A,B,C,D} and V2={P, Q,R}. As we may see that there no edge connecting two vertices from V1 or V2. For the graph in figure 3.8(b), the two disjoint sets of vertices are V1={A, B, C, D} and V2={P, Q, R, S}. In the last graph in figure 3.8(c) the bipartition of the vertices is V1={A, B, C} and V2={P, Q, R}.

Following are some terms related to bipartite graph-

- **Balanced Bipartite Graph:** If the two sets V1 and V2 have the same number of vertices then, the graph *G* is called a balanced bipartite graph.
- **Complete Bipartite Graph:** A bipartite graph, *G* is referred to as a complete bipartite graph if each vertex in one set is connected to every vertex in the other set. In other words for each vertex belong to V1, there exists an edge to each *v* belonging to V2 and vice versa. A complete bipartite graph is denoted by $K_{m,n}$ where m and n are the cardinalities (number of vertices) of set V1 and V2 respectively.

**Example:** The graph in figure 3.8(b) is a balanced bipartite graph as the bipartition of the graph V1 and V2 contains an equal number of vertices. The graph in figure 3.8(c) is an example of a complete bipartite graph as each vertex in V1 is connected to all the vertices in V2.

Space for learners:

### 3.5.1 Properties of Bipartite Graph

***Lemma 3.1:*** In a bipartite graph *G* with the vertex petitions sets V1 and V2, $\sum_{u \in V1} deg(u) = \sum_{v \in V2} deg(v)$

***Proof:*** Let's *G'* be a subgraph of *G* that contains only the vertices of *G* and doesn't contain contain any edge. Hence, initially for this subgraph, the degree of each vertex is zero. As *G* is a bipartite graph, each edge connects a vertex in V1 and to a vertex in V2. Thus, if we now add an edge of *G* to *G'*, say between the vertex $u \in$ V1 and $v \in$ V2, this will increase the sums of degrees of the vertices in both sets to 1. Thus, $\sum_{u \in V1} deg(u) = \sum_{v \in V2} deg(v) = 1$ after adding a single edge. Adding the second edge will connect another vertex from V1 to a vertex in V2. This will further increase the sums of the degrees of the vertices by 1. Thus, after adding the second edge, $\sum_{u \in V1} deg(u) = \sum_{v \in V2} deg(v) = 2$. Since, every edge contributes exactly one to the sum of the degrees of vertices in each side, continuing the process until we add all the edges of *G* will still maintain the equality.

**Example:** For the graph in figure 3.8(a), the bipartition of the vertices are V1={A, B, C, D} and V2={P, Q, R}. The sum of degrees of vertices in V1 =deg(A)+deg(B)+deg(C)+deg(D)=1+1+2+1=5. The sum of degrees of vertices in V2 = deg(P)+deg(Q)+deg(R)=2+1+2=5. Thus we may see that the sum of degrees of vertices in both the sets is the same. We can establish the same for the other two graphs in figure 3.8.

***Theorem 3.7:*** If *G* is a *k*-regular (*k*>0) bipartite graph with bipartition V1 and V2, then |V1|=|V2|, i.e. number of vertices in V1 must be equal to number of vertices in V2.

***Proof:*** A graph is *k*-regular if all the vertices in the graph are of degree *k*. As, *G* is a *k*-regular bipartite graph, all the vertices in *G* are of equal degree, i.e, *k*. Thus,

$$\sum_{u \in V1} deg(u) = k|V1| \text{ and } \sum_{v \in V2} deg(v) = k|V2|.$$

Form the Lemma 3.1-

$$\sum_{u \in V1} deg(u) = \sum_{v \in V2} deg(v).$$

This implies that,

$$k|V1| = k|V2| \Rightarrow |V1|=|V2|$$

**Example:** The bipartite graph in figure 3.8(c) is a 3-regular graph, as all the vertices in that graph are of degree 3. The bipartition of the graph is V1={A, B, C} and V2={P, Q, R}. As we may see that both the partitions have an equal number of vertices, i.e. 3.

***Theorem 3.8:*** All the circuits in a bipartite graph are of even length.

***Proof:***Let $G$ be a bipartitely graph with set of vertices partitioned into V1 and V2. Let, $C$=($u$, $v_1$,$v_2$….$v_{2k}$, $u$) be a circuit in $G$ with an odd length $2k+1$. Let the vertex $u \in$ V1. As $G$ is bipartite, $u$ must be connected to a vertex in V2.

Now, starting from $u$, following the sequence in $C$, the first edge connects $u$ to $v_1$. Thus$v_1$ must belong to V2. The second edge connects vertex $v_1$ with $v_2$. Using the same argument we can say that $v_2 \in$ V1. If we continue following the sequence, the $2k^{th}$ edge in $C$ connects the vertex $v_{2k-1} \in$ V2 and $v_{2k} \in$ V1. The final edge $2k+1$ connects $v_{2k} \in$ V1 to $u \in$ V2, which contradicts our initial assumption that $u \in$ V1. Thus, we can conclude that all the circuits in a bipartite graph must be of even length.

***Theorem 3.9:*** Every subgraph of a bipartite graph is a bipartite graph,

***Proof:*** Let $G$ be a bipartite graph with the set of vertices partitioned into V1 and V2. Let $G'$ be a valid subgraph of $G$. Then let V1'=V1 ∩ $G'$ and V2'=V2 ∩ $G'$. If (V1', V2') is an invalid bipartition of $G'$ then there must exist an edge that connects the vertices $u$ and $v$ such that $u$, $v \in$ V1' or $u$,$v \in$ V2'. However, as $G'$ is a valid subgraph of $G$ it must not contain any edge that is not in $G$. Thus, the edge ($u$, $v$) is not a valid edge which implies that V1' and V2' is a valid bipartition. So, G' is also a bipartite graph.

**Space for learners:**

***Theorem 3.10:*** A bipartite graph with at least one edge is *2*-colourable.

***Proof:*** Let *G* be a bipartite graph with a set of vertices partitioned into V1 and V2. None of the vertices in V1 are adjacent to each other. Thus, they can be coloured with 1-colour, say colour-1. The same is true for the vertices in set V2 and all the vertices in set V2 are coloured with the same colour, colour-2.

Now, we will have to prove that colour-1 and colour -2 cannot be the same. As there exist a positive number of edges in the graph, there is at least one edge in the bipartite graph that connects a vertex *u* in set V1 to a vertex *v* in V2. This implies that *u* and *v* are adjacent to each other and thus they cannot be coloured with the same colour.

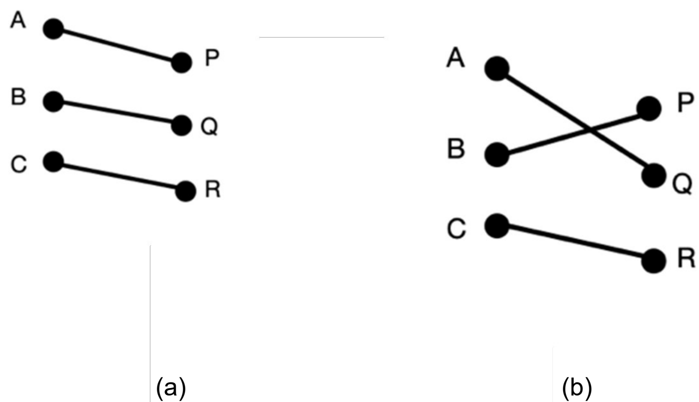



***Fig 3.10: Example of Matching***

***Figure 3.9: Example of colouring in a bipartite graph***

Thus colour -1 and colour-2 must be different.

**Example:** Consider the graph in figure 3.8(a). It can be coloured using *2*-colours as shown in figure 3.9. In this case, we have used red and green colours. As we may observe that none of the adjacent vertices are coloured with the same colour. The red colour has been used for the vertices A, B, C, and D, which are not adjacent to each other. The green colour has been used for the vertices P, Q, and R; none of which are adjacent to each other. Thus with two colours, we can properly colour the graph. The same can be shown for the other two graphs in figure 3.8.

***Space for learners:***

| CHECK YOUR PROGRESS |
|---|
| 5. Fill in the blanks<br>a. A bipartite graph can be coloured using ___ colours .<br>b. In a bipartite graph, the sum of degrees of vertices in one set is 8. The same of the other set is____.<br>6. State true or false:<br>a. A 6- regular bipartite graph contains equal number of vertices in both the set of bipartition.<br>b. The circuits of a bipartite graph can be of odd as well as of even length. |

***Space for learners:***

### 3.5.2 Matching in Bipartite Graph

In a graph *G*, a matching *M* is a subgraph, with a set of edges such that no two edges have a common vertex. Thus, in a matching each vertex has degree exactly 1.

A matching *M* is said to be maximal if it contains the largest possible number of edges from *G*. A perfect matching is a maximal matching that contains all the vertices of *G*.

The graph in figure 3.10(a) is a matching for graph 3.8(a). It is also the maximal matching for the graph. However, it is not a perfect matching as it does not include the vertex D. On the other hand, the graph in figure 3.10(b) is an example of matching for the graph in figure 3.8(c). It is a perfect matching as it includes all the vertices of the graph.

***Lemma 3.2:*** In bipartite graph *G*, with bipartition V1 and V2, there does not exist a perfect match if $|V1| \neq |V2|$

***Proof:*** Suppose, there exists a perfect matching *M* for *G*. Now, let's construct a subgraph $G'$ which contains all the vertices of *G* and the edges of *M*. According to theorem 3.9, $G'$ is also a bipartite graph. Since $G'$ contains the edges of *M*, all the vertices in $G'$ are of degree 1. Thus, $G'$ is a 1-regular bipartite graph, and applying the theorem 3.7, we can conclude that $|V1|=|V2|$.

**Example:** It is not possible to have a perfect match for the graph in figure 3.8(a), as the bipartition does not contain an equal number of vertices. On the contrary, if we examine the graph in figure 3.8(c), it contains an equal number of vertices in both sets. Thus, a perfect match is possible in this graph. The graph in figure 3.10(b) is an example of a perfect match for this graph.

***Space for learners:***

## 3.6 SUMMING UP

- In this module we have discussed - Euler graph, Hamiltonian graph, and Bipartite graph.
- An Euler line is a closed walk that contains all the edges of the graph exactly once. A graph containing an Euler line is an Euler graph.
- A graph is an Euler graph if and only if all the vertices are of even degree.
- A unicursal line is an open Euler line.
- A connected graph is an Euler graph if and only f it contains edge-disjoint circuits.
- An Euler graph is arbitrarily traceable with respect to a vertex $v$ , if $v$ is a part of every circuit in the graph.
- A Hamiltonian circuit contains all the vertices of a graph exactly once. A graph containing a Hamiltonian circuit is called a Hamiltonian graph.
- Using Dirac's theorem and Oreo's theorem we can check if a graph is Hamiltonian or not.
- A bipartite graph, $G$, is a graph, where the vertices can be partitioned into two disjoint sets such that no edge of $G$ connects two vertices from the same set.
- The subgraph of a bipartite graph is also a bipartite graph.
- A bipartite graph is $2$-colourable.
- In a bipartite graph, a perfect match exists if both set of vertices have equal cardinality.

## 3.7 ANSWERS TO CHECK YOUR PROGRESS

1.
   a. Even

Space for learners:

b. 3

2.

a. False
b. Flase.
c. True.

3.

a. once
b. 5

4.

a. False.
b. False.

5.

a. 2
b. 8

6.

a. True
b. False

## 3.8 POSSIBLE QUESTIONS

1. **Short Answer Type Questions:**
   a. Define Hamiltonian graph. List some of its applications.
   b. Define Euler graph and list some its applications.
   c. For a graph to be arbitrarily traceable with respect to a vertex *v*, what constraint *v* must satisfy?
   d. Define unicursal line. Why is it also called an open Euler line?
   e. What is a complete bipartite graph. Give an example.
2. **Long Answer Type Questions:**
   a. State the Königsberg bridge problem and illustrate Euler's solution to this problem.
   b. What is a matching? Explain with an examples the concept of perfect matching. Prove that perfect matching is not possible in a bipartite graph having different number of vertices in the bipartition.

c. Discuss the Dirac's theorem and Oreo's theorem for Hamiltonian gram with the help of examples.

d. State some applications of Bipartite graph. Prove that a bipartite graph is 2-colourable.

e. Prove that a subgraph of a bipartite graph is also a bipartite graph.

Space for learners:

## 3.9 REFERENCES AND SUGGESTED READINGS


- M.E Van Valkenburg, "Network Analysis", Prentice Hall, 2006.
- Abhijit Chakrabarti, "Circuit Theory (Analysis and Synthesis)", Dhanpat Rai & Co, 7th Edition, 2018.
- C. K Alexander and M.N.O Sadiku, "Electric Circuits", McGraw Hill Education, 2004

# UNIT 4: TREES

Space for learners:

**Unit Structure:**



## 4.1 INTRODUCTION

A tree is a nonlinear discrete data structure. This unit gives an overview of the tree and its properties. The types of trees such as rooted and binary trees are also discussed in this unit. A binary tree has a maximum of two leaf nodes. The counting tree with its properties is also reported in the unit. The concept of a circuit along with the minimum spanning tree is also discussed in the unit. A minimum spanning tree contains the minimum weight of the graph. The graph cut set and the weighted graph are also discussed in this unit.

## 4.2 UNIT OBJECTIVES

After going through this unit, you will be able to know

- About trees and their properties.
- About the rooted tree, counting tree, and binary trees.

- About the circuit and weighted graph.
- About the spanning tree.

Space for learners:

## 4.3 TREES AND THEIR PROPERTIES

The tree is a discrete nonlinear structure that represents hierarchical relationships between nodes. It is a connected and acyclic undirected graph. There is a path between the nodes of a tree. A tree with n nodes contains (n-1) numbers of edges. Every node has a degree. The node which has o degree, known as the root of the tree. The node with degrees 1 and 2 is known as the leaf and internal node of the tree.

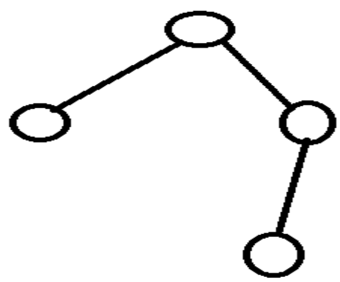


Fig. 4.1: Example of a tree

The properties of a tree are explained below:

i) A tree is a nonlinear data structure.

ii) Every tree has a root node with degree 2.

iii) The degree of a leaf node is 1.

iv) The degree of an internal node is at most 2.

v) Every tree has n-1 numbers of edges.

## 4.4 DISTANCE AND CENTER OF TREES

The vertex with the minimal eccentricity of a tree is known as the center of the tree. The eccentricity of a vertex is the maximum distance from that respective vertex to other vertexes in the tree and it is the diameter of the tree. Some trees may contain only one center and this type of tree is known as a central tree. Some trees may contain more than one tree and this type of tree is known as a bi-central tree.

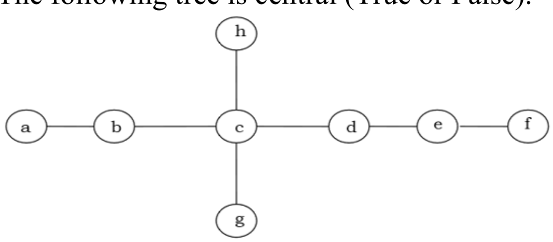
To understand the Center of a tree, let's consider the following tree.


In the above tree, five nodes are present. Initially, a node with degree 1 and its adjacent edges should remove from the tree to find the center of the tree. So, you remove node a and e as both nodes have degree 0. After removing a and e along with its adjacent edges, the resultant graph will be as follows.


Then apply the same procedure on the graph and remove b and d from the graph, Final the graph contains only one vertex and that is C. So, it is a central tree.

**CHECK YOUR PROGRESS - I**

1. What is degree of a tree?
2. What is te degree of the root node?
3. How many edges are there in a tree of n nodes?
4. The following tree is central (True or False).




## 4.5ROOTED AND BINARY TREE

A rooted tree is a connected and acyclic graph. The tree has a special node known as the root of the tree and each node is directly or indirectly connected to the root of the tree where children of the internal nodes are ordered. The internal node of the

Space for learners:

rooted tree may have fewer or exactly m children. The rooted tree in which m =2, is known as the binary tree.

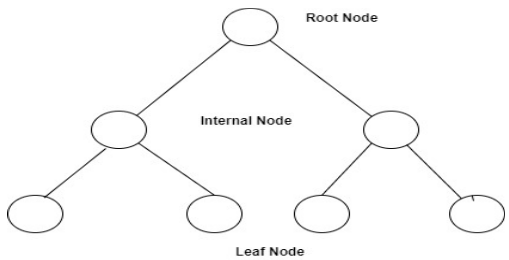



Fig. 4.2 Rooted tree

The Binary tree is that rooted that has a maximum of two children. It means that each node can have either 0, 1, or 2 children.

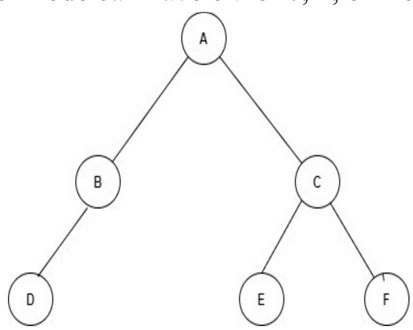



Fig. 4.3 Rooted tree

In Fig. 4.3, the root node is A and it has a maximum of 2 children, i.e., B and C. Node B has only one child and node C have two children. The leaf nodes D, E, and F have no children. So, it is a binary tree.

The properties of the binary tree are presented below.

i) The maximum number of nodes in a level "i" is $2^{i.}$

ii) The height of the tree is the longest path from the root node to the leaf node.

iii) The minimum number of nodes at height h is h+1.

iv) The maximum of nodes in a binary tree of height h is $2^{(h+1)}$-1

v) The height and number of nodes of the binary tree are inversely related.

The height of a binary tree with n nodes can be calculated as follows.

As you know that $n = 2^{h+1} - 1$

Space for learners:

Space for learners:

⇨ $n+1 = 2^{h+1}$

⇨ Now Taking log on both the sides

⇨ $\log_2(n+1) = \log2(2^{h+1})$

⇨ $\log_2(n+1) = h+1$

⇨ $h = \log_2(n+1) - 1$

Depending on the number of children of a node, the binary tree is further divided into the following categories.

i) Full or Strict Binary Tree:

The full or strict binary tree is one that each node must have either 0 or 2 children. The full binary tree can also be defined as the tree in which each node must contain 2 children except the leaf nodes.

ii) Complete Binary Tree:

The complete binary tree is one where all the nodes In a complete binary tree, the nodes should be added from the left.

iii) Perfect Binary Tree:

A perfect binary tree is one where all the internal nodes have 2 children, and all the leaf nodes are at the same level.

iv) Degenerate Binary Tree:

In this binary tree, the internal nodes have only one child.

v) Balanced Binary Tree:

The balanced binary tree is one where the left and right subtree differ by at most 1. For example, AVL and Red-Black trees.

## 4.6 COUNTING BINARY TREE

Let's have a binary tree. How do you count the number of nodes and the nodes have two children (two children) without or without using recursion.

i) Let's you have a binary tree, and you can count all the nodes in the binary tree using the following approach

a. Do post-order traversal of the tree.

b. If the root is null, then perform return 0.

c. If the root is not null then you can make a recursive call to the left child and right child. The result of these with 1 will be return

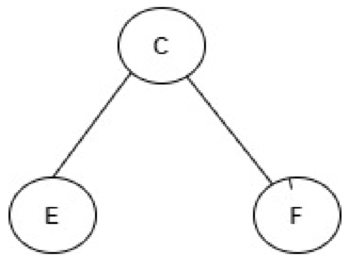



Fig. 4.4Binary tree

In Fig. 4.4, the number of nodes is 3.

The number of nodes that have both children or null can be count using the following approach.

1) Create an empty Queue and push the root node to Queue.
2) Do following while Queue is not empty.
   a. Pop an item from Queue and process it.
      i) If it is a full node then increment the counter.
   b. Push left child of the popped item to Queue, if available.
   c. Push the right child of the popped item to Queue, if available.

## 4.7 FUNDAMENTAL CIRCUITS

The fundamental circuit is related to the spanning tree. Let you have a connected graph G and T be a spanning tree. Then a circuit formed by adding a chord T in the spanning tree T is known as a fundamental circuit.

To understand it, lets you have a graph G. Now, form a spanning tree T from the graph. A spanning tree is a that tree which contains all vertices of the graph without any cycle.

*Space for learners:*

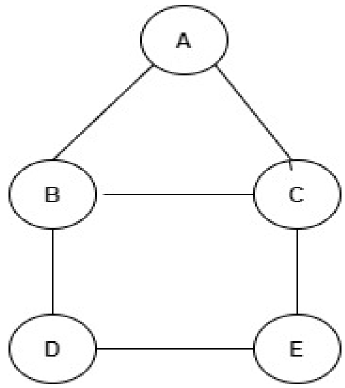


Fig. 4.5Example of Graph

Now, the spanning of the graph is

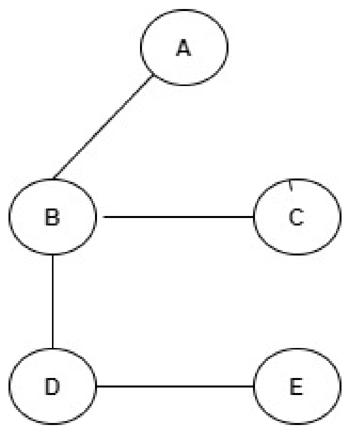


Fig. 4.6 Example of Spanning Tree

Now, you have to find the branch and chord set from the spanning tree. The branch set is that set that contains the edges of the spanning tree. The chord set is that one which does not present in the spanning tree.

So, the branch set = {AB, BC, BD, DE}

The chord set = {AC, CE}

Now, if you add AC in the spanning tree, then it will form a circuit (AB, BC, AC). So it is known as a fundamental circuit. Again if we add CE, then it will create another fundamental circuit (BC, CE, ED, DB).

Space for learners:

**CHECK YOUR PROGRESS- II**

5. What is rooted tree?
6. What is the maximum number of node associated with a height h?
7. What is the height of a binary tree with n number of nodes?
8. Is perfect and full binary is same?
9. Can you form a fundamental ciruit from a spanning tree?
10. What is chord and branch set?

***Space for learners:***

## 4.8 SPANNING-TREE IN WEIGHTED GRAPHS

A spanning tree contains all vertices of a graph without having any cycles. A spanning tree cannot be disconnected.

So, you can say that every connected and undirected graph has atleast one spanning tree. Let's you have the following graph.

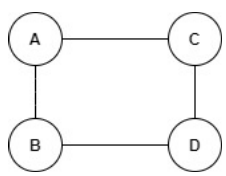



Fig. 4.7Example of Graph

In the graph, four vertices (A, B, C, and D) are present. From the graph, you can draw the following four spanning trees.

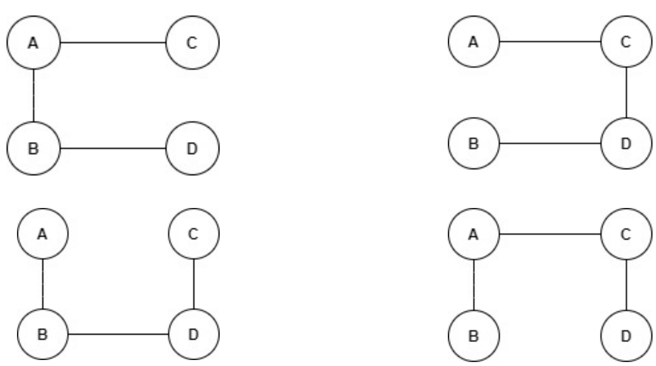



Fig. 4.8 Different Spanning Tree of Fig. 4.7

The above is graph is not complete. So in this graph, you can apply the Kirchhoff theorem to count the number of spanning trees. But if you have a complete undirected graph, you can count the number of spanning-tree using the formula $n^{n-2}$. Let's consider the following graph.

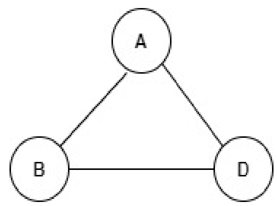



Fig. 4.9Example of Graph

The above graph is completely undirected. So, you can draw $n^{n-2} = 3^{3-2} = 3$. The spanning trees are as follows.

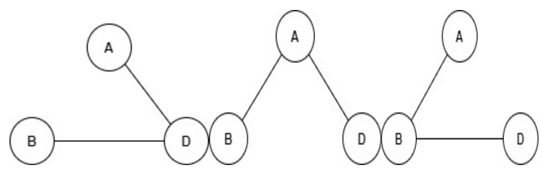



Fig. 4.10Different Spanning Tree of Fig. 4.9

The properties of a Spanning tree are presented below.

i) A connected graph can have more than one spanning tree.

ii) All possible spanning trees must have many edges and vertices.

iii) A spanning tree does not have a closed circuit.

iv) A spanning will be disconnected after removing one edge.

v) The addition of an extra edge in the spanning tree creates the fundamental circuit.

A minimum spanning tree contains the minimum cost of the graph. A minimum spanning tree can be found in a weighted graph. A weighted graph must have one weight associated with each edge. When you create spanning trees from this type of graph, one spanning must have minimum weight, which is known as a minimum spanning tree (MST).

You can find the minimum spanning tree from a graph using the Kruskal Algorithm.

*Space for learners:*

Let's you have the following graph.

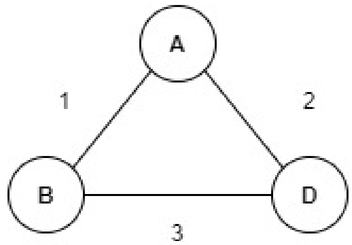



Fig. 4.11Example of Graph

In the above graph, three edges are there which have edge weights 1, 2, and 3, accordingly. So, this is a weighted graph. From the above-weighted graph, you can draw the following spanning trees.

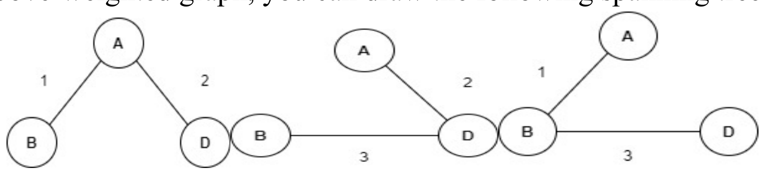



Fig. 4.12Different Spanning Tree with the weight of Fig. 4.11

In the above three spanning, the first spanning tree has the total weight = 1 + 2 = 3. The second spanning tree has the total weight = 2 + 3 = 5. The last spanning tree has the total weight = 1 + 3 =4.

So, now, you have three spanning three with three different weights. Among all, the first spanning tree has the minimum weight. So it is known as MST.

As mentioned above, you can find the MST using Kruskal's algorithm as follows.

i) Sort all the edge weight in ascending order.

ii) Consider and add one by one edge from the sorted list.

iii) Do not add an edge it creates a cycle.

If you apply the above steps in the above graph, then the execution will be as follows.

i) After sorting the edge, you will get 1, 2, 3.

ii) Now consider the first edge weight 1 and add it to the tree, as it will not create any cycle.

*Space for learners:*

iii) Then you can add edge 2, as it will also not create any cycle.

iv) Finally, consider edge weight 3. But you can not add it as it will create a cycle.

So, your spanning tree will contain only the edge weight 1 and 2. So, the MST is 3.

## 4.9 CUT SETS

Before discussing the cutsets, you first know about the cut edge and cut vertice. A cut vertice is that upon removing of which the graph will be a disconnect. Like vertice, upon removing of which edge, two or more graphs will be a disconnect, is known as the cut edge.

Let's have a graph G (V, E). A subset EE of E is called a cut set of G, if deletion of all the edges of EE from G, the G will disconnect. If deleting edges from a graph makes it disconnected, is known as cut sets.

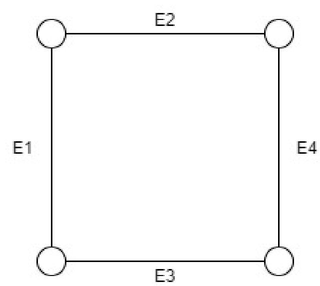



Fig. 4.13Example of Graph

In the above graph, the graph contains 4 edges i.e., {E1, E2, E3, E4}. Let the cut set = { E1, E4}. Upon removing E1, and E4 from the graph will look like two graphs (below). So, it is the cut set.

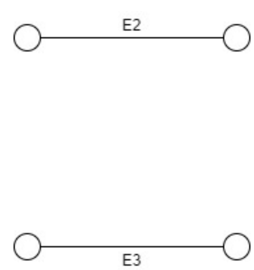



Fig. 4.14Example of Cut set

Space for learners:

Depending on the size of the cut, a cut set may be minimum, and maximum. If the size of the cut set is minimum as compared to the other cut set, then it is minimum otherwise the cut set may equal or maximum.

**CHECK YOUR PROGRESS - III**

11. What is MST?
12. Which algorithm is used to count the number of spanning tree of a graph?
13. A Spanning has a cycle (True or False).
14. What is cut vertex and cut edge?

Space for learners:

## 4.10 SUMMING UP

i) The tree is a discrete nonlinear structure that represents hierarchical relationships between nodes.

ii) A tree with n nodes contains (n-1) numbers of edges. Every node has a degree.

iii) Every tree has a root node with degree 2.

iv) The degree of an internal node is at most 2.

v) The vertex with the minimal eccentricity of a tree is known as the center of the tree.

vi) Some trees may contain only one center and this type of tree is known as a central tree. Some trees may contain more than one tree and this type of tree is known as a bi-central tree.

vii) A rooted tree is a connected and acyclic graph. The tree has a special node known as the root of the tree and each node is directly or indirectly connected to the root of the tree.

viii) The Binary tree is that rooted that hasa maximum of two children. It means that each node can have either 0, 1, or 2 children.

ix) The minimum number of nodes at height h is h+1.

x) The maximum of nodes in a binary tree of height h is 2^(h+1)-1

xi) The full or strict binary tree is one that each node must have either 0 or 2 children.

xii) A perfect binary tree is one where all the internal nodes have 2 children, and all the leaf nodes are at the same level.

xiii) The fundamental circuit is related to the spanning tree. Let you have a connected graph G and T be a spanning tree. Then a circuit formed by adding a chord T in the spanning tree T is known as a fundamental circuit.

xiv) A spanning tree contains all vertices of a graph without having any cycles.

xv) A minimum spanning tree contains the minimum cost of the graph. A minimum spanning tree can be found in a weighted graph.

xvi) A cut vertice is that upon removing of which the graph will be a disconnect. Like vertice, upon removing of which edge, two or more graphs will be a disconnect, is known as the cut edge.

Space for learners:

## 4.11 ANSWER TO CHECK YOUR PROGRESS

1) The number of edges associated with a vertex is known as the degree of a vertex.
2) 2
3) n-1
4) False
5) A rooted tree is a connected and acyclic graph. The tree has a special node known as the root of the tree and each node is directly or indirectly connected to the root of the treewhere children of the internal nodes are ordered.
6) $2^{(h+1)-}1$
7) h = log2(n+1) – 1
8) No
9) Yes

10) The branch set is that set that contains the edges of the spanning tree. The chord set is that one which does not present in the spanning tree.

11) A minimum spanning tree contains the minimum cost of the graph. A minimum spanning tree can be found in a weighted graph.

12) Kirchhoff theorem.

13) False

14) A cut vertice is that upon removing of which the graph will be a disconnect. Like vertice, upon removing of which edge, two or more graphs will be a disconnect, is known as the cut edge.

Space for learners:

## 4.12 POSSIBLE QUESTIONS

**Short answer type questions:**

i) What is a tree? What are the properties ofa tree?

ii) What is the center of the tree?

iii) What is the difference between centric and bicentric trees?

iv) What is a binary tree?

v) What are the properties of a binary tree?

vi) Show that the height of a binary tree of node n is h = log2(n+1) – 1

vii) What is the difference between a full and perfect binary tree?

viii) How do you count the number of nodes in a binary tree?

ix) What isa fundamental circuit? How do you form a fundamental circuit from a spanning tree?

x) What is MST?

xi) How many spanning trees will be formed from a connected graph with n vertices?

xii) What are the properties of a spanning tree?

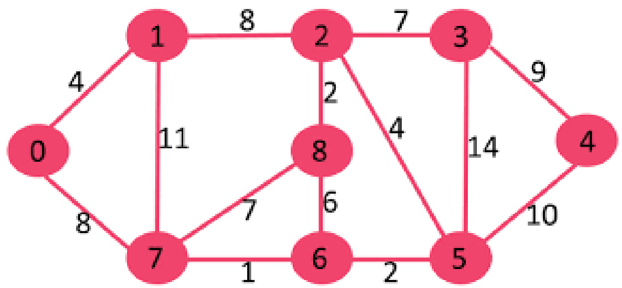
xiii) What is cur set?

**Long answer type questions:**

Space for learners:

i) Explain the MST with an example.

ii) Find the MST for the following tree




iii) Explain binary trees with their types with examples.

## 4.13 REFERENCES AND SUGGESTED READINGS

- Data Structures Using C by Reema Theraja Publisher: Oxford Publication

# UNIT 5: GRAPH REPRESENTATION

Space for learners:

**Unit Structure:**



## 5.1 INTRODUCTION

Graph theory has evolved into a powerful tool that can be used to a wide range of areas. Engineering mathematics, computer programming, networking and marketing are only a few of them. Paths generated by travelling along the edges of a graph can be used to simulate a variety of issues. Models that incorporate pathways in graphs can be used to address problems such as efficiently designing routes for parcel delivery, waste collection, and finding shortest path. Graphs may grow exceedingly complicated when faced with these types of problems, necessitating a more efficient means of expressing them in practice. The adjacency matrix and adjacency list are used to solve this problem.

## 5.2 UNIT OBJECTIVES

After going through this unit, you will be able to:

- To understand and apply the fundamental concepts in graph representations

- To explain the basic concepts on graph colouring and solving practical problems.
- To explain the basic concepts on graph covering and solving problems.

Space for learners:

## 5.3 MATRIX REPRESENTATION OF GRAPHS

In a computer, there are several methods to represent a graph. Graphs are typically depicted diagrammatically, although this is only viable when the number of vertices and edges is minimal. As a result, the notion of graph matrix representation is established.The computation of paths and cycles in graphs problems such as communication networks, power distribution, and transportation, among others, is one of the primary advantages of this representation. However, this format has the drawback of reducing the visual appeal of graphs.

## 5.4 ADJACENCY MATRIX

The most convenient way of representing any graph is the matrix representation. It is a square matrix of order (n xn) where n is the number of vertices in the graph. Generally represented by $M[a_{ij}]$ where $a_{ij}$is the $i^{th}$row, $j^{th}$ column element. The general form of adjacency matrix is given below.

$$\text{Where, } a_{ij} = \begin{cases} 1 \text{ ; if an edge in the graph between the vertex } v_i \text{ and } v_j \\ 0 \text{ ; otherwise} \end{cases}$$

$$
M = \begin{pmatrix}
a_{11} & a_{12} & a_{13} & a_{14} & \ldots\ldots & a_{1n} \\
a_{21} & a_{22} & a_{23} & a_{22} & \ldots\ldots & a_{2n} \\
a_{31} & a_{32} & a_{33} & a_{34} & \ldots\ldots & a_{3n} \\
. & . & . & . & \ldots\ldots & . \\
a_{n1} & a_{n2} & a_{n3} & a_{n4} & \ldots\ldots & a_{nn}
\end{pmatrix}
$$

The matrix is termed as adjacency matrix, because each entry in the matrix stores the information between the vertices as adjacent or not. The entry can be either 0 or 1.

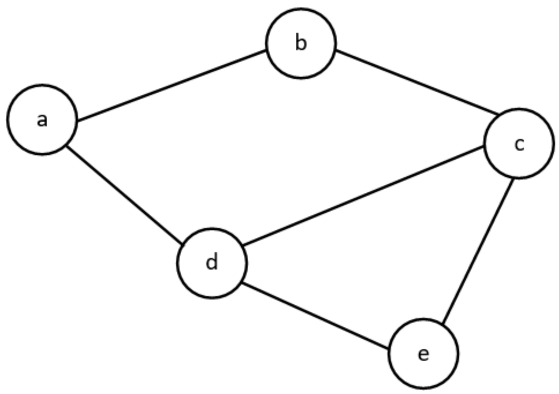



Figure 5.1: A Simple Graph

Consider the graph, G given in Figure 5.1, the adjacency matrix with respect to the vertices a, b, c, d, e is shown below. As there is an edge between vertex ‘a’ and vertex ‘b’, the corresponding position in the adjacency matrix is having the entry 1. As there is no edge between vertex ‘a’ and vertex ‘c’, the corresponding position in the adjacency matrix is having the entry 0.

$$
I = \begin{array}{c} \\ \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \end{array}
\begin{array}{c}
\begin{array}{ccccc} \mathbf{a} & \mathbf{b} & \mathbf{c} & \mathbf{d} & \mathbf{e} \end{array} \\
\begin{bmatrix}
0 & 1 & 0 & 1 & 0 \\
1 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 1 & 1
\end{bmatrix}
\end{array}
$$

***Space for learners:***

$$
\begin{array}{c}
d \\
e
\end{array}
\begin{bmatrix}
1 & 0 & 1 & 0 & 1 \\
0 & 0 & 1 & 1 & 0
\end{bmatrix}
$$

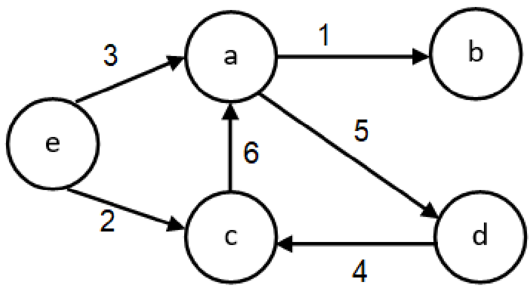



Figure 5.2: A Weighted Graph

Consider the graph, G given in Figure 5.2, the adjacency matrix with respect to the vertices a, b, c, d, e is shown below. As there is a directed edge from vertex ‘a’ to vertex ‘d’ having weight ‘5’ the corresponding position in the adjacency matrix is having the entry 5. As there is no directed edge from vertex ‘a’ to vertex ‘c’, the corresponding position in the adjacency matrix is having the entry 0.

$$
M = \begin{array}{c}
\\
a \\
b \\
c \\
d \\
e
\end{array}
\begin{array}{c}
\begin{array}{ccccc} a & b & c & d & e \end{array} \\
\begin{pmatrix}
0 & 1 & 0 & 5 & 0 \\
0 & 0 & 0 & 0 & 0 \\
6 & 0 & 0 & 0 & 0 \\
0 & 0 & 4 & 0 & 0 \\
3 & 0 & 2 & 0 & 0
\end{pmatrix}
\end{array}
$$

***Space for learners:***

**STOP TO CONSIDER**

In an adjacency matrix, if the diagonal elements are zero, then the graph is called simple graph.

In a multi graph i.e. a graph having parallel edges, adjacency matrix can be found using

$$a_{ij} = \begin{cases} n; \text{ number of edges between a pair of vertex } v_i \text{ and } v_j \\ 0; \text{ otherwise} \end{cases}$$

In a weighted graph, adjacency matrix can be found using

$$a_{ij} = \begin{cases} w; \ w \text{ is the weight of edge between the vertex between } v_i \text{ and } v_j \\ 0; \text{ otherwise} \end{cases}$$

Space for learners:

## 5.5 ADJACENCY LIST

An adjacency list is a group of unordered lists that is used to describe a finite graph. Each unordered list in an adjacency list describes a vertex's collection of neighbours in the graph. This is one of the graph representations frequently used in computer systems.

In Adjacency List, an array of a list "Adjlist[i]"is used to represent the graph.The list size is equal to the number of vertex(n).

If we assume that graph has *n* vertex, then
Adjlist[0] will have all the vertices that are connected to vertex 0.
Adjlist[1] will have all the vertices that are connected to vertex 1 and so on.

Consider the undirected graph G in Figure 5.3.

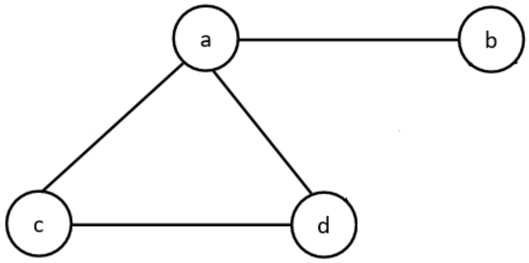
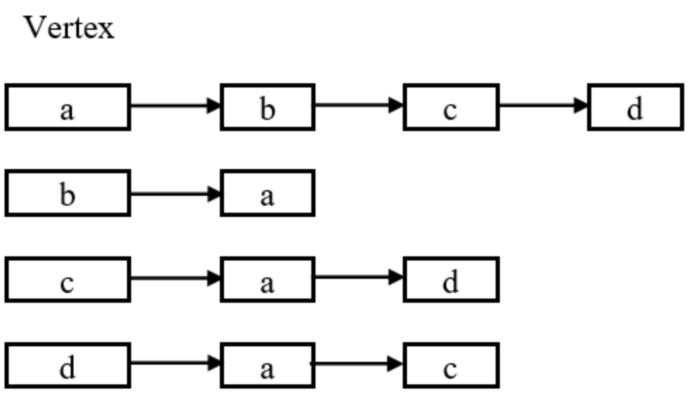



Figure 5.3: An undirected Graph

The adjacency list for the undirected graph is shown below, here the adjacency list for vertex ‘a’ are the vertices adjacent to ‘a’ that is there is an edge connecting ‘a’ with vertices ‘b’, ‘c’ and ‘d’. Similarly, adjacency list for vertex ‘c’ is ‘a’ and ‘d’, adjacency list for vertex ‘d’ is ‘a’ and ‘c’. Finally, adjacency list for vertex ‘b’ is ‘a’.




Consider the directed graph G in Figure 5.4

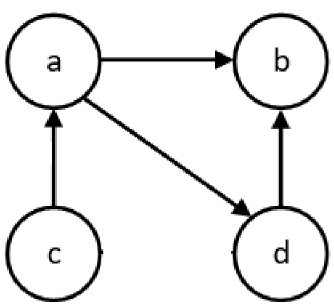
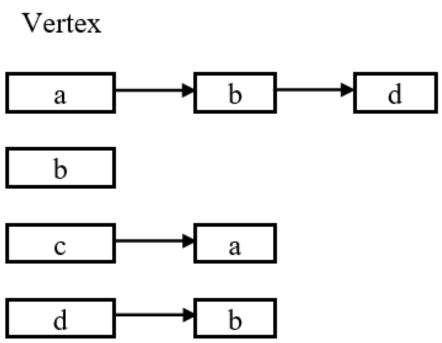



Figure 5.4: An directed Graph

***Space for learners:***

**Space for learners:**

The adjacency list for directed graph is shown below, here the adjacency list for vertex ‘a’ are the vertices adjacent to ‘a’ that is there is an outgoing edge from ‘a’ to vertex ‘b’ and ‘d’. Similarly, adjacency list for vertex ‘c’ is the outgoing edge from ‘c’ to ‘a’ and adjacency list for vertex ‘d’ is the outgoing edge from ‘d’ to ‘b’. Finally, adjacency list for vertex ‘b’ is nil, as there is no outgoing edge from vertex ‘b’.




## 5.6 INCIDENCE MATRIX

Consider a graph G with $n$ vertices and $e$ edges, then the incidence matrix $I[a_{ij}]$ is a matrix of order ($n$ x $e$) where the element $a_{ij}$, where rows corresponds to its vertices and columns correspond to its edges is defined as

$$a_{ij} = \begin{cases} 1; \text{ if vertex } i \text{ belongs to edge } j \\ 0; \text{ otherwise} \end{cases}$$

Consider the graph G in Figure 5.5

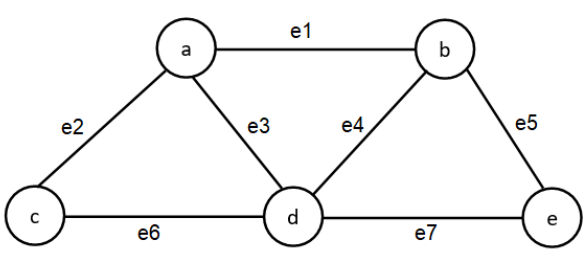


Figure 5.5: A Graph

Space for learners:

The incidence matrix with respect to the vertices a, b, c, d, e and edges e1, e2, e3, e4, e5, e6, e7 is shown below. As there is an edge incident on vertex 'a' and vertex 'b' the corresponding position in the incidence matrix is having the entry 1. As there is no edge e4, e5, e6 and e7 incident on vertex 'a', the corresponding position in the incidence matrix is having the entry 0.

$$I = \begin{array}{c} \\ \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \\ \mathbf{d} \\ \mathbf{e} \end{array} \begin{array}{c} \begin{array}{ccccccc} \mathbf{e1} & \mathbf{e2} & \mathbf{e3} & \mathbf{e4} & \mathbf{e5} & \mathbf{e6} & \mathbf{e7} \end{array} \\ \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} \end{array}$$

Similarly, the incidence matrix $I[a_{ij}]$ of a digraph G is defined as

$$a_{ij} = \begin{cases} 1; \text{ if edge } j \text{ is incident out of vertex } i \\ -1; \text{ if edge } j \text{ is incident into vertex } i \\ 0; \text{ otherwise} \end{cases}$$

Consider the graph G in Figure 5.6.

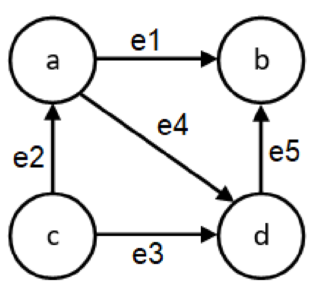



Figure 5.6: A Graph

The incidence matrix with respect to the vertices a, b, c, d and edges e1, e2, e3, e4, e5 is shown below. As there is an edge 'e1' and 'e4' incident out of vertex 'a' the corresponding position in the incidence matrix is having the entry 1. Whereas there is an edge 'e2' incident into vertex 'a', the corresponding position in the incidence matrix is having the entry -1. Finally, as neither the edge 'e3' and 'e4' is incident into or out of vertex 'a', the corresponding entry in the incidence matrix is marked as 0.

$$
I = \begin{array}{c} \\ \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \\ \mathbf{d} \end{array}
\begin{array}{c}
\begin{array}{ccccc} \mathbf{e1} & \mathbf{e2} & \mathbf{e3} & \mathbf{e4} & \mathbf{e5} \end{array} \\
\begin{bmatrix} 1 & -1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 & -1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & -1 & -1 & 1 \end{bmatrix}
\end{array}
$$

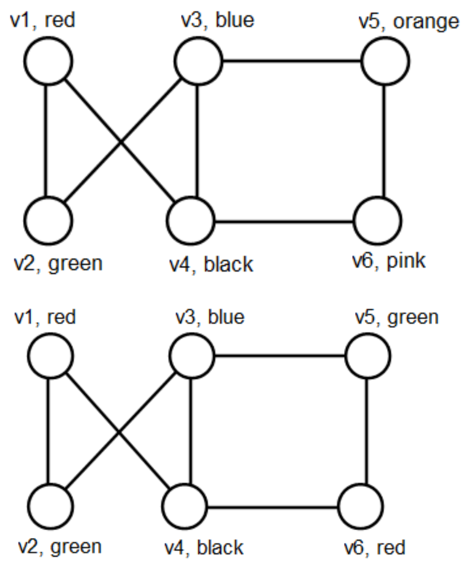
Space for learners:

## 5.7 BASIC CONCEPT OF GRAPH COLORING, COVERING AND PARTITIONING

**Graph Coloring:** Consider a graph G having n vertices. If we want to paint all the vertices such that no two adjacent vertices are of same colour then a question can be asked as to what should be the minimum number of colours required in such case? This type of problem constitutes graph colouring problem. Similarly, colouring problem in a graph can be applied also to the edges. One application of graph colouring is Map Coloring where geographical map of states where no two adjacent states can be assigned same color.

As an example in Figure 5.7 (a), assigning all the vertices with colours such that no two adjacent vertices are assigned same colour is called proper colouring. In some cases, proper colouring with minimum number of colours may be required. In Figure 5.7 (b), 4 different colours are used compared to six in Figure 5.7 (a).

The minimum number of colours required to colour a graph G is called its chromatic number. If a graph requires $k$ different colours for its proper colouring then it is known as $k$-chromatic or $k$-colourable.




(a) (b)

Figure 5.7: Proper Colouring of a Graph

**STOP TO CONSIDER**

A complete graph where each vertex is connected to every other vertex having $n$ vertices, has chromatic number $k = n$.

A cycle graph having $n$ vertices,has chromatic number $k$ = 3 if $n$ is odd or $k$ =2 if $n$ is even.

**Graph Covering:** A covering graph C is a subgraph which contains either all the vertices or all the edges corresponding to some other graph G.

**Space for learners:**

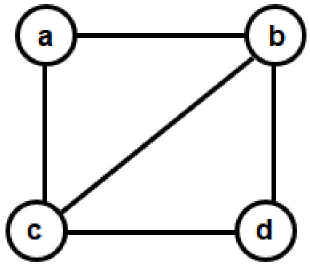


Figure 5.8: A simple Graph

A subset is called a line covering of a graph G if every vertex of G is incident with at least one edge. For example, in the graph given in Figure 5.8, subset $S_1$, $S_2$, $S_3$, $S_4$ are line covering as all the vertices are covered using the edges in each of the subset. However, subset $S_5$ is not line covering due to the fact that vertex 'c' is not covered.

$S_1$ = {(a, b), (c, d)}
$S_2$ = {(a, c), (b, d)}
$S_3$ = {(a, b), (b, d), (c, d)}
$S_4$ = {(a, b), (b, c), (b, d)}
$S_5$ = {(a, b), (b, d)}

A subset K of V is called a vertex covering of a graph G (V, E), if every edge of 'G' is incident with or covered by a vertex in 'K'. For example, in the graph given in Figure 5.8, subset $K_1$ contains vertex 'b' and 'c' which covers the edges that is 'ba', 'bc', 'bd' and 'ca', 'cb', 'cd' respectively. Thus all the edges are covered by vertex {b, c} and so is $K_1$ vertex covering. Similarly, subset $K_2$ contains vertex 'a' 'b' and 'c' which covers all the edges in the graph G. Also, subset $K_3$ contains vertex 'a' 'd' and 'c' which covers all the edges in the graph G. So, both $K_2$ and $K_3$ are vertex covering. However, subset $K_4$contains vertex 'a' and 'c' which do not cover the edge 'bd', therefore $K_4$ is not vertex covering.

$K_1$ = {b,c}
$K_2$ = {a, b,c}
$K_3$ = {a,d, c}
$K_4$= {a, c}

Space for learners:

**Graph Partitioning:**A graph partition is the process of reducing a large graph to a smaller one by grouping its nodes into mutually exclusive groups.

Graph chromatic partitioning: A proper colouring of a graph induces partitioning of vertices into different subsets such as the graph shown in Figure 5.7 (b) can be portioned into {v1, v6}, {v2, v5}, {v3} and {v4}. As it can be observed that no two vertices in the four subsets are adjacent. Such a subset of vertices is called an independent set.

A maximal independent set is an independent set to which no other vertex can be added without compromising its independence property. There can be many maximal independent sets of different sizes, however the one with largest number of vertices is of particular importance.

**CHECK YOUR PROGRESS**

i. In a graph G, number of vertices is 5. What is the total number of elements in the adjacency matrix?
   a) 5
   b) 25
   c) 10
   d) 125

ii. Which of these adjacency matrices represents a simple graph?
   a) [ [0, 0, 1], [1, 0, 1], [1, 0, 0] ]
   b) [ [1, 0, 0], [0, 1, 0], [0, 1, 1] ]
   c) [ [1, 1, 1], [1, 1, 1], [1, 1, 1] ]
   d) [ [0, 0, 1], [0, 0, 0], [0, 0, 1] ]

iii. In a simple graph, sum of the column in an incidence matrix is ___________
   a) number of edges
   b) greater than 2
   c) number of edges + 1
   d) equal to 2

***Space for learners:***

iv. The dimensions of an incidence matrix for graph having v as the number of vertices and e as the number of edges is given by ___________.

a) e x e
b) v x e
c) v x v
d) ex (v + e)

v. Incidence matrix and Adjacency matrix of a graph G will always have ________?

a) Same dimension
b) Different dimension
c) Some cases may have different dimension
d) None of the above

vi. Vertex coloring of a graph is_______________.

a) Adjacent vertices do not have same color
b) Adjacentvertices always have same color
c) All vertices should have a different color
d) All vertices should have same color

vii. Minimum number of unique colors required so that adjacent vertices do not have the same colour is given by_____________.

a) chromatic key
b) chromatic index
c) chromatic number
d) color number

viii. In an empty graph having n vertices_________ number of unique colours will be needed for vertex colouring.

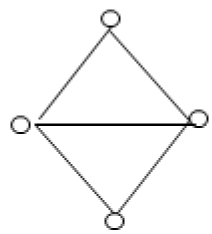
a) n + 1
b) 1
c) 2
d) n

ix. In an empty graph having n vertices_________ number of unique colours will be needed for vertex colouring.

***Space for learners:***

a) n-1
b) 1
c) n+1
d) n

x. How many unique colors will be required for vertex coloring of the following graph?



a) 2
b) 3
c) 4
d) 5

Space for learners:

## 5.8 SUMMING UP

- Adjacency matrix is represented by $M[a_{ij}]$ where $a_{ij}$ is the $i^{th}$ row, $j^{th}$ column element. The general form is given by:

$$a_{ij} = \begin{cases} 1; \text{if an edge in the graph between the vertex } v_i \text{ and } v_j \\ 0; \text{otherwise} \end{cases}$$

- Incidence matrix $I[a_{ij}]$ is a matrix of order ($n$ x $e$) where the element $a_{ij}$, where rows correspond to its vertices and columns correspond to its edges is defined as

$$\begin{cases} 1; \text{ if vertex } i \text{ belongs to edge } j \end{cases}$$

$a_{ij} =$

0; otherwise

- An adjacency list is a group of unordered lists that is used to describe a finite graph. Each unordered list in an adjacency list describes a vertex's collection of neighbours in the graph.
- Graph Coloring problem is to paint all the vertices such that no two adjacent vertices are of same colour.
- Graph Covering: A covering graph C is a subgraph which contains either all the vertices or all the edges corresponding to some other graph G.
- A graph partition is the process of reducing a large graph to a smaller one by grouping its nodes into mutually exclusive groups.

## 5.9 ANSWERS TO CHECK YOUR PROGRESS

| | | | | |
|---|---|---|---|---|
| i, b | ii, a | iii, d | iv, b | v, b |
| vi, a | vii, c | viii, b | ix, d | x, b |

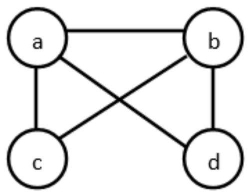
## 5.10 POSSIBLE QUESTIONS

Q1 Draw the graph having the following matrix as its adjacency matrix.

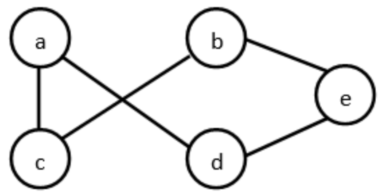
$$\begin{bmatrix} 0 & 1 & 2 & 3 & 4 \\ 1 & 1 & 2 & 2 & 3 \\ 4 & 1 & 3 & 1 & 2 \\ 4 & 3 & 2 & 1 & 0 \end{bmatrix}$$

Q2 Draw the adjacency matrix and adjacency list of the following graphs

Space for learners:

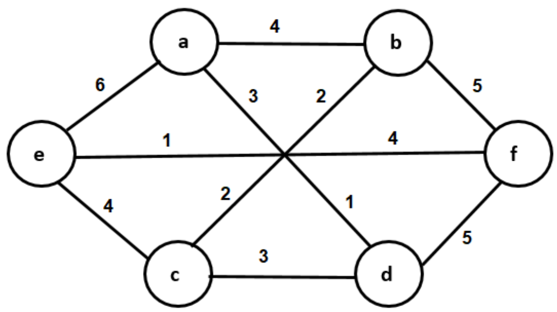



(a)




(b)

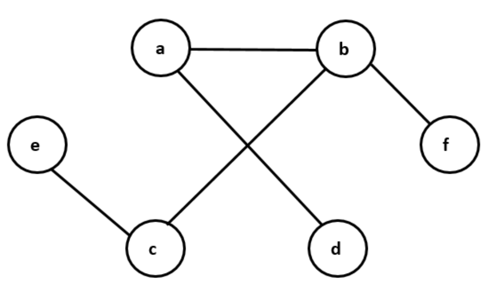
Q3 Write the adjacency matrix of the graph given below.




Q4 Draw the graph for the incidence matrix given below:

$$
\begin{array}{cc}
 & \begin{array}{ccccccc} \mathbf{e1} & \mathbf{e2} & \mathbf{e3} & \mathbf{e4} & \mathbf{e5} & \mathbf{e6} & \mathbf{e7} \end{array} \\
\mathbf{I} = \begin{array}{c} \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \\ \mathbf{d} \\ \mathbf{e} \end{array} & \left[\begin{array}{ccccccc}
1 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 & 1 & 1 \\
1 & 1 & 1 & 0 & 1 & 1 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 & 0
\end{array}\right]
\end{array}
$$

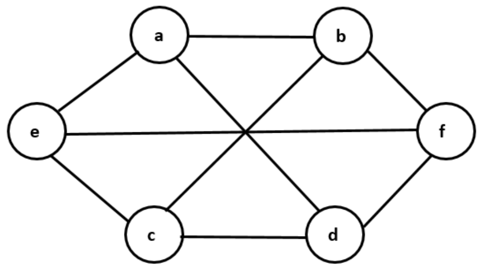
Q5 Draw the incidence matrix for the graph given below.

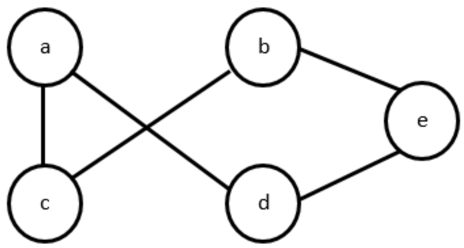



***Space for learners:***

Space for learners:

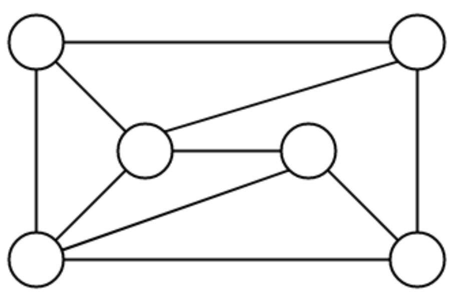
Q6 What is graph colouring? Explain using an example.
Q7 What is graph covering?
Q8 Differentiate between vertex covering and edge covering.
Q9 Find chromatic number of the following graphs




(a) (b)

Q10 Find chromatic number of the following graph.



## 5.11 FURTHER READING

- Graph Theory with Applications to Engineering and Computer Science by Narsingh Deo, Published by Prentice Hall India Learning Private Limited.
- Introduction to Graph Theory by Richard J Trudeau, Published by Courier Corporation.

- A First Course in Graph Theory by Gary Chartrand and Ping Zhang, Published by Courier Corporation.
- Graph theory with applications by John Adrian Bondy, Published by Elsevier Publishing Company


***Space for learners:***

# BLOCK III:

# AUTOMATA THEORY

# UNIT 1: INTRODUCTION TO LANGUAGES AND GRAMMAR

Space for learners:

**Unit Structure:**

## 1.1 INTRODUCTION

Space for learners:

To write instructions for machines it is important to learn syntax of the language and to designed computing machines, automata theory is important. For formalizing the notion of a language, we must include all the varieties of languages such as natural language produced by human being and languages for computer. Automata theory is a theoretical branch of Computer Science and Mathematics. It primarily deals with the logic of computation by some simple machine.

The word "automata" is a plural form of the word "automaton". The meaning of the word "automaton" is mechanization i.e., the condition of being automatically operated or controlled. Automating a process means performing it in a machine without the intervention of human. To perform a particular task in a mechanical environment, inputs, energy and control signal are required so thatit can produce the output without the direct involvement of human. Worked performed by machines are more accurate and efficient and it takes less time.

In the context of computer science, an automaton is a machine that can perform the computation in a mechanized manner. An automaton with a finite number of states is called a *finite automaton*. It is very important that the computing machine understands the instructions given by human. And it is necessary to develop languages for writing these instructions so that the machine can understand unambiguously.

This unit is an attempt to give the concept of languages and grammar in the context of computer science.

## 1.2 UNIT OBJECTIVES

After going through this unit, you will be able to:

- define alphabet, string, substring empty string, concatenation, Kleene closure etc.
- learn about variables, terminals, productions rules etc.
- define grammar and language in the context of theory of computer science
- determine language generated by a grammar

- learn about Chomsky classification of grammar

Space for learners:

## 1.3 INTRODUCTION TO FORMAL LANGUAGES

We begin our discussion with the concept of *set*. A set is a collection of elements or objects. For example, a set of three elements a, b and c can be written as S= {a, b, c}. It can be written in any order like S = {b, a, c}.

But a sequence is an ordered collection of elements. A sequence {a, b} is not same as {b, a}.An ordered collection is one in which the arrangement of objects matters.

Characters and their orders are important components in the formation of any language. For instance, the word "cat" does not carry the same meaning "act" though the letters or the elements are same. The most vital component of any language is its character set. In case of language such as English, any word can be formed from the English alphabet set S = {A, B, C, D,……,Z, a, b, c, d,…….,z}. Whereas a sentence is a combination of sequence of symbols from the Roman alphabet along with punctuation marks such as *comma*, *full-stop*, *colon* and *blank-space* which is used to separate two words.

Suppose we want to form words consisting of 5 letters from the set of English alphabets, say S. Then $S^5$ will be the all possible sequence of word of length 5. Thus, $S^n$ represents the set of all possible n letter sequences. A word becomes valid when it carries some meaning. For example, the word "*cake"* has meaning but the reverse "*ekac"* has no meaning. Whereas, the word "*cat"* and its reverse "*act*" carries definite meaning in English language. The one who understands a language L can able to differentiate the meaningful and meaningless word of that particular language.

The same is true in the context of computer language. For example, in case of C programming language, we write codes for program using the character sets of the C language. Each and every component of statements of the programming language is formed by combining the characters from the character sets so that the compiler can understand and compile the statements of the program.

We can define a language is a set of valid words over its character set. If we denote the set of alphabets or character set by the symbol $\sum$, then $\sum^*$ represents the set of all possible words or strings that can be constructed using the characters in ∑.It is therefore observed that formal learning of a language includes the following:

- Learning the alphabet
- Words which are formed by various sequence of symbols of its alphabet.
- Sentence formation; Sentences are formed by combining sequence of various words following some rules.

Formal language is designed for use in which natural language is unsuitable. If we want to instruct an abstract machine, we have to use something much more precise. An abstract machine takes instruction given by humans and provides the desired output. So, it is necessary to develop languages for these instructions. The languages that are developed with precise *syntax* and *semantics* are called ***formal languag***e. Syntaxes are precise rules that tell us the symbols we are allowed to use and how to put them together into legal expressions. And semantics tell us the meanings of the symbols and legal expressions. Formal languages play an important role in the development of compilers.

Let us now discuss the basic terminologies which are important and frequently used in formal languages and automata theory.

### 1.3.1 Basic Terminologies

**Symbols**:

Symbol can be any alphabet, letter or any element which is considered as the smallest building block of a language. Symbol can be considered as the atom of a language. For example, a, b, c, x, y, 0, 1, #, $\varepsilon$, etc. are symbols.

**Alphabet**:

Dictionary meaning of alphabet is a finite set of characters that include letters and is used to write a language. Mathematically we can define alphabet as a finite, non-empty set of symbols. Alphabet of a language is generally denoted by the summation " ∑ " symbol.

For example,

***Space for learners:***

Binary alphabet consists of 0 and 1 only. It is represented as $\sum = \{0, 1\}$

Roman alphabet is denoted by $\sum = \{a, b, \ldots.., z\}$.

$\sum = \{a, b, c, \varepsilon\}$ is an alphabet.

$\sum = \{\alpha, \beta, \gamma\}$ is an alphabet.

But, $A = \{1,2,3, \ldots \ldots\}$ is not an alphabet because it is infinite.

**String**:

A "string" over an alphabet $\sum$ is a finite sequence of symbols from that alphabet$\sum$ which is written next to one another and not separated by commas. A string is also known as a *word*. For example,

i) If $\sum = \{a,b\}$ be an alphabet; then ab, ba, aa, bb, aba, bab, bbaab, aababa,…are some examples of strings over $\sum$.

ii) If $\sum = \{0,1\}$ then 110010, 001, 10, 01, 0001, 111101, etc. are some stringsover $\sum$.

iii) If $\sum = \{a, b, c, d, \ldots., z\}$ the any combination of symbols *abp, bac, pqre, bcatdpr,* etc.are some strings over $\sum$.

iv) The set of all strings over an alphabet $\sum$ is denoted by $\sum^*$. For example, if $\sum = \{0,1\}$ then$\sum^* = \{\varepsilon, 0, 1, 00, 11, 01, 10, 111, 000, 010, 101, \ldots \ldots.\}$

**Length of a string**:

Length of a stringwis defined as the number of symbols or element present in the string. For example, length of 110010 is 6. Length of the string *bcatdpr* is 7. Length of a string w is represented within two vertical bars "|w|" as follows:

$|00111010| = 8$

$|101010101110| = 12$

$|abba|=4$

$|a|=1$

$\varepsilon$ is the empty string and has length zero.

**Empty String**:

The string that has no element i.e., of zero length is called the "empty string". Empty string is denoted by the symbol $\varepsilon$ (epsilon).Length of empty string is

$|\varepsilon| =0$

***Space for learners:***

**Reversing a string**:

Writing a string in reverse order is known as reversing a string.

If w= $w_1w_2w_3\ldots..w_n$ where $w_i\in\sum$, the reverse of the string w is $w_n w_{n-1}w_{n-2}\ldots w_1$

**String concatenation**:

When we write one string appending the other string at end,then it is known as *string concatenation*. It is one of the most fundamental operations used for string manipulation.

Let $x =$ $a_1a_2a_3\ldots..a_n$ and $y$ = $b_1b_2b_3\ldots\ldots b_m$ be two strings of length n and m, then the concatenationof the two strings $x$ and $y$ is written as$xy$, which is the string obtained by appending y to the end of x. The concatenated string $xy$is $xy$= $a_1a_2a_3\ldots..a_n\ b_1b_2b_3\ldots\ldots b_m$

The empty string $\varepsilon$ satisfies the property $\varepsilon w = xw = w$ where w is a string.

**Substring:**

We say that $x$ is a *substring* of wif $x$ occurs in w, that is w= *uxv*for some strings *u* and *v*. For example, "***put***" is a substring of the string *com**put**er*.

**Suffix and Prefix**:

If w = xv for some x, then v is a suffix of w. Similarly, if w = ux for some x, then u is the prefix of w.

Again, for w=uxv, the substring x will be the prefix of w if u= $\varepsilon$ and x will be the suffix of w if v= $\varepsilon$.

**Languages**

We have already been acquainted with the concept of strings. Any set of strings over an alphabet $\sum$ is called a *language*.Language can be finite or infinite. We usually denote a language by the letter L. As $\sum^*$ represents the set of all strings, including the empty string $\varepsilon$ over the alphabet $\sum$, we can define a language L over an alphabet $\sum$ as a subset of $\sum^*$. Thus

$L = \{w \in \sum^*$: w has some property P$\}$

Some other examples of language are as follows:

i) $L = \{w \in \{a, b\}^*$: $w$ has an equal number of a's and b's$\}$

ii) $L = \{w \in \sum^* : w = w^R\}$ where $w^R$ reverse string of w.

Space for learners:

iii) The set of all strings over {0,1} that start with 0.

iv) L= { $\varepsilon$ , 0, 00, 000, ……, } is a language over alphabet { 0}.

v) L= $\{0^n1^n2^n : n \geq 1\}$ is a language.

vi) The set of all strings over {a, b, c} having ab as a substring.

vii) The set of empty string {$\varepsilon$ } is also a language over any alphabet.

viii) The empty set $\emptyset$ is a language over any alphabet. {$\varepsilon$ }

Space for learners:

It should be noted that $\emptyset \neq \{\varepsilon\}$. Because the language does not contain any string but {$\varepsilon$ } contains a string $\varepsilon$.Also, length of $|\{\varepsilon\}|$ =1 but $|\emptyset| = 0$

**Concatenation of languages:**

If $L_1$ and $L_2$ are languages over some alphabet $\sum$, their concatenation can be denoted by $L=L_1.L_2$ or, $L = L_1L_2$ where

L={w∈ $\sum^*$ : w = x . y for some x $L = \{w \in \sum^* : w = x.y$ for some $x \in L_1$,
$y \in L_2\}$.

For example,

i) If $L_1=\{0,1\}$ and $L_2= \{1,00\}$, then $L_1L_2$ = {01,000,11,100}

ii) If $L_1=\{0,1,2\}$ and $L_2= \{1,00\}$, then $L_1L_2$ = {01,000,11,100, 21,200}

iii) If $L_1$={a, ab, abab} and $L_2$= {pq, ppqq, pppqqq}, then $L_1L_2$={apq, appqq, apppqqq, abpq, abppqq, abpppqqq, ababpq,
ababppqq, ababpppqqq}

iv) If $L_1$={b,ba,bab } and $L_2$= { $\varepsilon$ ,b,bb}, then $L_1L_2$ = {b, bb, bbb, ba, bab, babb, babbb}

Since string concatenation supports the *associative* property, so the concatenation of languages is also *associative*. Thus, if $L_1$, $L_2$ and $L_3$ are three languages then

$$(L_1L_2)L_3 = L_1(L_2L_3)$$

It is to be noted that $L_1L_2 \neq L_2L_1$

**Kleene Closure:**

In terms of formal languages, another important operation is *Kleene closure* or*Kleenestar*. *Kleene closure* of a language L is denoted by $L^*$.

Space for learners:

$L^*$ can be define as
$L^* = \{w \in \sum^*: w = w_1w_2w_3......w_n$, for some $n \geq 0$ and some $w_1,w_2,w_3,....,w_n \in L$. It can also be defined as follows:

$$L^* = \bigcup_{n\geq 0} L^n$$

$L^* = \{$Set of all strings over $\sum\}$

Examples:

i) If $\sum = \{a,b\}$ and a language over L over $\sum$, then

$$L^* = L^0 \cup L^1 \cup L^2 \cup L^3 \dots\dots\dots$$

$L^0 = \{\varepsilon\}$
$L^1 = \{a, b\}$
$L^2 = \{aa, ab, ba, bb\}$ and so on.
So, $L^* = \{\varepsilon, a, b, aa, ab, ba, bb, \dots\dots\}$

ii) If $\sum = \{0\}$ and a language over L over $\sum$,
then $L^* = \{\varepsilon, 0,00,000,0000, \dots\dots\dots\dots\}$

**Positive Closure:**

If $\sum$ is an alphabet then positive closure of the language L denoted by $L^+$ is the set of all strings over $\sum$ excluding the empty string $\varepsilon$.

$$L^+ = L^* - \{ \varepsilon \}$$

For example, if $\sum = \{0\}$,the $L^+ = \{0,00,000,0000, \dots\dots\dots\dots\}\sum$
The positive closure of a language L is

$$L^+ = \bigcup_{n\geq 1} L^n$$

Space for learners:

## CHECK YOUR PROGRESS

1. Given a string 011 over $\sum = \{0,1\}$. Find all the substring of the string.
2. For the binary alphabet { 0, 1 }, find $\sum^2$ and $\sum^3$.
3. If L = { a, ab }, find $L^*$ and $L^+$.

## 1.4 REGULAR LANGUAGES AND REGULAR EXPRESSIONS

In this section we are going to introduce the concept of regular languages and regular expressions. In mathematics, we can use operations like + and × to represent expression such as

$$(2+4)\times 5$$

The value of the above arithmetic expression is number 30.

Similarly, we can use regular operations to build up expressions describing languages, which are called regular expressions. The value of a regular expression is a language. As an example,

$$(0+1)^*11$$

In this case the value is the language L over {0,1} such that every string in L ends with two consecutive one.

We can define a ***regular expression***over an alphabet $\sum$ recursively as follows:

- Every character or alphabet belonging to $\sum$ is a regular expression.
- $\emptyset$, empty string $\varepsilon$, and $a$, for each $a \in \sum$, are regular expressions representing the languages $\emptyset, \{\epsilon\}$ and {a}, respectively.
- If r and s are regular expressions representing the language R and S respectively, then*concatenation* of these represented as *rs* is also a regular expression.
- If r and s are regular expressions representing the language R and S respectively, then the union of these represented as $r \cup s$ or $r+s$ is also a regular expression.
- The Kleene closure $r^*$ is a regular expression representing the language $R^*$.

A class of languages can be generated by applying operations like union, concatenation, Kleene star etc. on the elements. These languages are known as **regular languages** and the corresponding finite representations are known as **regular expressions.**

Some regular expressions and their corresponding regular sets are as follows:

| **Regular expression** | **Corresponding regular set** |
|---|---|
| 0 | {0} |
| 0+1 | {0,1} |
| a+b+c | {a, b, c} |
| $(11)^*$ | $\{\varepsilon, 11, 1111, 111111, \ldots.\}$ |
| ab+ba | {ab,ba} |
| $(a+b)^*c$ | {a, ac, acc, accc,…..b, bc, bcc, bccc, ………} |
| $(abc)^*$ | {$\varepsilon$, abc, abcabc,abcabcabc, …….} |
| $(abc)^*d$ | d, abcd,abcabcd,abcabcabcd,……..} |
| $ab^*cd$ | acd, abcd, abbcd, abbbcd, abbbbcd,…..} |
| ab(p+q) | {abp, abq} |

If $r$ is a regular expression, then the language representedby $r$ is denoted by $L(r)$. Further, a language $L$ is said to be *regular* if there exists a regular expression $r$ such that $L = L(r)$.

## 1.5 GRAMMAR

It is required to learn the grammar of a language while learning a specific language. For instance, it is required to learn English grammar while learning English language for forming meaning correct sentences. For the formation of sentences in any language, concept of grammar is very necessary. For getting the concept of grammar in the context of computer, let us first take some examples from English grammar. Here, we are considering two types of sentences in English; sentences having a ***noun and a verb*** or those with a ***noun, verb and adverb***.

**Space for learners:**

Space for learners:

| Noun- verb -adverb | Noun - verb |
|---|---|
| Barun ate quickly. | Barun ate. |
| Rita walked slowly. | Rita walked. |
| Neha talks slowly. | Neha sang. |
| Rishi writes slowly. | Rishi ran. |

We can see that **Noun- verb –adverb** and **Noun- verb** are description of two types of sentences in English grammar. If we replace noun, verb, adverb with some suitable word, we get grammatically correct sentences. In the example, we have seen in the example that sentences are formed by replacing noun with some name like Barun, Rita, Neha, Rishi, verb with ate,walked, talks, writes, etc. and adverb with quickly, slowly, etc.

If we call*noun-verb-adverb*or *Noun-verb*as variables(V), words like Barun, Neha, ate, writes, quickly, slowly, etc. as terminals(T), S be a variable representing a sentence, then following will be the rules (P) for generating two types of sentences:

$$S \rightarrow < noun >< verb >< adverb >$$
$$S \rightarrow < noun >< verb >$$
$$< noun > \rightarrow Barun$$
$$< noun > \rightarrow Rita$$
$$< noun > \rightarrow Neha$$
$$< verb > \rightarrow ate$$
$$< verb > \rightarrow write$$
$$< verb > \rightarrow walked$$
$$< adverb > \rightarrow slowly$$
$$< adverb > \rightarrow quickly$$

Thus we can describe a grammar by a 4-tuple: Variable (V), Terminals (T), S is a special symbol from V, P is a collection of rules which is termed a productions. The sentences are formed by starting with S, replacing words/strings using the production rules, and terminating when string of terminals is obtained.

A grammar consists of a set of rules (called *productions*) that specify the sequence of characters (or lexical items or sentences) that form allowable programs in the language been defined.Meaningful sentences (or statements) are formed using the grammar of the language. We have learnt that a grammar should have the following components

- A set of nonterminals symbols. These symbols are represented using capital letter like A, B, C, etc.
- A set of terminal symbols. Terminals are generally represented using small case letter like a, b, c etc.
- A start symbol from the set of nonterminals to represent a sentence from which various sentences of the language can be generated.
- A set of production rules.

### 1.5.1 Formal Definition of a Grammar

***Noam Chomsky*** gave a mathematical model of grammar in 1956 which turned out to be useful for writing computer language although it was not useful for describing natural languages. We will briefly discuss the different categories of grammar provided by Noam Chomsky in the next section.

A formal grammar is just a grammar specified using a strictly defined notation. For compiler technology, there are two useful grammars, which are regular grammar and context free grammar. Let us now write the formal definition of a grammar.

A grammar is a quadruple **G=(V, ∑,P,S)**
where $V$is a finite set of variables (non-terminals),
∑ is a finite set of terminals. Terminals are denoted by T also.
$S$ is the start symbol, where $S\epsilon V$
$P$is a finite non-empty set of rules whose elements are $\alpha \rightarrow \beta$,
where $\alpha, \beta$ are strings on $V \cup \sum$.
$\alpha$ has at least one symbol from V. The elements of P are called production rules.

Following points are to be noted while writing and substituting productions:

i) A production rule of a grammar is of the form $A \rightarrow \alpha$ where $A$ is a nonterminal symbol. The production rule $A \rightarrow \alpha$ is same as$(A, \alpha)\epsilon P$. But it is more convenient to write the production as$A \rightarrow \alpha$.

Space for learners:

ii) If $S \rightarrow AB$ is a production, then we can replace S by AB, but reverse substitution is not allowed. i.e., we cannot replace AB by S.

iii) $S \rightarrow AB$ is a production but $AB \rightarrow S$ is not.

Space for learners:

**Examples 1:** If G = ({S}, {a}, $\{S \rightarrow SS\}, S$), find the language generated by G.

**Solution:** Here we have V={S}, T={a}, Start symbol S, production rule

$$P: S \rightarrow SS$$

Since we have only one production rule $S \rightarrow SS$ in G and it has no terminal on the right-hand side, so we will not get any string from the production. Therefore, the language generated by G is $L(G) = \emptyset$.

**Example 2:** Consider the Grammar G = ( V,T,P, S) where T= {a, b},

P=$\{A \rightarrow Aa, A \rightarrow Ab, A \rightarrow a, A \rightarrow b, A \rightarrow \varepsilon\}$, S={A}. White a common generated by this grammar exacting few strings of the grammar.

**Solution:** Here the start symbol is A.

A$\rightarrow Aa \rightarrow aa$

A$\rightarrow Ab \rightarrow ba$

A$\rightarrow Ab \rightarrow Aab \rightarrow Aaab \rightarrow aaab$

A$\rightarrow Ab \rightarrow Abb \rightarrow Aabb \rightarrow babb$

A$\rightarrow Ab \rightarrow \varepsilon b \rightarrow b$

A$\rightarrow Aa \rightarrow \varepsilon a \rightarrow a$

A$\rightarrow Aa \rightarrow Aaa \rightarrow Abaa \rightarrow Abbaa \rightarrow \varepsilon bbaa$

Hence this grammar can be used to produce the strings of the form $(a+b)^*$

## 1.6 CHOMSKY CLASSIFICATION OF GRAMMARS

So far, we have seen that a grammar depends on its production rules to derive strings in the associated language. *Noam Chomsky* classified the grammar into four categories which based on their production rules.

***Type 3***: The first category is known as the *type 3*which is also referred to as *regular grammar*. We have already been acquainted with regular grammar and regular language in our previous section. The production rules for type 3 grammar are of the following forms:

$$A \rightarrow a$$

$$A \rightarrow aB$$

where A and $B$ are some nonterminals and $a$ is some terminal in the grammar. Type 3 grammars are recognized by *finite automaton*. In case of type 3 grammar, productions in the left-hand-side consists of a non-terminal only and the productions in the right-hand-side contains either a single terminal or a terminal followed by a single nonterminal.

***Type 2***: The second category is the *type 2* category which is also known as *context-freegrammar* (CFG). The productions of type 2 grammar are of the form

$$A \rightarrow (\sum \cup V)^*$$

The left-hand-side of every production in *type 2* grammar consists of one non terminal only, while the right-hand-side consists of a combination (union) of terminals from ∑ and nonterminals from V. The name of the automaton which accepts the type 2 grammar is *pushdown automaton*.

***Type 1:***The third category of Chomsky classification of grammar is type 1 grammar which is also known as *context-sensitive grammar*. It has the following form of production:

$$(\sum \cup V)^* \rightarrow (\sum \cup V)^*$$

Here, combination of variable and terminals are in both side. But the size of the string produced on the right-hand-side should either be greater than or equal to the size of the string on the left-hand-side of the production. *Linear bounded automaton* recognizes the language generated by type 1 grammar.

***Type 0:***The fourth category is termed as type 0 grammar. This grammar is also known as *unrestricted grammar*. The language generated by type 0 grammars are accepted by Turing machine. The form of production of type 0 grammar is

$$(\sum \cup V)^* \rightarrow (\sum \cup V)^*$$

The production rules are same as type 1 but it has no restrictions.

**Space for learners:**

## CHECK YOUR PROGRESS

**4. Choose the correct option:**

i. Language of finite automata is generated by
   a) Type 0 grammar
   b) Type 1 grammar
   c) Type 2 grammar
   d) Type 3 grammar

ii. Regular expression of all strings start with ab and ends with ba is
   a) $(a+b)^*ab(a+b)^*$
   b) $ab(a+b)^*ba$
   c) $(a+b)^*ab(b+a)^*$
   d) $aba^*b^*ba$

iii. L= $\{\varepsilon, b, bb, bbb, bbbb, \ldots ....\}$ is represented by
   a) $a^+$
   b) $a^*$
   c) both a) and b)
   d) none of these

iv. Given: $\sum = \{a, b\}$, L= $\{x \in \sum^* | x$ is a string combination$\}$.
$\sum^4$ represents which among the following?
   a) {aa, ab, ba, bb}
   b) {aaaa, abab, ε, abaa, aabb}
   c) {aaa, aab, aba, bbb}
   d) All of these

v. Regular expression for all strings starts with *ab* and ends with *bba*
is
   a) $ab(a+b)^*bba$
   b) $aba^*b^*bba$
   c) $ab(ab)^*bba$
   d) All of these

Space for learners:

## 1.7 SUMMING UP

- A formal language is a set of strings of symbols drawn from a finite alphabet. It can be specified either by a set of rules that generates the language, or by a machine that accepts or recognizes the language.

- An alphabet $\sum$ is a finite and non empty set of symbols.
- A string is a finite sequence of symbols from some alphabet.
- A language L over some alphabet $\sum$, is a collection of strings over the alphabet. For example,

L= {ε,1,111, …} is a language over the alphabet {1}

L=$\{0^n1^n2^n : n\geq 1\}$ is a language.

- The Kleene closure of a language L is denoted by $L^*$.

$L^*$ = {Set of all words over $\sum$}

={word of length zero, words of length one, words of length two, ..}

$= L^0 \cup L^1 \cup L^2 \cup \ldots \ldots$

- If is an alphabet then positive closure of $\sum$ is denoted by $\sum^+$ and is defined as $\sum^+=\sum^*-\{\varepsilon\}$
- A grammar consists of four items: a set of terminals $\sum$, a set of nontermninals V, a set of productions P, and a special symbol S, known as start symbol which is a nonterminal and is belongs to V.
- Noam Chomsky classified the grammar into four categories which are based on their type of production.
- Type 3 is known as regular grammar, type 2 is known as or context-free grammar, type 1 in known as context-sensitive grammar and type 0 grammar is known as unrestricted grammar.

## 1.8 ANSWERS TO CHECK YOUR PROGRESS

**Ans. 1:**

Substrings of the string 011 are : ε, 0, 1, 01, 11, 011

**Ans 2:**

$\sum^2$= {00, 01,10,11

$\sum^3$= {000,001,010,011,100,101,110,111}

**Ans 3:**

Given L = { a, ab }. Then we can determine $L^*$ as

$L^*= L^0 \cup L^1 \cup L^2 \cup \ldots\ldots\ldots$

Space for learners:

$= \{\varepsilon\} \cup \{a, ab\} \cup \{aa, abab, aab, aba\} \cup \ldots\ldots..$

$L^{+} = L^{1} \cup L^{2} \cup \ldots\ldots\ldots$

$= \{a, ab\} \cup \{aa, abab, aab, aba\} \cup \ldots\ldots..$

**Ans 4:**

(i)(d) Type 3 grammar

(ii)(b) $ab(a+b)^{*}ba$

iii)(b)$a^{*}$

iv)(b) {aaaa, abab, ε, abaa, aabb}

v) (a) $ab(a+b)^{*}bba$

## 1.9 POSSIBLE QUESTIONS

1. Define Kleene star. Give examples.
2. What is a language?
3. Define $\sum^{+}$.
4. Define empty string.
5. Define prefix and suffix of a string with examples.
6. Define length of a string.
7. Define alphabet with suitable examples.
8. Define a regular language.
9. Give the formal definition of grammar. Write the categories of grammars provided by Noam Chomsky with their production types.
10. Define a grammar of a language.
11. Define regular expressions? Give some examples of regular expression.
12. Write the Regular expression for the following languages/sets
    i) L= {aa, aaaa, aaaaaa, aaaaaaaa,….}
    ii) Language L over {0,1} such that every string in L ends with 11
    iii) Language L over { a, b, c} such that every string in L ends with 11
    iv) L= {00,001,0011,00111,….}

Space for learners:

v) Set of all string over {0,1}containing exactly one 0

vi) Set of all strings over { a, b} containing exactly two a' s.

vii) Set of all strings over {a, b, c} beginning with c and ending with cc.

Space for learners:

## 1.10 REFERENCES AND SUGGESTED READINGS


1. Mishra, K. L. P., & Chandrasekaran, N. (2006). *Theory of Computer Science: Automata, Languages and Computation*. PHI Learning Pvt. Ltd.
2. Nagpal, C. K. (2012). *Formal Languages and Automata Theory*. Oxford University Press.
3. Hopcroft, J. E., Motwani, R., & Ullman, J. D. (2007). *Introduction to automata theory, languages, and computation*. Pearson Education

# UNIT 2: INTRODUCTION TO FINITE AUTOMATA

Space for learners:

**Unit Structure:**

## 2.1 INTRODUCTION

Space for learners:

This Unit discusses fundamental concepts of Theory of Computations. This unit covers the concepts around Deterministic and Non deterministic automata thoroughly with easily understandable examples. Differences, theorems and conversions are also easily discussed with mathematical techniques. It discusses how to construct finite automata for any language, whether it is a DFA, NFA or NFA with empty moves. Basically, for different input symbols, when the machine state is not determined, i.e., machine can move to any states of the automaton, it is called as Nondeterministic finite automata (NFA) and if the machine state is determined then it is known as Deterministic finite automata (DFA). So, let's study their detailed definitions with different type of properties.

## 2.2 UNIT OBJECTIVES

This unit revolves around DFA, NFA and ε-NFA. After going through this unit, you will be able to –

- Get full concept on DFA and NFA with examples.
- Construct DFA, NFA for any given example
- Convert an NFA into DFA
- Convert a ε-NFA into its equivalent DFA.
- Equivalence of NFA and DFA.
- Discuss the differences between DFA, NFA and ε-NFA.
- Minimize states in a DFA.
- Check a particular string is belongs to finite automata or not.

## 2.3 DETERMINISTIC FINITE AUTOMATA

It is a finite automaton (FA) where our machine exists only one place at a time. For each input symbol, the machine state is determined, i.e., machine will move to only one certain state, hence it is called as deterministic finite automata. A deterministic finite automata consists of five tuples {**Q, Σ, $q_0$, F, δ**} where-

**Q** represents finite set of states.

**Σ** represents set of all input symbols, i.e., Alphabet.

**$q_0$** represents Initial state.

**F** represents finite set of final state/states.

**δ** represents Transition Function, which takes two arguments, a state and an input symbol, it returns a single state. So, **δ : Q X Σ -> Q.**

Some real-life examples of Deterministic Finite Automata (DFA) are lifts in buildings, text parsing, video game character behavior, security analysis etc. A Deterministic Finite Automata (DFA) generally represented by digraph, which is known as transition diagram or state diagram, where vertices represent states and arcs shows the transition from one state to another state.

Let us now discuss one example of Deterministic Finite Automata (DFA). Suppose our DFA's tuples look like below-

Q = {a, b}

Σ = {0, 1}

$q_0$ = {a}

F = {b}

For transition function **δ**, we need to show the mappings from one state to another state on each input symbols, that's why we need a table:

| **States** | **Input Symbols 01** | |
|---|---|---|
| a | b | a |
| b | b | a |

So, from the table we can easily understand that on accepting input symbol '0' state 'a' is moving to state 'b' and on accepting input symbol '1' state 'a' remains unchanged, i.e., self-looped and In this manner, we can easily identify the transitions of state 'b' as well. So, let's draw a diagram for the above example:

Space for learners:

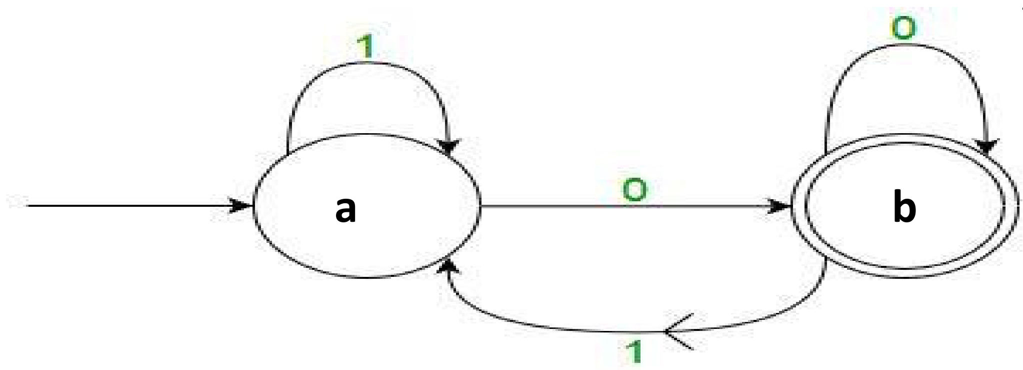



*Figure 1: A DFA example*

Now, additionally, we can discuss this above diagram more. Check the diagram, see in the final state 'b', input symbol '0' is self looped and from state 'a', by consuming only '0' we can reach the final state. So, we can easily say, this is a DFA, which will accepts all strings ending with zero. Since it is mathematical way to design a finite automata, so there can be numerous number of ways or patterns to design and draw it. But, a finite automaton with minimum number of states is always preferred.

Space for learners:

## 2.4 NONDETERMINISTIC FINITE AUTOMATA (NFA)

### 2.4.1 Nondeterministic Finite Automata (NFA)

A kind of finite automata (FA) where our machine can exist in multiple states at the same time. For input symbols, the machine state is not determined, i.e., machine can move to any states of the automaton, hence it is called as nondeterministic finite automata (NFA). That's why Non Deterministic Automata(NFA) is more complex than DFA. Just like Deterministic Finite Automata (DFA), NFA also consists of five-tuple {$\mathbf{Q, \Sigma, q_0, F, \delta}$} where-

**Q** represents set of all states.

**Σ** represents set of all input symbols, i.e. Alphabet.

$\mathbf{q_0}$ represents Initial state.

**F** represents set of final state or states.

**δ** represents Transition Function, which takes two arguments, a state and an input symbol, it returns any combination of Q states. So, **$\delta : Q \times \Sigma \rightarrow 2^Q$.**

If we compare this transition function with DFA's transition function, we know that Q is the subset of **$2^Q$** which indicates Q is contained in **$2^Q$** or Q is a part of **$2^Q$**, however, the reverse isn't true. So mathematically, we can say that all DFA is NFA but inverse is not true.

## 2.4.2 NFA Detailed Example

Some real-life examples of Nondeterministic Finite Automata (NFA) include playing cards, Tic tac toe and Ludo etc. As we have studied in the section of Deterministic Finite Automata (DFA), same notation technique i.e., digraph is used to draw Nondeterministic Finite Automata (NFA).

Let us now discuss earlier example for Non-Deterministic Automata (NFA). So, suppose our NFA's tuples look like below-

Q = {a, b}

Σ = {0, 1}

$q_0$ = {a}

F = {b}

For transition function **δ**, we need to show the mappings of states on each input symbols, that's why we need a table:

| **States** | **Input Symbols**<br>**01** |
|---|---|
| a | aa,b |
| b | -b |

From the above table, we can easily understand that on accepting input symbol '1' state 'a' can move either to state 'b' or to state 'a' and on

Space for learners:

accepting input symbol ‘0’ state ‘a’ remains unchanged, i.e., self-looped. So, let’s draw a diagram for the above example:

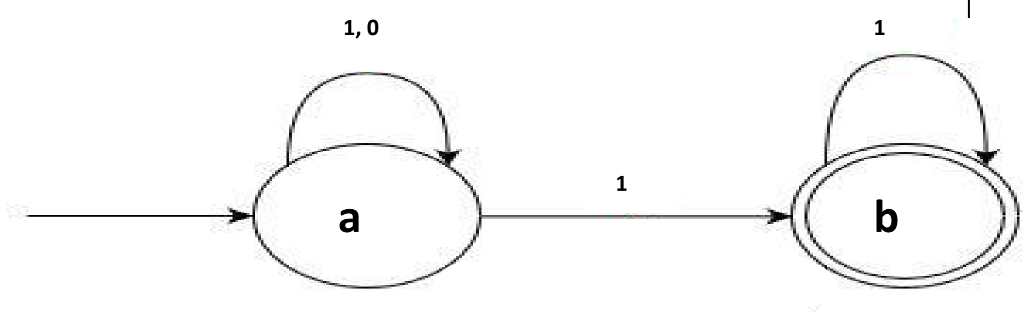



Figure 2: A NFA example

So, according to the diagram, see in the final state ‘b’, input symbol ‘1’ is self looped and from state ‘a’, by consuming only ‘1’ we can reach the final state. So, we can easily say, this is a NFA, which will accepts all strings ending with one. Furthermore, we can see there is no transition for input symbol ‘0’ from final state ‘b’, this type of mechanisms we can use while constructing a Non-deterministic finite Automata.

### 2.4.3 NFA v/s DFA

We have studied Deterministic Finite Automata (DFA) and Non-deterministic Finite Automata (NFA) in previous sections and learnt to know how these automaton works. We have seen that both the types of automaton are quite similar to each other, both NFA and DFA have same power and each NFA can be converted into a DFA, but they broadly differ from each other. Their differences are listed below:

***Space for learners:***

| Deterministic Finite Automata(DFA) | Nondeterministic Finite Automata(NFA) |
|---|---|
| For a particular input symbol, machine can be only in one state. | For a particular input symbol, machine can be in one state or multiple states. |
| In this type of Automata, next state is clearly specified. | In NFA, there can be numerous numbers of possible next states. |
| Every DFA is NFA. | Every NFA is not DFA. |
| A string is accepted by DFA, if it is terminates in a final state. Otherwise it rejects the string. | A string is accepted by NFA, if it's at least one of combinational transitions terminates in a final states. |
| It requires more space. | It requires less space than DFA. |
| Transition function is $\delta : Q \times \Sigma \rightarrow Q$, which means on any input symbol from our alphabet, machine state will move to only one state from Q. | Transition function is $\delta : Q \times \Sigma \rightarrow 2^Q$, which means on any input symbol from our alphabet, machine state will move to any possible combinations of states from set Q. |
| It is more difficult to construct. | It is easier to construct. |

Space for learners:

**CHECK YOUR PROGRESS**

Question 1: What are DFA and NFA's?

Question 2: What is Transition Table? Give one example.

Question 3: Write down the differences between DFA and NFA.

Question 4: Write the transition function for DFA and NFA.

## 2.4.4 Membership Example

Given the NFA M, is 01001 accepted by the NFA? The transition function for the given NFA is

| Inputs States | 0 | 1 |
|---|---|---|
| ->q0 | {q0, q3} | {q0, q1} |
| q1 | _ | {q2} |
| q2 | {q2} | {q2} |
| q3 | {q4} | _ |
| *q4 | {q4} | {q4} |

Solution: So, we will use transition function to solve it. As per transition table, q0 & q4 are initial and accepting states respectively.

$$\delta(q0, 0) = \{q0, q3\}$$

$$\begin{aligned}\delta(q0, 01) &= \delta(\delta(q0, 0), 1)\\ &= \delta(\{q0, q3\}, 1)\\ &= \delta(q0, 1) \cup \delta(q3, 1)\\ &= \{q0, q1\}\end{aligned}$$

$$\begin{aligned}\delta(q0, 010) &= \delta(\delta(q0, 01), 0)\\ &= \delta(\{q0, q1\}, 0)\\ &= \delta(q0, 0) \cup \delta(q1, 0)\\ &= \{q0, q3\}\end{aligned}$$

$$\delta(q0, 0100) = \{q0, q3, q4\}$$

$$\begin{aligned}\delta(q0, 01001) &= \delta(\delta(q0, 0100), 1)\\ &= \delta(\{q0, q3, q4\}, 1)\\ &= \delta(q0, 1) \cup \delta(q3, 1) \cup \delta(q4, 1)\\ &= \{q0, q1, q4\}\end{aligned}$$

We know, q4 is the final state and q4 is in the final set of states. So, we can say the string 01001 is accepted by the NFA. In similar way, we can check any string is either accepted or rejected by NFA and DFA.

## 2.4.5 NFA with Empty Moves

A kind of finite automata which contains ε (null or empty) move or instantaneous transition. As we studied, a nondeterministic finite automaton (NFA) can have zero, one, or multiple transitions corresponding to a particular symbol. It is defined to accept the input if there exists some choice of transitions that cause the machine to end up in an accept state. With NFA, we can easily solve complex problems. Epsilon NFA is nothing but an NFA with an additional feature named Epsilon (ε), is a convenient feature with which we can construct even more complex and bigger problems. Both NFA and ε-NFA can recognize same language. An example of ε-NFA is given below:

**Space for learners:**

ce for learners:

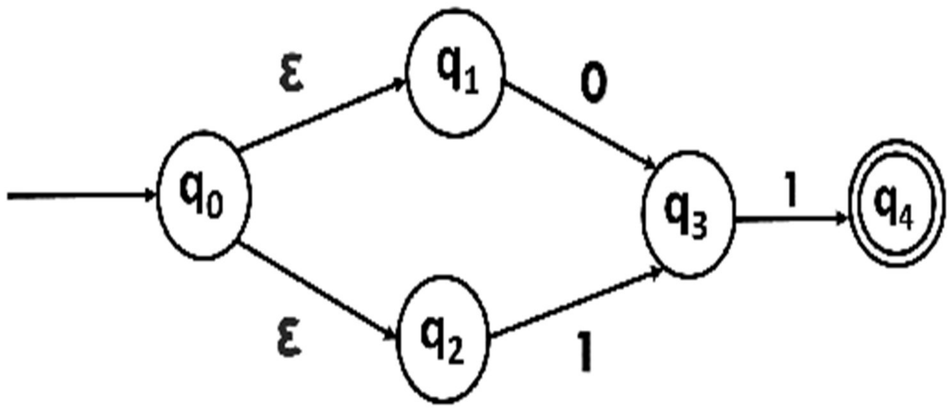



Figure 3: a ε-NFA Example

Check the above diagram, and see from the state $q_0$, the machine is moving to state $q_1$ and$q_2$ with ε transition that means without input symbol $q_0$ is changing its state. The transition table will look like:

| **States** | **Input Symbols** <br> **0 ε 1** |
|---|---|
| **$q_0$** | **- $q_1$ ,$q_2$ -** |
| **$q_1$** | **$q_3$- -** |
| **$q_2$** | **- -$q_3$** |
| **$q_3$** | **- -$q_4$** |
| **$q_4$** | **- - -** |

## 2.4.4.1 ε-closure

ε-closure is calculated for different states of an ε-NFA. ε-closure of a state 'q' means a set of states which can be reached from the state 'q' with ε move (empty/null move) including the self-state. That means set of states that can be reached without any input symbol is ε-closure of a state. Now, Let us find out ε-closure for each state of a ε-NFA given in figure 3:

$$\varepsilon\text{-closure } \{q_0\} = \{q_0, q_1, q_2\}$$
$$\varepsilon\text{-closure } \{q_1\} = \{q_1\}$$

ε-closure $\{q_2\} = \{q_2\}$
ε-closure $\{q_3\} = \{q_3\}$
ε-closure $\{q_4\} = \{q_4\}$

**CHECK YOUR PROGRESS**

Question 5: Discuss Epsilon NFA with example.

Question 6: What is ε-closure?

## 2.5 EQUIVALENCE OF DFA AND NFA

In this section, we will discuss the equivalence of DFA and NFA, which means their capability of recognizing language. As we studied deterministic finite automata and non-deterministic finite automata, it looked like they are different from each other. Their transition diagram, working etc. are different but when comes to recognize a language it turns out to be an equivalent of each other. We can convert an NFA to its equivalent DFA by any conversion algorithm. So, here we will prove the equivalence of NFA and DFA i.e.both NFA and DFA can recognize same type of languages which means for any DFA D, there is an NFA N such that L(N) = L(D) and For any NFA N, there is a DFA D such that L(D) = L(N).

### 2.5.1 Equivalence Theorem

Let's formally state the theorem below:
Let for any language, and suppose L is accepted by NFA N = $(\Sigma, Q, q_0, F, \delta)$. There exists a DFA D= $(\Sigma, Q', q'_0, F', \delta')$ which also accepts L. (L(N) = L(D)).

We just need to prove that DFA D is equivalent to NFA N. Through Induction method, we can prove it if we allow each state of DFA D to represent the state or set of states in the NFA N. So, firstly, let's configure the parameters of DFA D$(\Sigma, Q', q'_0, F', \delta')$,where-

$Q' = 2^Q$ and $q'_0 = \{q_0\}$

Space for learners:

$F' = \{q \in Q' | q \cap F \neq \emptyset\}$, where F' is the set of states in Q' and F is the set of final states in NFA.

$\delta'$ is the transition function of DFA D.

$\delta'(q,a) = U_{p\in q}\ \delta(p,a)$ for $q \in Q'$ and $a \in \Sigma$

We know from the transition function of both NFA and DFA, each state in the set of states Q' in D is nothing but a set of states itself from Q in N. For each state p in state q in Q' of D(p is a single state from Q), determine the transition $\delta(p,a)$. $\delta(p,a)$ is the union of all $\delta(p,a)$.

Now, we can easily prove that $\delta''(q_0',x) = \delta''(q_0,x)$ for every x. i.e., L(D) = L(N)

Basic Step:Let x be the empty string ε.

$$\begin{aligned}\delta''(q_0',x) &= \delta''(q_0', \varepsilon) \\ &= q_0' \\ &= \{q_0\} \\ &= \delta'(q_0, \varepsilon) \\ &= \delta'(q_0, x)\end{aligned}$$

Inductive Step:

Assume that for any y with |y|>=0, $\delta''(q_0',y) = \delta'(q_0',y)$

If we let n=|y|, then we need to prove that for a string z with |z| = n+1, $\delta''(q_0',z) = \delta'(q_0',z)$. We can then represent the string z as a concatenation of string y and symbol a from the alphabet Σ ($a \in \Sigma$).

So, z=ya

$$\begin{aligned}\delta''(q_0',z) &= \delta''(q_0',ya) \\ &= \delta'(\delta''(q_0',y),a) \\ &= \delta'(\delta'(q_0,y),a) \quad \text{(assumption)} \\ &= U_{p\in \delta'(q0,y)}\ \delta'(p,a) \\ &= \delta'(q_0, ay)\end{aligned}$$

Space for learners:

$= \delta' (q_0, z)$

Space for learners:

Now, DFA D accepts a string iff $\delta''(q_0', x) \in F'$. From the above explanation, it follows that D accepts x iff $\delta'(q_0, x) \cap F \neq \emptyset$. So, a string is accepted by DFA D, if and only if, it is accepted by NFA N.

There is another alternative and easy way to prove this theorem, approach is given below:

**Theorem:**

A language L is accepted by a DFA *if and only if* it is accepted by an NFA.

**Proof:**

If part:

Prove by showing every NFA can be converted to an equivalent DFA.

Only-if part:

Every DFA is a special case of an NFA where each state has exactly one transition for every input symbol. Therefore, if L is accepted by a DFA, it is accepted by a corresponding NFA.

By showing these two parts, we can easily solve the above Theorem.

## 2.5.2 NFA to DFA Construction

From the above equivalence theorem, we can conclude that there exists an equivalent DFA for any NFA. In this section we will learn to construct corresponding DFA for an NFA. So, let's discuss subset construction method to construct a DFA for NFA:

Let our NFA is $N = \{Q_N, \Sigma, \delta_N, q_0, F_N\}$. Our aim is to build a corresponding DFA $D=\{Q_D, \Sigma, \delta_D, \{q_0\}, F_D\}$ such that $L(D)=L(N)$

**Subset Construction:**

1. $Q_D$= all subsets of $Q_N$ (i.e., power set)

2. $F_D$=set of subsets S of $Q_N$ such that $S \cap F_N \neq \Phi$

3. $\delta_D$:for each subset S of $Q_N$ and for each input symbol a in $\Sigma$:

$\delta_D(S,a) = \cup\ \delta_N(p,a)$

For easy understanding of Subset construction method, we will take an example to construct a corresponding DFA from its NFA. Let's take a language $L = \{w \mid w \text{ ends in } 01\}$, for this NFA will be:

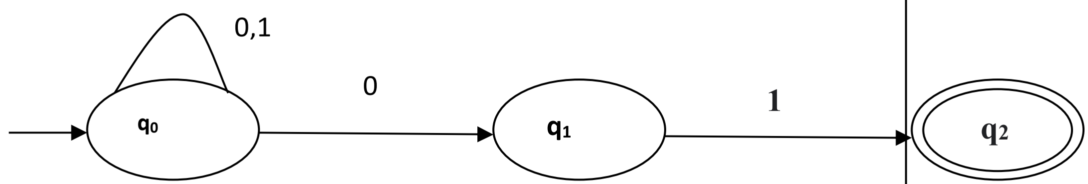



Figure 4: a NFA Example-to convert to DFA

We need to construct a DFA for this NFA by subset construction method. So, transition table of the NFA will look like:

| **States** | **Input Symbols** <br> **01** |
|---|---|
| $q_0$ | $\{q_0, q_1\}$ $\{q_0\}$ |
| $q_1$ | Ø $\{q_2\}$ |
| $q_2$ | Ø Ø |

Now, as per algorithm, we first need to find out all subsets of states. We have three states $\{q_0, q_1, q_2\}$, then there subsets are – Ø, $\{q_0\}$, $\{q_1\}$, $\{q_2\}$, $\{q_0, q_1\}$, $\{q_1, q_2\}$, $\{q_0, q_0\}$, $\{q_0, q_1, q_2\}$ i.e., $2^3=8$ subsets of states can be possible. After enumerating all the possible subsets, check the transition table of NFA, from which we have to determine important transitions. We have to give importance to the starting state, here starting state is $q_0$, now check,from $q_0$ only we can easily reach to other subsets of states like $\{q_0, q_1\}$ and $\{q_2\}$. So we will retain only those states which are reachable from $\{q_0\}$. So, let's construct the transition table of DFA. Since, our starting state is $\{q_0\}$, we will start from this, after writing its mappings, we need to find out the mappings of our new state $\{q_0, q_1\}$, for this we need to check the mappings of $q_0$, $q_1$ separately in the NFA table and then union it. In this way, we will find the new subset of states and eventually we will move to the final state.

***Space for learners:***

**Space for learners:**

Since, our final state in NFA is $q_2$, so, in DFA table, any combination of subset of states, which contains $q_2$ will become final state of the corresponding DFA.

| **States** | **Input Symbols**<br>**01** |
|---|---|
| $\{q_0\}$ | $\{ q_0, q_1\}$ $\{ q_0\}$ |
| $\{ q_0, q_1\}$ | $\{ q_0, q_1\}$ $\{ q_0, q_2\}$ |
| $\{ q_0, q_2\}$ | $\{ q_0, q_1\}\{ q_0\}$ |

So, from this DFA transition table, we can finally construct our resulting DFA, which is:

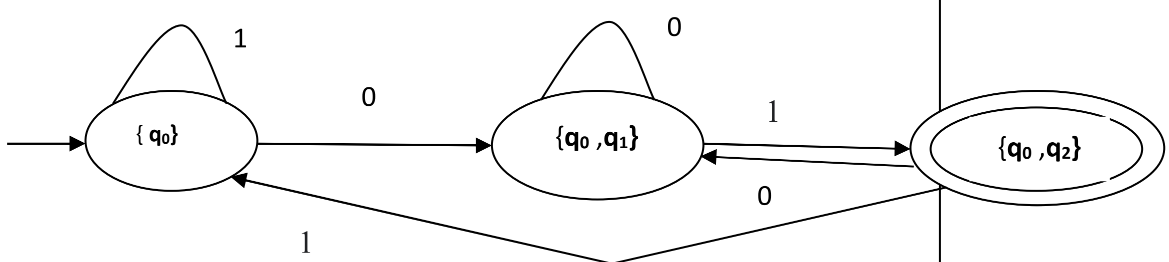



Figure 5: Resulting DFA of an NFA (Figure 4)

Now, we will discuss another method, which is easier than this. Method is known as Lazy Creation, Aim is to avoid enumerating all of power sets of states. In this method, we will create states of the resulting DFA by lazy creation of sets. Let us try to understand this method:

Let's take earlier example of figure 4 and check its transition table. Now, checking only that transition table, we can easily construct our DFA transition table by lazy creation of sets. Firstly, we need to check the starting state, from starting state $\{q_0\}$ we can write the states for its different input symbols. Now, see we get a new state named $\{ q_0, q_1\}$, so whenever we get a new state we need to define it first, i.e., we need to find its mappings. Now, we get another new state $\{ q_0, q_2\}$, so we need to define it. After defining it, check if there is any new state present or not, if there is a new state then define it. Define all the new states until no new state is there. When there is no new state present in my transition table, we are ready to draw the diagram of corresponding DFA, So for the example's DFA diagram, check figure 5.

| **States** | **Input Symbols**<br>**01** |
|---|---|
| $\{q_0\}$ | $\{q_0, q_1\}$ $\{q_0\}$ |
| $\{q_0, q_1\}$ | $\{q_0, q_1\}$ $\{q_0, q_2\}$ |
| $\{q_0, q_2\}$ | $\{q_0, q_1\}$ $\{q_0\}$ |

## 2.5.3 ε-NFA to DFA conversion

We need to recall ε-closure definition from previous sections. We have learnt how to find out ε-closure of each state of a ε-NFA. ε-NFA to DFA conversion is the easier conversion technique among all conversion techniques. Let's take an example of an ε-NFA as in figure 6, then very first we need to find out ε-closure of each states. Steps to convert ε-NFA to DFA are-

Step 1 : Take ε-closure for the beginning state of NFA as beginning state of DFA.

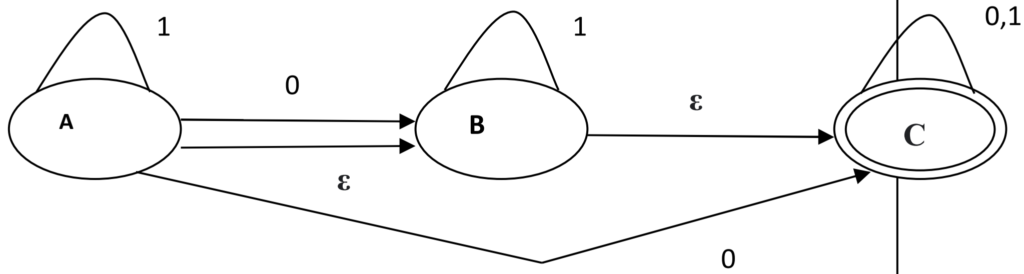
Step 2 : Find the states that can be traversed from the present for each input symbol .

Step 3 : If any new state is found take it as current state and repeat step 2.

Step 4 : Do repeat Step 2 and Step 3 until no new state present in DFA transition table.

Step 5 : Mark the states of DFA which contains final state of NFA as final states of DFA.

***Space for learners:***




*or learners:*

Figure 6: Epsilon NFA

**Transition Table will be:**

| **States** | **Input Symbols**<br>**0ε1** |
|---|---|
| A | B,C B A |
| B | - C<br>B |
| C | C C |

For the above example ε-closure are as follows :

ε-closure (A) : {A, B, C}
ε-closure (B) : {B, C}
ε-closure (C) : {C}

Now, using the algorithm steps, we will construct the transition table of DFA:

| **States** | **Input Symbols**<br>**01** |
|---|---|
| {A,B,C} | B,C<br>A,B,C |
| {B,C} | C<br>B,C |
| {C} | C C |

So, the resulting DFA is:

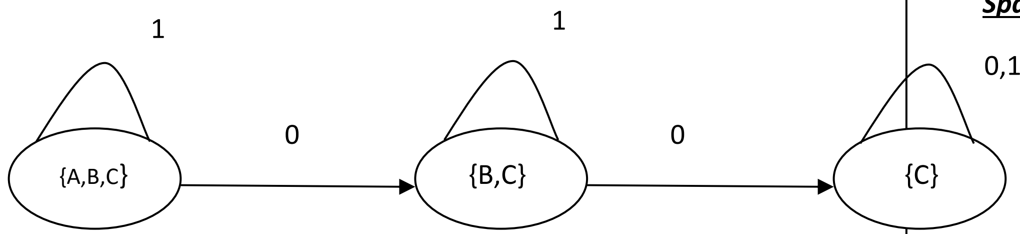



Figure 7: DFA of an Epsilon NFA (fig. 6)

**CHECK YOUR PROGRESS**

Question 7: What do you mean by lazy creation of sets?

Question 8: Discuss the equivalence theorem of DFA and NFA.

Question 9: What is the power of NFA and DFA in recognizing languages?

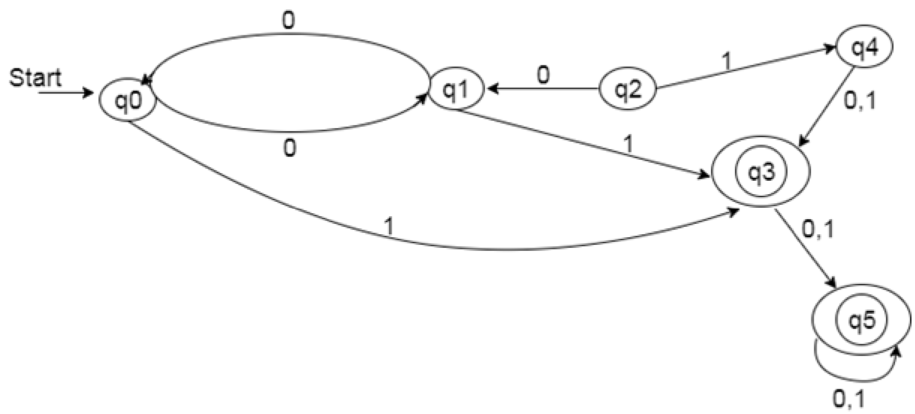
## 2.6 MINIMIZATION OF FA

Minimization of Finite Automata means reducing the useless and redundant states from given Finite automata. Here, we are saying FA, we are mainly indicating DFA. Reducing number of states leads our automaton faster, consumes very less space and eventually become easier to implement. Steps to minimizing DFA are given below:

Step 1: Remove all the states that are unreachable from the initial state via any set of the transition of DFA.
Step 2: Draw the transition table for all pair of states.
Step 3: Now split the transition table into two tables T1 and T2. T1 contains all final states, and T2 contains non-final states.
Step 4: Find similar rows from T1 such that:
1. $\delta$ (q, a) =p
2. $\delta$ (r, a) = p
That means, find the two states which have the same value of a and b and remove one of them.

Step 5: Repeat step 3 until we find no similar rows available in the transition table T1.
Step 6: Repeat step 3 and step 4 for table T2 also.
Step 7: Now combine the reduced T1 and T2 tables. The combined transition table is the transition table of minimized DFA.

Let us take an example to illustrate this:
Suppose we have finite automata -




In the first step, we will try to find unreachable states. From the diagram, we can say that q2 and q4 are the unreachable states. We will remove all unreachable states.
In the second step, we will construct the transition table for the rest of the states. q0 is the initial state, q3 and q5 are the final states.

| **States** | **Input Symbols**<br>**01** |
|---|---|
| q0 | q1 q3 |
| q1 | q0 q3 |
| *q3 | q5q5 |
| *q5 | q5 q5 |

In the third step, we will divide rows of transition table into two sets as:

1. One set contains those rows, which start from non-final states:

***Space for learners:***

| States | Input Symbols<br>01 |
| --- | --- |
| q0<br>q1 | q1<br>q3<br>q0<br>q3 |

2. Another set contains those rows, which starts from final states.

| States | Input Symbols<br>01 |
| --- | --- |
| q3<br>q5 | q5<br>q5<br>q5<br>q5 |

In fourth step, Set 1 has no similar rows so set 1 will be the same.
In fifth step, in set 2, row 1 and row 2 are similar since q3 and q5 transit to the same state on 0 and 1. So skip q5 and then replace q5 by q3 in the rest.

| States | Input Symbols<br>01 |
| --- | --- |
| q3 | q3<br>q3 |

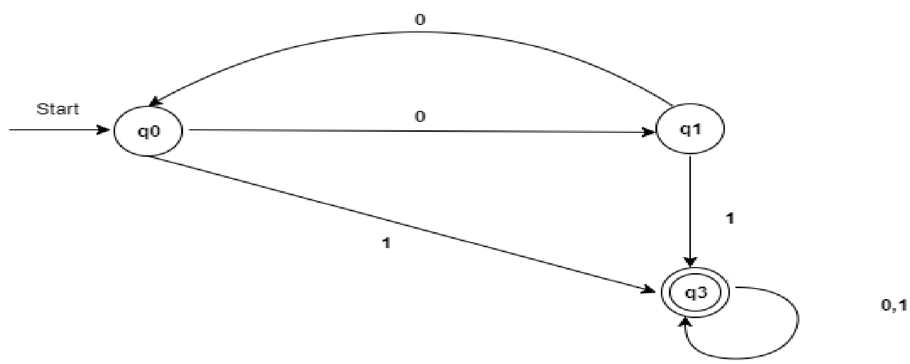
In sixth step, we will combine set 1 and set 2 as:

| States | Input Symbols<br>01 |
| --- | --- |
| q0 | q1 |

***Space for learners:***

| | |
|---|---|
| q1 | q3 |
| *q3 | q0 q3<br>q3q3 |

This is nothing but final minimized DFA transition table. Using this table, we can draw our DFA-




## 2.7 SUMMING UP

- For each input symbol, if our machine will move to only one certain state, it is a deterministic finite automaton
- If our machine is moving to any combination of states in the machine then it becomes non deterministic in nature.
- For describing complex problems, NFA is used and ε– NFA is a kind of NFA having epsilon feature, which helps states to move to other states or self state without input symbol.
- We can convert NFA, ε– NFA to its equivalent DFA.
- Some finite automata has large number of useless and redundant states, which consumes time and space both, that is why we learnt to reducing states of finite automata.

## 2.8 ANSWERS TO CHECK YOUR PROGRESS

Space for learners:

**1.** DFA: When our finite automata's state is determined i.e. for each input symbol, machine will move to only one certain state, hence it is called as deterministic finite automata. A deterministic finite automata consists of five tuples {**Q, Σ, $q_0$, F, δ**} where-

**Q** represents finite set of states.

**Σ** represents set of all input symbols, i.e. Alphabet.

**$q_0$** represents Initial state.

**F** represents finite set of final state/states.

**δ** represents Transition Function, which takes two arguments, a state and an input symbol, it returns a single state. So, **δ : Q X Σ -> Q.**

Some real-life examples of Deterministic Finite Automata (DFA) are lifts in buildings, text parsing, video game character behavior, security analysis etc.

NFA: When our finite automata's state is not determined i.e. machine can move to any states of the automaton. It consists of five-tuples {**Q, Σ, $q_0$, F, δ**} where-

**Q** represents set of all states.

**Σ** represents set of all input symbols, i.e., Alphabet.

**$q_0$** represents Initial state.

**F** represents set of final state or states.

**δ** represents Transition Function, which takes two arguments, a state and an input symbol, it returns any combination of Q states. So,**δ : Q X Σ -> $2^Q$.**

**2.** Transition Table contains the information regarding states and its input symbols. On different input symbols, different states of machine are moving to different states, these information are available in transition table.

For Example, typical transition tables look like-

| **States** | **Input Symbols**<br>**01** |
|---|---|
| a | b |

Space for learners:

| | |
|---|---|
| b | a<br>b<br>a |

**3.** Refer the section 2.3.3

**4.** Transition function of DFA is δ: Q X Σ -> Q, which means the function takes two arguments, a state and an input symbol, it returns a single state.

Transition function of NFA is δ : Q X Σ -> $2^Q$, which means the function takes two arguments, a state and an input symbol, it returns any combination of Q states.

**5.** Refer section 2.3.5

**6.** ε-closure means set of states that can be reached without any input symbol from any state of the ε-NFA.

**7.** Lazy creation of sets is a technique to convert a NFA to DFA. It used while we are constructing a DFA from given NFA. Unlike subset construction method, here we don't use to enumerate all the subsets of states. Whenever a new state arrived, we just calculate its mappings. When there is no new state, we draw equivalent DFA.

**8.** Refer section 2.4.1

**9.** From the theorem, we can say, A language L is recognized by a DFA if and only if there is an NFA N such that L (N) = L. So, NFA and DFA can recognize same set of languages.

## 2.9 POSSIBLE QUESTIONS

Question 1: Discuss about NFA with example.

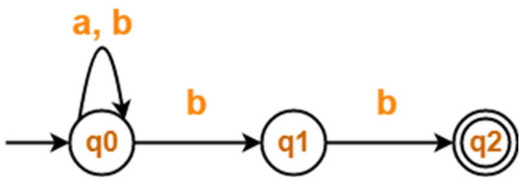
Question 2: Draw deterministic and non-deterministic finite automata which accept 00 and 11 at the end of a string containing 0, 1 in it.

Question 3: Write down the differences between NFA and DFA.

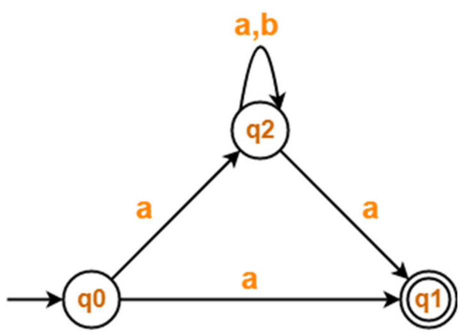
Question 4: Convert Following NFA to its equivalent DFA.

a)

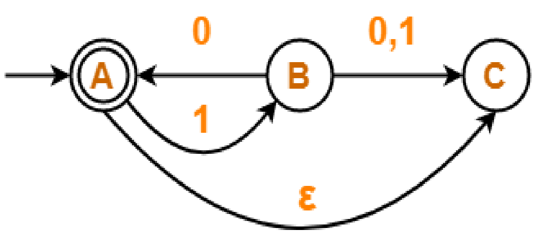
**Space for learners:**




b)

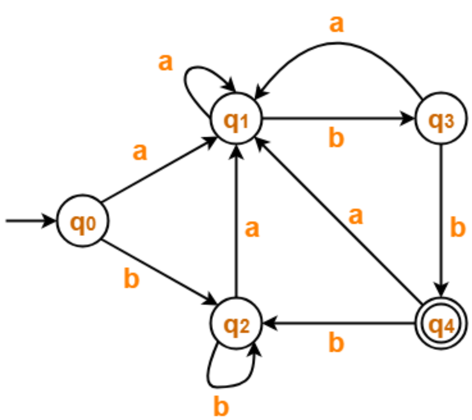



Question 5: Convert the following ε - NFA to its equivalent DFA.




Question 6: Write down the applications of Finite automata.

Question 7: Minimize states of the following FA.




***Space for learners:***

## 2.10 REFERENCES AND SUGGESTED READINGS

Space for learners:


- Introduction to Automata Theory, Languages and Computation, *John E Hopcroft ,Matwani& Jeffery D. Ullman*
- Introduction to Languages and the Theory of Computation, *John C. Martin*
- Elements of the Theory of Computation, *Lewis & Papadimitriou*
- GeeksforGeeks - https://www.geeksforgeeks.org/
- Gatevidyalay- https://www.gatevidyalay.com/
- Equivalence Theorem - https://www.neuraldump.net/
- Minimization Techniques - https://www.javatpoint.com/

# UNIT3: REGULAR SETS AND REGULAR EXPRESSIONS

Space for learners:

**Unit Structure:**

## 3.1 INTRODUCTION

Space for learners:

Regular Expressions are the simplest way to describe set of various strings in a language. Starting from our own utility software like compilers, game player etc to real life, regular expressions has enormous number of applications in various fields. That is why we need to study regular expressions, set and eventually regular language. This Unit basically discusses various definitions, various properties of regular language with examples in easy language. Finite automaton recognizes regular language, so this unit covers the way of designing DFA for its regular language. Regular language has some theorems like other languages, so for proving that we have used easy mathematical ways. Whenever we are studying different definitions and properties of different languages, those mathematical terms are very important in this subject. Only using mathematical terms, pumping lemma concept is introduced and it is very crucial to determine the type of the language. So, basically Regular language is nothing but regular sets which are generated from regular expressions using mathematical operations and this language can be recognized by finite automata.

## 3.2 UNIT OBJECTIVES

This unit covers all about regular expressions, sets and language and their operations. After going through this unit you will be able to –

- Get full concept on regular expressions with examples.
- Create any regular expressions, then language and eventually a DFA for it.
- What is the relation between NFA, DFA and regular expressions?
- How easily we can proof certain theorems using mathematical notations?
- Explain closure properties of regular language.
- Question about membership for any string in a regular language.
- Get to know about different algebraic laws of languages.

- Explain decision problems of regular language.
- Find a given language is regular or not, using Pumping Lemma.

**Space for learners:**

## 3.3 REGULAR SETS AND REGULAR EXPRESSIONS

Regular Expressions are an effective way to represent a language. These are nothing but string-like simple expressions which can be used to define any finite automata language. These expressions are generally a sequence of patterns. Any set which can be represented by regular expression is called a regular set.

Regular Expressions over an input alphabet $\Sigma$ are-

- $\phi$ is a regular expression denoting $\phi$.
- $\varepsilon$ is a regular expression denoting $\{\varepsilon\}$.
- For each $a \in \Sigma$, a is a regular expression denoting $\{a\}$.
- If a is regular expression, a* (0 or more times a) is also regular.
- If E and F are regular expressions denoting languages L(E) and L(F), then (E+F) is a regular expression denoting $L(E) \cup L(F)$
- If E and F are regular expressions denoting languages L(E) and L(F),then (EF) is a regular expression denoting L(E) L(F)
- If E and F are regular expressions denoting languages L(E) and L(F),then (E*) is a regular expression denoting L(E)*.

Example: Some regular expressions with their meanings are given below:

| Regular Expression | Meaning |
|---|---|
| 01 | A zero followed by a one(concatenation) |
| 0+1 | Either a zero or a one |
| 0* | Any number of zeroes |

| | |
|---|---|
| $1^+$ | One or more number of ones. |
| (0+1)* | All strings over {0,1} |
| 0*10*10* | Strings containing exactly two ones |
| (0+1)*11 | strings which end with two ones |

If a language can be expressed and described by regular expressions then it is known as regular language. And obviously, grammar of that language is called regular grammar.

### 3.3.1 Precedence Rules

Precedence rules of regular expression are similar to the rules of general arithmetic expression. We will consider exponentiation first, then multiplication, then addition. We will take Kleene closure as exponentiation, concatenation as multiplication, and union as addition and the precedence rules are identical.

### 3.3.2 Finite Automata and Regular Expressions

Regular expressions can define the same class of languages as finite automata. DFA, NFA, Epsilon-NFA all can express regular languages.

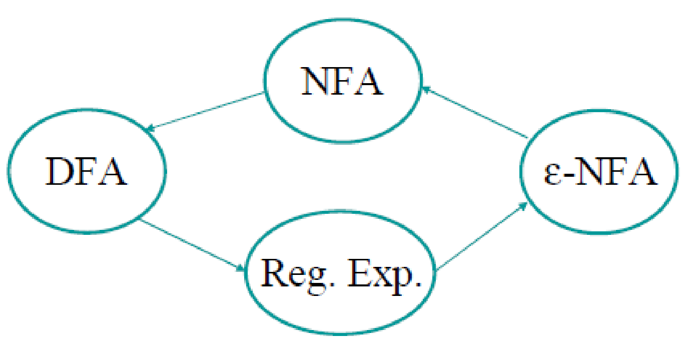



Figure 1: Relation of Finite automaton and regular expression

**_Space for learners:_**

**CHECK YOUR PROGRESS**

Question 1: What is a regular expression?

Question 2: How regular languages behave in case of union and concatenation operation?

Question 3: What is the difference between regular expression 0* and $0^+$.

Question 4: Discuss the relation between regular expressions and Finite automata.

Space for learners:

### 3.3.3 Inductive Definition of Regular Language

Base Case Definition: Let our input alphabet Σ, then {} is a regular language; {ε} is a regular language; {a} is a regular language for any character a in Σ.

Inductive Case Definition: If $L_1$ and $L_2$ are regular languages, then $L_1 \cup L_2$, $L_1L_2$, $L_1$* are also regular languages.

Completeness: Regular languages are those languages which can be generated using the above rules.

## 3.4 CLOSURE PROPERTIES OF REGULAR LANGUAGES

Regular expressions are defined through union, concatenation, and closure.

Union: $L_1 \cup L_2 = \{x \mid x \text{ in } L_1 \text{ or } x \text{ in } L_2\}$.

Concatenation: $L_1L_2 = \{xy \mid x \text{ in } L_1 \text{ and } y \text{ in } L_2\}$.

(Kleene) Closure: $L^* = \cup_{i=0,1,..\infty} L_i$, where $L_0 = \{\varepsilon\}$, $L_i = LL_{i-1}$

Example: {01, 10}* = {ε, 01, 10, 0110, 0101, 1010 …}

Since we knew that regular languages are closed under union, concatenation and star operation. So we will try to prove that regular languages are closed under intersection and complement operation as well.

Space for learners:

## 3.4.1 Complement

If $L_1$ is a regular language, then $\overline{L_1}$ is also a regular language.

Suppose we take a finite automata, let's say a DFA D that accepts $L_1$, and modifies every non-final states as final states and final states as non final. That means we are just complementing our DFA D. So, our new DFA $\overline{D}$ is nothing but the compliment of D, it will accept some strings because it has final and non final sates just like other DFAs. So, DFA $\overline{D}$ will accepts some new set of strings, eventually a language. By definition, we know that a language a DFA can recognize its nothing but a regular language. So, $\overline{L_1}$ is the regular language which will be recognized by our new DFA $\overline{D}$. Hence, if $L_1$ is a regular language, then $\overline{L_1}$ is also a regular language.

## 3.4.2 Intersection

For regular languages $L_1$ and $L_2$, their $\overline{L_1} \cap \overline{L_2}$ is also a regular language.

It is given that $L_1$ and $L_2$ are both regular languages, then -

By definition, $\overline{L_1}$ and $\overline{L_2}$ are also regular languages.

By definition, $\overline{L_1} \cup \overline{L_2}$ is a regular language.

By definition, complement of $\overline{L_1} \cup \overline{L_2}$ is also a regular language.

So, by applying de-Morgan's law, we can say $\overline{L_1} \cap \overline{L_2}$ is also a regular language.

Hence, regular languages are closed under union, intersection, concatenation, complement and star operation.

Space for learners:

**CHECK YOUR PROGRESS**

Question 5: Discuss Closure properties of regular language.

Question 6: How regular languages are closed under Complement and Intersection operation?

Question 7: What is Kleene Closure or Star Closure?

### 3.4.3 Algebraic Laws of Languages

Here are some important algebraic laws of Languages we need to know. If L, M and N are three regular languages, then -

- Union is commutative: $L \cup M = M \cup L$
- Union is associative: $(L \cup M) \cup N = L \cup (M \cup N)$
- Concatenation is associative: $(LM)N = L(MN)$
- $\phi$ is identity for union: $\phi \cup L = L \cup \phi = L$
- $\varepsilon$ is left and right identity for concatenation: $\{\varepsilon\}L = L\{\varepsilon\} = L$
- $\phi$ is left and right annihilator for concatenation: $\phi L = L \phi = L$
- Concatenation is left distributive over union : $L(M \cup N) = LM \cup LN$
- Concatenation is right distributive over union: $(M \cup N)L = ML \cup NL$
- Union is idempotent: $L \cup L = L$
- Star is idempotent: $(L^*)^* = L^*$
- $\phi^* = \{\varepsilon\}$, $\{\varepsilon\}^* = \{\varepsilon\}$
- $L^+ = LL^* = L^*L$, $L^* = L^+ \cup \{\varepsilon\}$

## 3.5 DECISION ALGORITHMS FOR REGULAR SETS

Decision Algorithms for a class of languages are properties which try to provide description of a language and discuss whether or not some properties hold. Decision problems can be solved very quickly, very computationally demanding, or unsolvable. Some decision properties of regular class of languages are given below-

- **Emptiness Problem:** Suppose we have given a regular language

L, how to check L is empty or not.

For this, we have to take the DFA for that regular language; we can easily draw the corresponding DFA for L. Now, we will check if there exists a path from initial state to final state. If there is a path, then it is not empty, otherwise it is empty one.

- **Finiteness Problem:** Suppose we have given a regular language L, how to check L is finite or not.

  For this, again we have to draw the DFA for that regular language. Now, we will check if there is a walk with cycle from initial state to final state. If there is at least one cycle in the path, then it is infinite and if there is no cycle present in the DFA, then L is finite.

- **Equivalence Problem:** Suppose we have given two regular languages $L_1$ and $L_2$, how to check if $L_1 = L_2$.

  We have to show the symmetric difference of $L_1$ and $L_2$ is empty that is, there is no string belonging to one but not both of the languages. So, symmetric difference of $L_1$ and $L_2$ can be expressed as: $(L_1 \cap \overline{L}_2) \cup (L_2 \cap \overline{L}_1)$

  Now we need to show $(L_1 \cap \overline{L}_2) \cup (L_2 \cap \overline{L}_1) = \phi$

  For getting $\phi$, we have to show $(L_1 \cap \overline{L}_2)$ and $(L_2 \cap \overline{L}_1) = \phi$

  By looking at the languages, $(L_1 \cap \overline{L}_2)$, if we can write $L_1 \subseteq L_2$ and

  By looking at the languages, $(L_2 \cap \overline{L}_1)$, if we can write $L_2 \subseteq L_1$, then we can conclude $L_1 = L_2$, or if any of these two $L_1 \subseteq L_2$ ,$L_2 \subseteq L_1$ is false, then we can conclude $L_1 \neq L_2$

- **Membership Problem:** Given a regular language L, we need to check a string suppose x is belongs to that L or not. Simplest solution for this problem is just to draw a DFA for regular language L and then check string x is accepted or not.

For example, L= $\{a^{3n} \mid n \geq 0\}$, which means the language contains strings of a's where count of numbers of a in the string is divisible by 3. Let's draw a DFA for it:

*Space for learners:*

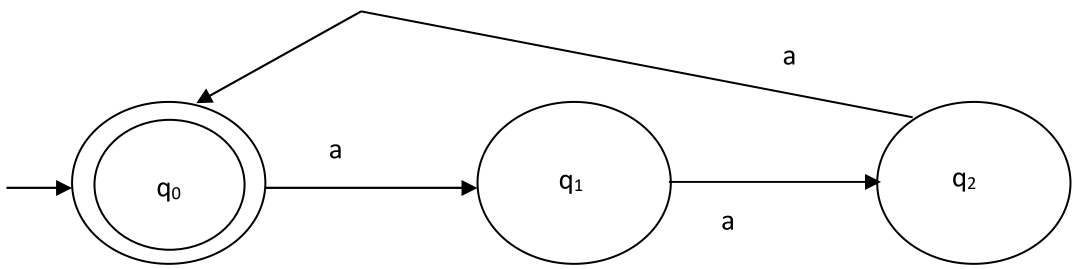



Figure 2: DFA for the Language L= $\{a^{3n} \mid n \geq 0\}$

From the above diagram, we can easily find out the acceptance status of different strings for this language. Suppose a string is ‘aaaaaa’, so, our initial state and final state is $q_0$, we will start moving from $q_0$, and the string is ending at $q_0$. Hence, this string belongs to the language. Now, take another example ‘aaaaa’, & from the diagram we can see that the string is ending at the state $q_2$. Hence, this string ‘aaaaa’ does not belong to the language. Since, we have learnt many topics about regular language, now we need to learn how to check a given language is regular or not. For this we will study Pumping Lemma in the next section.

## 3.6 PUMPING LEMMA FOR REGULAR LANGUAGES

Pumping lemma is used as a proof that the language is not regular. We used pumping lemma as a contradictory measure to proof a language is not regular, but if the language satisfies pumping lemma then it can be regular.

### 3.6.1 Pumping Lemma

Pumping lemma for Regular Language is -

For any regular language L, there exists an integer n, such that for all $w \in L$ with $|w| \geq n$, and $x,y,z \in \Sigma$, such that $w = xyz$,
(1) $|xy| \leq n$
(2)$y \neq \varepsilon$
(3) for all $i \geq 0$: $xy^iz \in L$

**_Space for learners:_**

In the last condition, we are pumping the string ‘y’. So whatever the value of i, i.e. string y can be inserted any number of times, but the resultant string should belongs to the language L.If there exists at least one string made from pumping which is not in L, then L is not regular.

Space for learners:

| CHECK YOUR PROGRESS |
| --- |
| Question 8: What are the decision problems or algorithms for regular language? |
| Question 9: State pumping lemma for regular language? |
| Question 10: Check ‘001100’ string is accepted or rejected by the language L= {x \| x ends with at least one zero}. |

## 3.6.2 Examples

Let us discuss some pumping lemma examples-

**Approach to solve:** Try to find a contradiction to prove that the language is not regular, if we able to do so, then the language is not regular, otherwise it is regular. That’s why; we will first assume that the language is a regular one.

Example 1: Checking irregularity of the language **L= $\{a^nb^n : n>=0\}$**

So, Let’s say our language L= $\{a^nb^n: n>=0\}$ is regular.

Let m be an integer, we will choose a string w such that w ∈ L and length $|w|\geq m$

Suppose w= $a^mb^m$, According to pumping lemma, w=xyz, so we will divide $a^mb^m$ into three parts and the middile part i.e. y we will pump. Conditions should be fulfilled i.e. $|xy| \leq m$, $|y|\geq 1$.

So, let’s divide it like - $x=a^{m-k}$ , $y=a^k$ , $z=b^m$ , where $|k|\geq 1$

From the pumping lemma $xy^iz \in L$ , i=0,1,2,3,……..

For i=2, $xy^2z = a^{m-k}a^{2k}b^m = a^{m+k}b^m$

BUT, $a^{m+k}b^m \notin L$, because our language is $L= \{a^nb^n: n>=0\}$, that means equal number of a's followed by equal number of b's. So, It's a contradiction and hence, the language is not regular.

Space for learners:

Example 2: Checking irregularity of the language **$L= \{ww \mid w \in\Sigma^*\}$**

So, let's say our language $L= \{ww \mid w \in\Sigma^*\}$ is regular. Here, from the L, we can easily understand that a string w is repeating twice, and the string w is taken from the input alphabet. Assuming our input alphabet $\Sigma =\{a,b\}$, we can take our string as $a^nba^nb$, where n is an integer.

So, According to pumping lemma, $w = a^nba^nb$.

So, let's divide it like - $x=a^{n/2}$, $y= a^{n/2}$ , $z=ba^nb$

We know to be a regular language, $xy^iz\in L$ , i=0,1,2,3,…….., this rule must be satisfied.

BUT, if we take $xy^0z = a^{n/2}ba^nb \notin L$.

For $xy^0z$, the same string is not repeating twice. It's a simple contradiction and hence, the language is not regular.

Example 3: Checking irregularity of the language **$L = \{ 0^i1^j \mid i > j\}$**

So, let's say our language $L = \{0^i1^j \mid i > j\}$ is regular. By looking at the language definition, we can find out this language has strings which have number of zeroes followed by number of ones, but number of zeroes should be greater than the number of ones. We can take our string as $0^{n+1}1^n$, where n is apositive integer number.

Like earlier example, lets divide our w = xyz, in such a way that $|xy| \leq n$, $|y|\geq 1$.

$x= 0$ , $y=0^{n-1}$ , $z=01^n$ , from the pumping lemma $xy^iz\in L$ , i=0,1,2,3,……..

If we check, $xy^2z = 0(0^{n-1})^2\, 01^n = 00^{2n-2}01^n = 0^{2n}1^n \in L$

BUT, if we pump down it, $xy^0z = 0(0^{n-1})^0\, 01^n = 00^001^n = 0^21^n \notin L$, when n>1 and n can be any integer. So, it's contradicting the definition of given language and hence, the language is not regular.

## 3.7 SUMMING UP

- Regular Expressions are nothing but strings, which can be used to express and describe a language, regular language.
- Finite automata recognized this regular class of languages.
- While generating regular sets from regular expressions, precedence rules are important to consider.
- Regular languages are closed under union, intersection, concatenation, star and complement operation. Using these mathematical properties we can proof many algorithms on regular languages.
- Like every class of languages, regular languages have also decision problems or algorithms.
- We use pumping lemma to prove that a language is not regular.

Space for learners:

## 3.8 ANSWERS TO CHECK YOUR PROGRESS

**1.** A regular expression is a string that describes the whole set of strings according to certain rules. Regular Expressions over an input alphabet Σ are-

- $\phi$ is a regular expression denoting $\phi$.
- $\varepsilon$ is a regular expression denoting $\{\varepsilon\}$.
- For each $a \in \Sigma$, a is a regular expression denoting $\{a\}$.
- If a is regular expression, a* (0 or more times a) is also regular.

**2.** If E and F are regular expressions denoting languages L(E) and L(F), then (E+F) is a regular expression denoting $L(E) \cup L(F)$ and (EF) is a regular expression denoting L(E) L(F).

**3.** 0* indicates the sets of any number of zeroes and $0^+$ indicates the sets of one or more number of zeroes.

**4.** Regular expressions can define the same class of languages as finite automata. DFA, NFA, Epsilon-NFA all can express regular languages. For any regular expressions, we can design corresponding finite automata.

**5.** Closure Properties of Regular languages are-

- Union: $L_1 \cup L_2 = \{x \mid x \text{ in } L_1 \text{ or } x \text{ in } L_2\}$.
- Concatenation: $L_1L_2 = \{xy \mid x \text{ in } L_1 \text{ and } y \text{ in } L_2\}$.
- (Kleene) Closure: $L^* = \cup_{i=0,1,..\infty} L_i$, where $L_0 = \{\varepsilon\}$, $L_i = LL_{i-1}$

Besides from that, Regular languages are also closed under intersection and complement.

**6.** Regular language is closed under Complement because If $L_1$ is a regular language, then $\overline{L_1}$is also a regular language. Regular language is closed under Intersection because for any two regular languages $L_1$ and $L_2$, their $\overline{L_1} \cap \overline{L_2}$is also a regular language.

**7.** Kleene or star Closure: If $L_1$ is a regular langauge, then $L_1^*$ (the Kleene closure of $L_1$) is also a regular language.

**8.** The decision problems for regular languages are-

a) Emptiness Problem

b) Finiteness Problem

c) Equivalence Problem

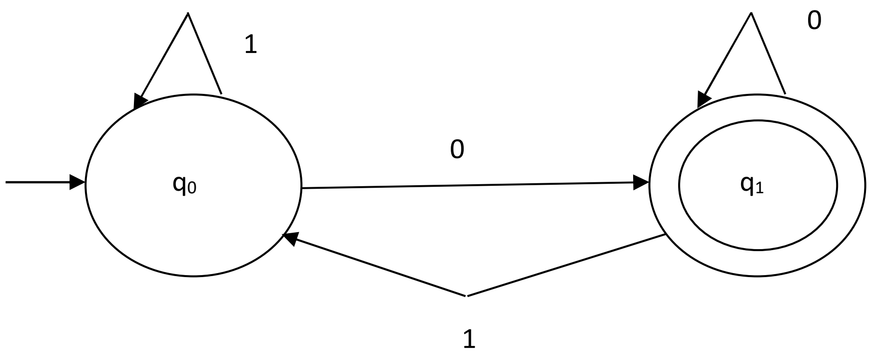
d) Membership Problem

**9.**Pumping lemma for Regular Language is -

For any regular language L, there exists an integer n, such that for all $w \in L$ with $|w| \geq n$, and $x,y,z \in \Sigma$, such that $w = xyz$,
(1) $|xy| \leq n$
(2) $y \neq \varepsilon$
(3) $xy^iz \in L$, for all $i \geq 0$;

**10.** We have given a language L= {x | x ends with at least one zero}, i.e a language which accepts strings which are ending with two zeroes. We need to check the string '001100' is either accepted or rejected. Let's design a DFA for this language:

***Space for learners:***




From the above diagram, we can easily find out the acceptance status of different strings for this language. From our Finite Automata, we can see, our initial state is $q_0$ and final state is $q_1$. We have given our string '001100', so we will start from $q_0$. It is moving like $q_0 \rightarrow q_1 \rightarrow q_1 \rightarrow q_0 \rightarrow q_0 \rightarrow q_1 \rightarrow q_1$, finally it is ending at $q_1$, which is our DFA's final state, so the string is accepted by the language.

## 3.9 POSSIBLE QUESTIONS

**Question 1:** Which one of the following languages over the alphabet {0,1} is described by the regular expression?
(0+1)*0(0+1)*0(0+1)*
a) The set of all strings containing the substring 00.
b) The set of all strings containing at most two 0's.
c) The set of all strings containing at least two 0's.
d) The set of all strings that begin and end with either 0 or 1.

**Question 2:** Regular expressions are closed under
a) Union
b) Intersection
c) Kleen star
d) All of the mentioned

**Question 3:** Which of the following is true?
a) (01)*0 = 0(10)*
b) (0+1)*0(0+1)*1(0+1) = (0+1)*01(0+1)*
c) (0+1)*01(0+1)*+1*0* = (0+1)*
d) All of the mentioned

Space for learners:

Space for learners:

**Question 4:** Write the regular expression for the language accepting all the string containing any number of a's and b's, over the input alphabet $\sum = \{a,b\}$.

**Question 5:** Check the language **L= {** $0^n$ **|** ***n is a prime number*}** is regular or not.

**Question 6:** We have studied that two regular languages are equal if they have the same regular expression representation or DFAs. Let $L_1$ and $L_2$ denote two regular languages, one of them is given to you as a regular expression while the other is represented as a DFA. How would you verify that they are equal?

**Question 7:** Discuss the closure properties of regular languages.

**Question 8:** Discuss the decision properties of regular languages.

**Question 9:** What will be the regular sets of the following?

(a) $(0+1)^*$ (b) $(01)^*$ (c) $(0+1)$ (d) $(0+1)^+$

**Question 10:** What are the applications of Regular expressions and Finite automata?

## 3.10 REFERENCES AND SUGGESTED READINGS


- Introduction to Automata Theory, Languages and Computation, *John E Hopcroft, Matwani & Jeffery D. Ullman*
- Introduction to Languages and the Theory of Computation, *John C. Martin*
- Elements of the Theory of Computation, *Lewis & Papadimitriou*
- Sanfoundry - https://www.sanfoundry.com/
- GeeksforGeeks- https://www.geeksforgeeks.org/

# UNIT 4: CONTEXT FREE LANGUAGE

Space for learners:

**Unit Structure:**

## 4.1 INTRODUCTION

Space for learners:

There are certain languages that cannot be described or expressed by finite automata, so we need more powerful mechanism which can recognize complex languages. The recursive structure of CFG is useful for recognizing some set of complex languages. CFG are used for basis of compiler design and implementation, computer vision, linguistics, specification mechanisms for programming languages. We can easily derive any string which belongs to the language of the grammar using derivation techniques. A context free grammar is very flexible because it can be simplified if there are any useless productions or symbols.

## 4.2 UNIT OBJECTIVES

This unit covers about context free languages, context free grammar and its derivation techniques. After going through this unit you will be able to:

- Explain about context free languages with examples.
- Create context free grammar for a language.
- Discuss the closure properties of context free languages.
- Find leftmost and rightmost derivation of strings.
- Draw parse tree of a string for a given grammar.
- Check a context free grammar is ambiguous or not.
- Find out useless productions, null productions and unit productions of any context free grammar.
- Simplify any context free grammar.

## 4.3 CONTEXT FREE LANGUAGE

Context free language (CFL) is a type of language which is generated by a context free grammar (or Type 2 grammar) i.e., a language L is context free if there is a context free grammar (CFG) G, such that L is generated from G. Regular languages are subset of context free languages. Just like finite automata, which can recognize the set of regular languages, Pushdown automata (PDA) can recognize context free languages. E.g. Arithmetic operations can be generated by context free grammars, so these are context free languages. Simply, Languages which are specified by context free grammars are called context free languages.

Space for learners:

## 4.3.1 Contexts Free Grammar

It is a formal grammar used to generate possible patterns of strings for a given language. A context free grammar (CFG) is a 4-tuple (V, **Σ** , R, S), where –

V is a set of non-terminals (NT) are also called variables, which are generally denoted by capital letters.

Σ is an alphabet, characters in the alphabet are known as terminals, which are generally denoted by lowercase letters.

R is a set of production or substitution rules that represents the recursive definition of the language. R is a subset of NT×(Σ∪NT)*. If (α , β) ∈ R, if we can write α → β, then α → β is a production rule, where α contains non-terminal symbols and β may contains terminals or non-terminal symbols or combination of terminal and non-terminal symbols.

S is the starting variable, which is used to derive the string and is one of the variables from the set of V.

## 4.3.2 CFG Notation

While defining CFG, we used some notations, these are-

- Uppercase letters are non-terminals (NT) and everything else is a terminal symbols.
- Start symbol is always from non-terminals set, it will be on the left hand side of the first production rule.
- Left hand side of the production rule only contains non-terminals, using which we derive strings.
- Right hand side of the production rule can be anything from

terminals and non-terminals set and ε together.

- Rules with common left hand sides are combined with right-hand sides separated by "|"

As an example consider the grammar:

$S \rightarrow xSy \mid \varepsilon$

The implication is start symbols is S and the rules are:

$S \rightarrow xSy$

$S \rightarrow \varepsilon$

## 4.3.3 Closure Properties of Context Free Language

Closure properties discuss about various operations on Context Free Language is closed. If we are doing an operation on a set and it always produces a member which of the same set type, then we can say the set is closed under that operation. Context free languages have the following closure properties –

Union: Context-free languages are closed under union operation i.e. that if X and Y are both context-free languages, then X∪Y is also a context-free language.

Concatenation: Context-free languages are closed under concatenation operation i.e. that if X and Y are both context-free languages, then XY is also a context-free language.

Kleene Star: Context-free languages are closed under concatenation operation i.e. that if L is a context free language then L* is also a context free language.

Unlike regular languages, Context free languages are not closed under intersection or complement operation.

For the decision properties of context free languages, emptiness problem, finiteness problem and membership problem all are decidable.

## 4.3.4 Examples

Space for learners:

Let's try to construct the CFG for the language having any number of a's over the set $\sum = \{a\}$.

We know the regular expression for above language a*.

Let's try to construct the production rules for this-

$S \rightarrow aS$

$S \rightarrow \varepsilon$

Where S is the starting variable i.e. a non terminal and a is the input symbol from set $\sum$ and ε is just a empty string. So, if we want to derive any number of a's then we will start from the starting variable –

S

aS

aaS S is replaced by aS because of first production rule

aaaS S is replaced by aS because of first production rule

aaaaS S is replaced by aS because of first production rule

aaaaaS S is replaced by aS because of first production rule

aaaaaaS S is replaced by aS because of first production rule

aaaaaaε S is replaced by ε because of second production rule

aaaaaa

So, from the derivation we can easily understand that we can get any number of a's. If we want to get zero number of a's, that means just a empty string, we will first choose second production rule, because S is our starting variable.

Now let's try to construct one intermediate CFG language L = { $wxw^R$ | where w € (a, b)*}, where, $w^R$ is a reverse string of w.

The string that can be generated for a given language is {aaxaa, bxb, abxba, baxab, abbxbba,….}

Production rules for the grammer can be –

$S \rightarrow aSa$

$S \rightarrow bSb$

$S \rightarrow x$

Space for learners:

Space for learners:

From these production rules, we can derive any string of {aaxaa, bxb, abxba, baxab, abbxbba,….}. Suppose for example, String 'abbxbba' can be derived as-

S → aSa

S → abSba S is replaced by bSb using second production rule

S → abbSbba S is replaced by bSb using second production rule

S → abbxbba S is replaced by x using third production rule

Since, at the last line, no non-terminals (in our example, only one NT is given, which is S) are there, hence this is our derived required string.

**CHECK YOUR PROGRESS**

Question 1: What is Context Free Grammar?

Question 2: Construct a CFG for the language $L = \{0^n1^n \mid n>1\}$

Question 3: Explain closure properties of Context Free language.

## 4.4 DERIVATION

A derivation is a sequence of steps which begins with the start symbol, uses the production rules to do replacements, and ends with a terminal string.

In one step derivation, u **yields** v in **one-step**, written like $u \Rightarrow v$, if for some u,v in $(V \cup \sum)^*$, $u = x\alpha z$and $v = x\beta z$where$\alpha \rightarrow \beta$is a rule.

In multistep derivation, u **derives** v, written like $u \Rightarrow^* v$, if there is a chain of one step derivations in the form:

$$u \Rightarrow u_1 \Rightarrow u_2 \Rightarrow u_3 \Rightarrow u_4 \Rightarrow u_5 \ldots\ldots \Rightarrow v$$

### 4.4.1 Leftmost and Rightmost Derivations

A leftmost derivation of a sentential form is one in which rules transforming the leftmost non terminal are always applied. Simply, in leftmost derivations, we will always replace the leftmost non-terminals.

A rightmost derivation of a sentential form is one in which rules transforming the rightmost non terminal are always applied. Simply, in rightmost derivations, we will always replace the rightmost non-terminals.

As for example, let's take a simple grammar, consider our earlier example of language a*. For that we have productions rules like-

S → aS

S → ε

Let us consider a string w =aaa

Leftmost Derivation-

S →a**S**

→ aa**S** (Using first production rule)

→ aaa**S** (Using first production rule)

→ aaaε (Using second production rule)

→ aaa

Rightmost Derivation-

S → a**S**

→ aa**S** (Using first production rule)

→ aaa**S** (Using first production rule)

→ aaaε (Using second production rule)

→ aaa

Hence, leftmost derivation = rightmost derivation

Leftmost and rightmost derivations are just two techniques to derive our strings. So, whatever the strings, if it belongs to the language, we can get it easily either by leftmost or rightmost derivations. These two derivations techniques will become very easy once we study the concept of parse tree.

***Space for learners:***

Space for learners:

## 4.4.2 Parse Tree

Parse tree or derivation tree is a geometrical representation of derivations. There always exist a parse tree corresponding to each leftmost derivation and rightmost derivation. A parse tree of a derivation $u \Rightarrow u_1 \Rightarrow u_2 \Rightarrow u_3 \ldots\ldots \Rightarrow v$ is a tree in which:

- Each internal node is labeled with a non-terminal symbol.
- Root node of a parse tree is the start symbol of the grammar.
- Each leaf node is labeled with a terminal symbol.
- If a rule T→ T1T2…Tn occurs in the derivation then T is a parent node of nodes labeled T1, T2, …, Tn

As for example, consider the following grammar-

$$S \rightarrow aB \mid bA$$

$$S \rightarrow aS \mid bAA \mid a$$

$$B \rightarrow bS \mid aBB \mid b$$

Let us consider a string w = aaabbabbba

Now, let us derive the string w using leftmost derivation.

**<u>Derivation-</u>**

S → a**B**

→ aa**B**B (Using B → aBB)

→ aaa**B**BB (Using B → aBB)

→ aaab**B**B (Using B → b)

→ aaabb**B** (Using B → b)

→ aaabba**B**B (Using B → aBB)

→ aaabbab**B** (Using B → b)

→ aaabbabb**S** (Using B → bS)

→ aaabbabbb**A** (Using S → bA)

→ aaabbabbba (Using A → a)

So, by looking at the required derived string, we will use our production rules. Let's draw a parse tree for this derivation. Our root node will be S, because S is a starting variable-

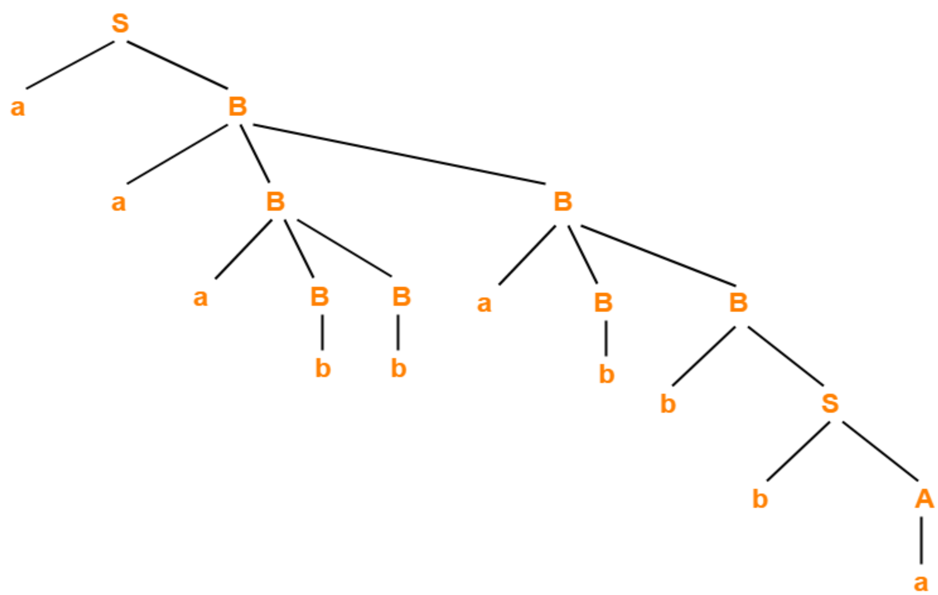



Figure: A Typical Parse Tree

From the figure, if we consider all leaf nodes from leftmost side, we get our string 'aaabbabbba'. So, we can derive strings from any language easily using parse tree that is why it is known as derivation tree.

**CHECK YOUR PROGRESS**

Question 4: What do you mean by leftmost and rightmost derivations?

Question 5: What is multiple steps derivation?

Question 6: What is parse tree? Draw parse tree for the string 'aaaa' for the following CFG:

S → aS

S → ε∈

***Space for learners:***

## 4.4.3 Ambiguous Grammar

A grammar G is ambiguous if there is a word w ∈L (G) having are least two different leftmost or rightmost derivations. Simply, for a string in a Context Free Grammar (CFG), more than one leftmost derivation and more than one rightmost derivation exist. For ambiguous grammar, there will be two or more sparse trees for a string. Let's figure this out with an example:

Suppose our grammar is:

$E \rightarrow E + E \mid E * E \mid (E) \mid N$

$N \rightarrow 1N \mid 2N \mid 1 \mid 2$

This is one of arithmetic operation type grammar, where we are using terminals like +, |, * etc. symbols. Let's try to draw parse tree for the string 1 + 2 * 2

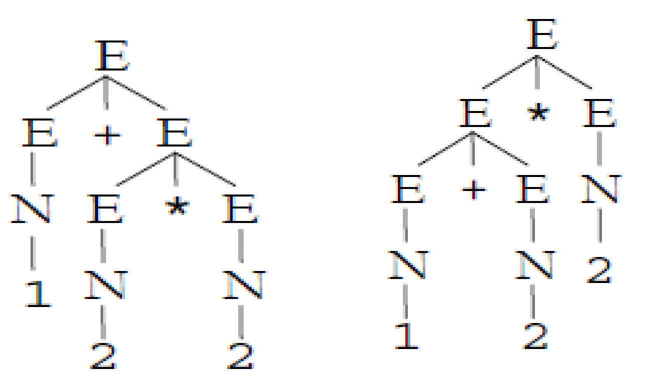



Figure 2: Two different parse tree for same string

Since for a string, the grammar has more than one parse tree, hence this grammar is an ambiguous grammar. Additionally, from the figure 2, if we calculate parse tree derivations from arithmetic point of view, then left parse tree value and right parse tree value will be 5 and 6 respectively. That is why ambiguity in grammar increases difficulties for parser exponentially.

Let us discuss another example -

Check whether the given grammar is ambiguous or not-

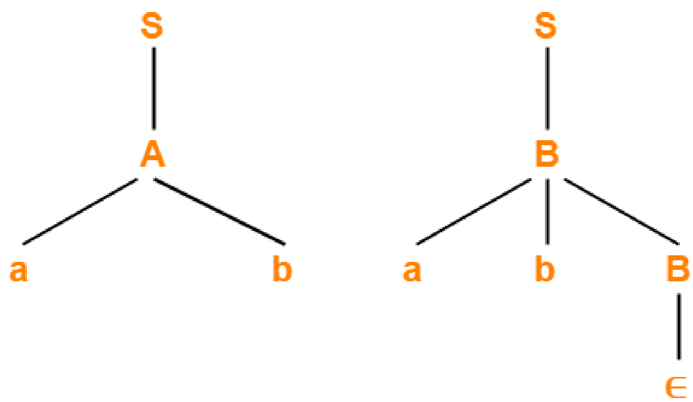
Space for learners:

$$S \rightarrow A \mid B$$

$$A \rightarrow aAb \mid ab$$

$$B \rightarrow abB \mid \varepsilon$$

Now, let us draw parse trees for this string ab –




Given grammar is ambiguous because two different parse trees exist for string ab.

## 4.5 SIMPLIFYING CONTEXT FREE GRAMMARS

While preparing context free grammar, we tend to write some unnecessary redundant productions because CFG allows us to develop a wide variety of grammars. That is why all the grammars are not always optimized i.e. grammar may consists of some useless symbols or productions. Simplification of CFG means reduction of grammar by removing unnecessary productions, while keeping the transformed grammar equivalent to the original grammar. Two grammars are called equivalent if they produce the same language.

### 4.5.1 Types of Redundant Productions

Useless productions: Productions which do not take part in the derivation of any string. Same is applicable for symbol or variable in context free grammar. Consider the following grammar –

$S \rightarrow aaB \mid aaS$

Space for learners:

$B \rightarrow ab \mid b$

$E \rightarrow ad$

Production $E \rightarrow ad$ will never come in the derivation of any string because it is not reachable from the starting variable S.

Null Productions: The productions of type $P \rightarrow \varepsilon$ are called null productions or ε productions (also called lambda productions). Null productions or ε productions are frequently used to develop context free grammar.

$S \rightarrow ABCd$

$A \rightarrow BC$

$B \rightarrow bB \mid \varepsilon$

$C \rightarrow cC \mid \varepsilon$

Productions $B \rightarrow \varepsilon$ and $C \rightarrow \varepsilon$ are both null productions and ε productions.

Unit Productions: The productions of type $P \rightarrow Q$ are called unit productions. Simply, the production where a non terminal implies another non terminal is known as unit productions. Consider the following grammar –

$S \rightarrow 0A \mid 11 \mid C$

$A \rightarrow 0S \mid 00$

$C \rightarrow 01$

Production $S \rightarrow C$ is a unit production in the above grammar.

## 4.5.2 Elimination of Useless Productions

We have studied about useless productions. Let's try to understand how to eliminate useless productions from a context free grammar with a proper example -

$T \rightarrow aaB \mid abA \mid aaT$
$A \rightarrow aA$
$B \rightarrow ab \mid b$
$C \rightarrow ad$

*Space for learners:*

Space for learners:

In the example, the production $C \rightarrow ad$ is useless, because C is not reachable from S, so it will never occur in the derivation of any string. So we will eliminate it.

Production $A \rightarrow aA$ is also useless because we don't have any way to terminate it. If a production never terminates, then it can never produce a string. To remove this useless production $A \rightarrow aA$, we will first find all the variables which will never lead to a terminal string such as variable 'A'. Then we will remove all the productions in which the variable 'A' occurs. So, after removing useless symbols and productions, grammar will be –

$T \rightarrow aaB \mid aaT$
$B \rightarrow ab \mid b$

## 4.5.3 Elimination of Null Or E Productions

We have studied about null or ε productions. For removing null productions from the grammar, we need to do -

Step 1: Find out all non-terminal variables which derives ε, those non terminals are also known as nullable variables.

Step 2: For each production, which contains nullable variables, construct new productions by replacing nullable variables.

Step 3: Combine productions of step 2 with the original productions and remove ε productions.

Consider the following grammar:

$S \rightarrow XYX$

$X \rightarrow 0X \mid \varepsilon$

$Y \rightarrow 1Y \mid \varepsilon$

We need to remove the production rules $X \rightarrow \varepsilon$ and $Y \rightarrow \varepsilon$. To preserve the meaning of CFG we are actually placing ε at the right-hand side whenever X and Y have appeared, so we need to check every possibility while removing ε.

$S \rightarrow XYX$

If the first X at right-hand side is ε and the last X at right-hand side is ε, then we can write

$S \rightarrow YX$

$S \rightarrow XY$

If $Y = \varepsilon$ then

$S \rightarrow XX$

If Y and X are ε then,

$S \rightarrow X$

If both X are replaced by ε

$S \rightarrow Y$

Now,

$S \rightarrow XY \mid YX \mid XX \mid X \mid Y$

Let's take another production rule,

$X \rightarrow 0X$

If we place ε at right-hand side for X then,

$X \rightarrow 0$

$X \rightarrow 0X \mid 0$

Similarly in case of last production rule,

$Y \rightarrow 1Y \mid 1$

So, after removing null productions, our CFG will look like –

$S \rightarrow XY \mid YX \mid XX \mid X \mid Y$

$X \rightarrow 0X \mid 0$

$Y \rightarrow 1Y \mid 1$

**CHECK YOUR PROGRESS**

Question 7: What is ambiguous grammar?

Question 8: Why we need to simplify context free grammar? Explain.

Question 9: What are the different types of redundant productions?

Question 10: Check the following CFG is ambiguous or not?

$S \rightarrow aSb \mid SS \quad , \quad S \rightarrow \varepsilon$

***Space for learners:***

## 4.5.4 Elimination of Unit Productions

Space for learners:

For removing unit productions i.e. productions of type $X \rightarrow Y$, we need to follow -

**Step 1:** To remove $X \rightarrow Y$, add production $X \rightarrow a$ to the grammar rule whenever $Y \rightarrow a$ occurs in the grammar.

**Step 2:** Now delete $X \rightarrow Y$ from the grammar.

**Step 3:** Repeat step 1 and step 2 until all unit productions are removed.

Considering the following grammar:

$S \rightarrow 0A \mid 1B \mid C$

$A \rightarrow 0S \mid 00$

$B \rightarrow 1 \mid A$

$C \rightarrow 01$

In the above example, $S \rightarrow C$ is a unit production, while removing $S \rightarrow C$ we have to consider what C implies. Depending on that, we can add a rule to S.

$S \rightarrow 0A \mid 1B \mid 01$

In the above example, $B \rightarrow A$ is also a unit production-

$B \rightarrow 1 \mid 0S \mid 00$

Thus finally our CFG without unit production is –

$S \rightarrow 0A \mid 1B \mid 01$

$A \rightarrow 0S \mid 00$

$B \rightarrow 1 \mid 0S \mid 00$

$C \rightarrow 01$

## 4.6 SUMMING UP

- Context free languages are the languages which are specified by context free grammars.
- Context free grammar is developed to address a complex set of languages, as we have studied it's a 4-tuple grammar.
- For obtaining a string from a CFG, derivations techniques we need such as leftmost derivations, rightmost derivations etc...

- Parse tree is a geometrical representation of derivation of a string, if for a particular string, there is more than one parse tree then the corresponding grammar is ambiguous.
- We have to eliminate the ambiguity nature of the grammar.
- Sometimes parser faces problem because of the unnecessary productions present in context free grammar that is why we need to check for unnecessary productions in a grammar and if present we have to remove those productions.

Space for learners:

## 4.7 ANSWERS TO CHECK YOUR PROGRESSES

**1.** A context free grammar (CFG) is a 4-tuple (V, **Σ** , R, S) grammar, whereV is a set of non-terminals (NT) are also called variables, Σ is an alphabet, characters in the alphabet are known as terminals and S is the starting variable. R is a set of production or substitution rules that represents the recursive definition of the language.

**2.**Given CFG language is $L = \{0^n1^n \mid n \geq 1\}$.

The string that can be generated for a given language is {01, 0011, 000111, 00001111,….}

Production rules for the grammar can be –

$S \rightarrow 0S1$

$S \rightarrow 01$

From these production rules, we can derive any string of {01, 0011, 000111, 00001111…}. Suppose for example, String '000111' can be derived as-

$S \rightarrow 0S1$

$S \rightarrow 00S11$ Usingfirst production rule

$S \rightarrow 000111$ Using second production rule

**3.** Closure properties of Context free languages are –

Union: Context-free languages are closed under union operation i.e. that if X and Y are both context-free languages, then X∪Y is also a context-free language.

Concatenation: Context-free languages are closed under concatenation operation i.e. that if X and Y are both context-free languages, then XY is also a context-free language.

Kleene Star: Context-free languages are closed under concatenation operation i.e. that if L is a context free language then L* is also a context free language.

Unlike regular languages, Context free languages are not closed under intersection or complement operation.

**4.**A leftmost derivationof a sentential form is one in which rules transforming the leftmost non terminal are always applied. A rightmost derivationof a sentential form is one in which rules transforming the rightmost non terminal are always applied.

**5.**In multiple steps derivation, u derivesv, i.e.u $\Rightarrow^{*}$v, if there is a chain of one step derivations in the form:u$\Rightarrow u_1 \Rightarrow u_2 \Rightarrow u_3 \Rightarrow u_4 \Rightarrow u_5 \ldots\ldots \Rightarrow$ v

**6.**A parse tree is a geometrical representation of derivations in which:

- Each internal node is labeled with a non terminal symbol.
- Root node of a parse tree is the start symbol of the grammar.
- Each leaf node is labeled with a terminal symbol.
- If a rule T→ T1T2…Tn occurs in the derivation then T is a parent node of nodes labeled T1, T2, …, Tn

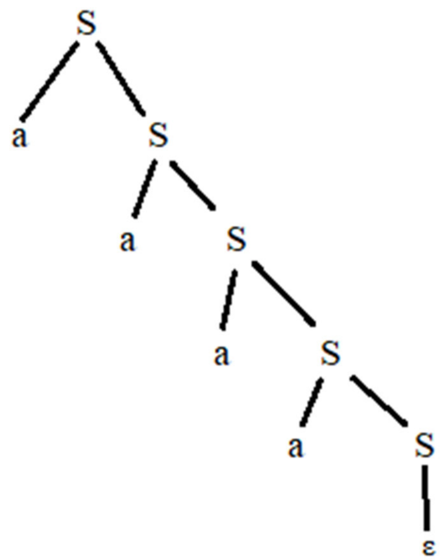
We need to draw a parse tree for a string ‘aaaa’ for a given grammar:

$S \rightarrow aS$

$S \rightarrow \varepsilon$

So, parse tree is -

**Space for learners:**




Space for learners:

**7.** For a string x in a Context Free Grammar (CFG), if there exist more than one leftmost derivation or rightmost derivations, then it is a ambiguous grammar.

**8.** Simplification of CFG means reduction of grammar by removing unnecessary productions, while keeping the transformed grammar equivalent to the original grammar. Simplification is required because all the grammars are not always optimized i.e. grammar may consists of some redundant symbols or productions.

9. Redundant productions are –

Useless productions: Productions or symbols which do not take part in the derivation of any string.

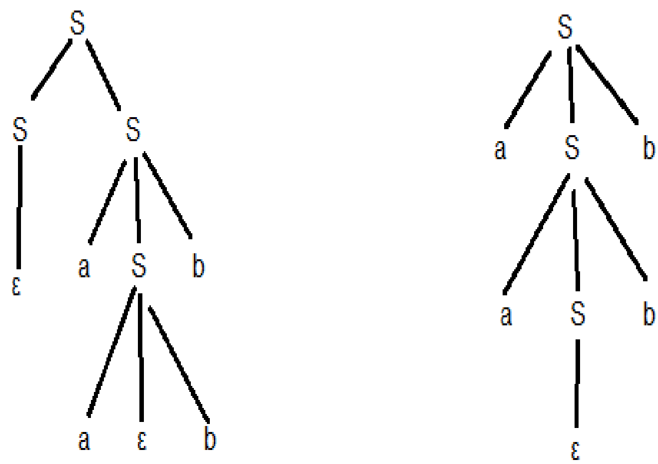
Null Productions: The productions of type P → ε are called null productions or ε productions (also called lambda productions). Null productions or ε productions are frequently used to develop context free grammar.

Unit Productions: The productions of type P → Q are called unit productions.

**10.** We have given following grammar:

S → aSb | SS

S → ε




So, for the string ‘aabb’, there are two parse trees, hence given CFG is an ambiguous grammar.

## 4.8 POSSIBLE QUESTIONS

1. What is Context Free Language?
2. Define Context free grammar. Write some applications of it.
3. Explain the concept of parse tree with suitable example.
4. Explain Closure properties of context free languages.
5. Discuss various derivation techniques of CFG.
6. What are the three ways to simplify a context free grammar?
7. Discuss the simplification ways of Context free grammar with examples.
8. Eliminate useless productions from the following grammar:

   T → abA | aaT

   A → aA

   C → ad
9. Check the following grammar is ambiguous or not:

Space for learners:

$E \rightarrow E + E \mid E * E \mid (E) \mid id$

10. Construct CFG without Є production from the grammar:

$S \rightarrow a \mid Ab \mid aBa$ , $A \rightarrow b \mid \epsilon$ , $B \rightarrow b \mid A$.

## 4.9 REFERENCES AND SUGGESTED READINGS


- Introduction to Automata Theory, Languages and Computation, *John E Hopcroft, Matwani & Jeffery D. Ullman*
- Introduction to Languages and the Theory of Computation, *John C. Martin*
- Elements of the Theory of Computation, *Lewis & Papadimitriou*
- http://infolab.stanford.edu/
- https://www.geeksforgeeks.org/
- https://www.gatevidyalay.com/
- https://www.cs.wcupa.edu/
- https://www.javatpoint.com/


***Space for learners:***

# UNIT 5: PDA AND CHOMSKY NORMAL FORMS

Space for learners:

**Unit Structure:**



## 4.1 INTRODUCTION

In this unit we will study thoroughly about pushdown automata, normal forms of context free grammar and lastly the pumping lemma for context free languages. Finite automata cannot able to implement complex problems, that is why pushdown automata comes with an additional element called stack. Pushdown automata can implement a context free languages and stack is used for different mechanisms and memorizing purpose. We can design pushdown automata for any context free languages. In last chapter, we have studied about simplification of CFG, we will study another advanced related topic in this chapter, which is Normal forms of CFG. Normal forms deals with certain rules or forms of writing productions in the grammar. Lastly, we will study the pumping lemma for context free languages, applying which we can find out a language is context free or not.

Space for learners:

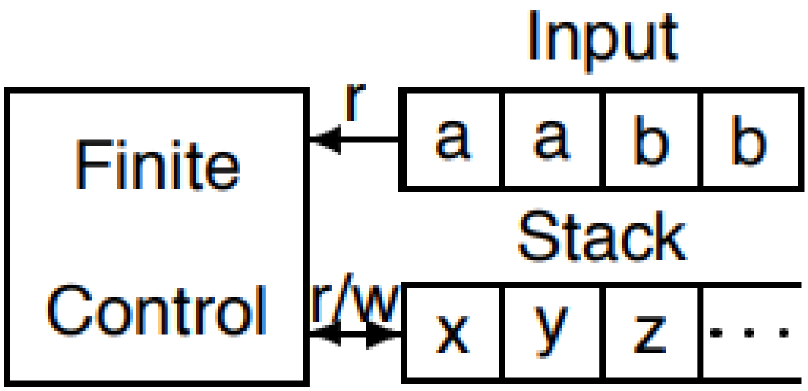
## 4.2 UNIT OBJECTIVES

This unit covers Pushdown automata, Normal forms of context free grammar and pumping lemma of Context free languages. After going through this unit you will be able to:

- How Pushdown automata work?
- Designing Pushdown automata for any context free languages.
- Explain about normal forms of context free grammar.
- Convert a context free grammar into its Chomsky's Normal Form (CNF).
- Discuss about Chomsky's Normal Form(CNF) and Greibach Normal Form (GNF)
- Convert a context free grammar into its Greibach Normal Form (CNF).
- Discuss the pumping lemma for context free languages.
- Check a language is context free language or not.

## 4.3 PUSHDOWN AUTOMATA (PDA)

Just as we design DFA for a regular grammar, pushdown automata (PDA) is a way to implement a context free grammar. Pushdown Automata are new type of computational model, which is like finite automata but have an extra memory component called stack. Stack allows PDA to recognize some complex languages that is why A PDA is more powerful than finite automata (FA). A language which can be acceptable by finite automata (FA) can also be acceptable by pushdown automata (PDA). We can visualize a PDA like-




Space for learners:

PDA reads input symbol from alphabet and it can read/write to stack. It makes transitions based on input symbol and top of stack.

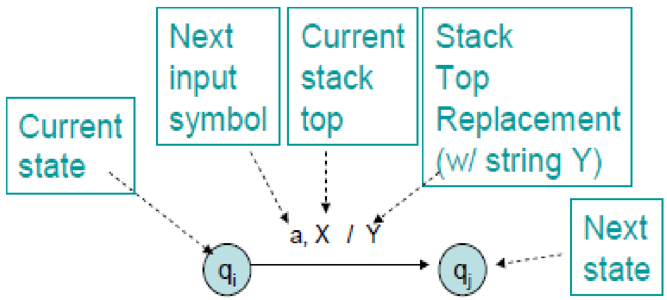
Formally, a PDA can be defined by 7-tuple (Q, $\sum$, $\Gamma$, $\delta$, $q_0$ ,$Z_0$, F), where:

- Q is the finite number of states
- $\sum$ is the finite set of input symbols, the alphabet
- $\Gamma$ is the finite set of stack symbols, symbols which are allowed to push/pop into the stack
- $q_0$ is the initial state of PDA
- $Z_0$ is the initial stack symbol
- F is the set of final states
- $\delta$ is a transition function: $Q \times \{\Sigma \cup \in\} \times \Gamma \rightarrow Q \times \Gamma^*$ , i.e. PDA will read input symbol and stack symbol (top of the stack) and move to a new state and change the symbol of stack.

## 4.3.1 PDA as a State Diagram

Space for learners:

$\delta(q_i, a, X) = \{(q_j, Y)\}$




## 4.3.2 Instantaneous Description

An instantaneous description of PDA is described by a triple (q, w, α) where:

**q** is the current state.

**w** is the unconsumed input.

**α** is the stack contents.

For transition purpose, we use turnstile notation (⊢ sign), which represents one move of PDA. And for multiple moves, we use ⊢* sign.

For example,

$(q, aw, X\beta) \vdash (p, w, \alpha\beta)$

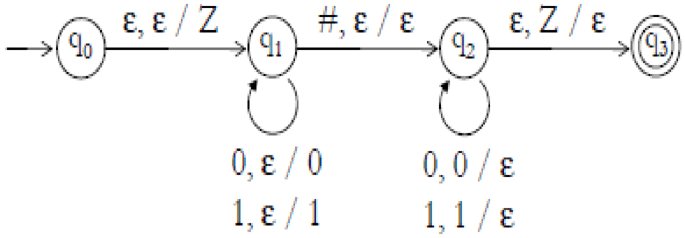
In the above example, we took a transition such that we went from state q to p, we consumed input symbol a, and we replaced the top of the stack X with some new string α.

## 4.3.3 Examples of PDA

**Example 1:** Design a PDA for the language $L = \{w\#w^R : w \in \{0,1\}^*\}$

Solution: From the given language, we can say our strings will look like #, 0#0, 01#10, 0110#0110 etc.

We can design the PDA using state diagrams only. For better understanding of PDA, we will use state diagrams for constructing PDA, because state diagrams are best mathematical tools for designing PDA.




For constructing PDA for a language, we need to think different mathematical mechanisms. There can be numerous numbers of mechanisms for designing a particular PDA. Here we designed this PDA with the mechanism – 'write w on stack and read $w^R$ from the stack' . In this PDA, we are assuming our stack alphabet is {0, 1} and initial stack symbol is Z. So, from $q_0$ to $q_1$, we just push a Z into the stack and then at the state $q_1$, we are pushing 1's and 0's for input symbol 1's and 0's, that means we are writing 'w' at $q_1$, then by consuming input symbol #, we reached at $q_2$, here we are popping 1's and 0's for input symbol 1's and 0's. Since our language is of type $w\#w^R$, that is why first we pushed one part and secondly, we popped the other part. If any string is not in type of $w\#w^R$, then machine will never go to the final sate. Hence, we designed our PDA for the given language.

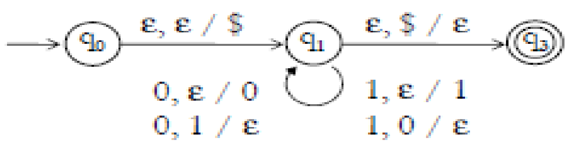
**Example 2:** Design a PDA for the language L = {w :w has same number of 0's and 1's}

**Solution:** From the given language, we can say our strings will look like 01, 0110, 011100, 001110 etc., so our input alphabet will be Σ = {0,1}.

As we have seen in the earlier example, we need to develop a mechanism for constructing PDA. Suppose, our stack is keeping track of number of 0's and 1's in the string and if we pop 1 for consuming input symbol 0 and vice versa and finally at the end, if we find our stack

**Space for learners:**

is empty, then we can easily say number of 0's and 1's are equal. So, let's construct it using state diagram -




For the above PDA, let's check membership of a string for the given language. Suppose our string is w= 001110, now for each consumed input symbol, our stack contents will be –

| Input symbol | Stack contents |
|---|---|
| 0 | $0 |
| 0 | $00 |
| 1 | $0 |
| 1 | $ |
| 1 | $1 |
| 0 | $ |

So, finally PDA's stack is empty, it will move to the state $q_3$ and which is a final state. Hence, this string is accepted.

**CHECK YOUR PROGRESS**

Question 1: What are Pushdown automata?

Question 2: What is the transition function of PDA?

Question 3: Can we construct a PDA without its state diagram?

Question 4: Construct a PDA for the language $L = \{w\#w^R : w \in \{0,1\}^*\}$

Space for learners:

## 4.4 NORMAL FORMS

Space for learners:

Generally, it's easier to work with context free grammar when it is in normal forms. While parsing in computer, sometimes CFG causes lots of problem such as redundant loops, infinite loops etc. that is why normal forms are often convenient to simplify CFG. There are mainly two normal forms, these are:

### 4.4.1 Chomsky Normal form (CNF)

A Context free Grammar G is in Chomsky Normal Form where every production is either of the form:

$$A \rightarrow BC$$

$$A \rightarrow a$$

Where a is a terminal and A,B,C are non terminals.

E.g. consider the following grammar G

$S \rightarrow AB$

$S \rightarrow c$

$A \rightarrow a$

$B \rightarrow b$

Production rules of Grammar G are in the forms of CNF, so grammar G is in CNF.

When a CFG is not in the form of Chomsky's Normal Form (CNF), then we need to convert it. The conversion requires some easy steps, which are –

Step 1: Remove the start symbol from RHS of production. If the start symbol S is at the right-hand side of any production, create a new production as:

$S1 \rightarrow S$, where S1 is the new start symbol.

Step 2: Remove null, useless and unit productions if needed.

Step 3:Replace terminals from the RHS of the production if they exist with other non-terminals or terminals. For example, the production $X \rightarrow aP$ can be written as:

$X \rightarrow QP$

$Q \rightarrow a$

Step 4: Productions which are having more than two non terminals, change it in the form $A \rightarrow BC$.

For example, $S \rightarrow ASB$ can be decomposed as:

$S \rightarrow QS$

$Q \rightarrow AS$

Example: Consider the following grammar:

$S \rightarrow ASB$

$A \rightarrow aAS|a|\varepsilon$

$B \rightarrow SbS|A|bb$

We need to convert this grammar to its CNF form.

So according to step 1, this grammar has start symbol in the RHS, we need to remove them.

$S1 \rightarrow S$

$S \rightarrow ASB$

$A \rightarrow aAS|a|\varepsilon$

$B \rightarrow SbS|A|bb$

Now, from step 2, we need to simplify our CFG by removing null, unit and useless productions, and this grammar has null productions –

$S1 \rightarrow S$

$S \rightarrow ASB|SB$

$A \rightarrow aAS|aS|a$

$B \rightarrow SbS| A|\varepsilon|bb$

So, it creates a new null production $B \rightarrow \varepsilon$, we need to remove it –

$S1 \rightarrow S$

Space for learners:

S → AS|ASB| SB| S

A → aAS|aS|a

B → SbS| A|bb

Now, it creates unit production B->A

S1 → S

S → AS|ASB| SB| S

A → aAS|aS|a

B → SbS|bb|aAS|aS|a

Now, another unit production is S1 → S is there –

S1 → AS|ASB| SB| S

S → AS|ASB| SB| S

A → aAS|aS|a

B → SbS|bb|aAS|aS|a

Again, S1 → S and S → S exists, after removing them –

S1 → AS|ASB| SB

S → AS|ASB| SB

A → aAS|aS|a

B → SbS|bb|aAS|aS|a

Now, applying rule of step 3 in the production rule A->aAS |aS and B->SbS|aAS|aS

S1-> AS|ASB| SB

S → AS|ASB| SB

A → XAS|XS|a

B → SYS|bb|XAS|XS|a

X →a

Y→b

In the fourth production, B->bb can't be part of CNF

S1-> AS|ASB| SB

Space for learners:

$S \rightarrow AS|ASB|\ SB$

$A \rightarrow XAS|XS|a$

$B \rightarrow SYS|VV|XAS|XS|a$

$X \rightarrow a$

$Y \rightarrow b$

$V \rightarrow b$

Now, according to step 4, in the production rule S1->ASB, we will get –

S1-> AS|PB| SB

$S \rightarrow AS|ASB|\ SB$

$A \rightarrow XAS|XS|a$

$B \rightarrow SYS|VV|XAS|XS|a$

$X \rightarrow a$

$Y \rightarrow b$

$V \rightarrow b$

$P \rightarrow AS$

Similarly, we will do the needful for the productionS->ASB,

S1-> AS|PB| SB

$S \rightarrow AS|QB|\ SB$

$A \rightarrow XAS|XS|a$

$B \rightarrow SYS|VV|XAS|XS|a$

$X \rightarrow a$

$Y \rightarrow b$

$V \rightarrow b$

$P \rightarrow AS$

$Q \rightarrow AS$

Again for the production A->XAS,

S1-> AS|PB| SB

Space for learners:

$S \rightarrow AS|QB|\ SB$

$A \rightarrow RS|XS|a$

$B \rightarrow SYS|VV|XAS|XS|a$

$X \rightarrow a$

$Y \rightarrow b$

$V \rightarrow b$

$P \rightarrow AS$

$Q \rightarrow AS$

$R \rightarrow XA$

Again for B->SYS,

S1 -> AS|PB| SB

$S \rightarrow AS|QB|\ SB$

$A \rightarrow RS|XS|a$

$B \rightarrow TS|VV|XAS|XS|a$

$X \rightarrow a$

$Y \rightarrow b$

$V \rightarrow b$

$P \rightarrow AS$

$Q \rightarrow AS$

$R \rightarrow XA$

$T \rightarrow SY$

Lastly for B->XAX, Now the grammar will look like –

S1-> AS|PB| SB

$S \rightarrow AS|QB|\ SB$

$A \rightarrow RS|XS|a$

$B \rightarrow TS|VV|US|XS|a$

$X \rightarrow a$

$Y \rightarrow b$

**Space for learners:**

$V \rightarrow b$

$P \rightarrow AS$

$Q \rightarrow AS$

$R \rightarrow XA$

$T \rightarrow SY$

$U \rightarrow XA$

So, this grammar satisfies the conditions of Chomsky's Normal form(CNF), Hence the grammar is in CNF.

## 4.4.2 Greibach Normal Form (GNF)

Context free Grammar G is in Greibach Normal Form(GNF) where every production is of the form:

$$A \rightarrow a\alpha$$

Where a is a terminal and $\alpha$ consists of any number of non terminals and If $\varepsilon$ is in the language, then we will allow the rule $S \rightarrow \varepsilon$.

For example, consider a grammar G –

$S \rightarrow aAB \mid aB$

$A \rightarrow aA \mid a$

$B \rightarrow bB \mid b$

In the grammar G, every productions are in the form of $A \rightarrow a\alpha$, hence grammar G is in GNF.

If we need to convert any context free grammar into its Greibach normal form(GNF), then –

Step 1: Convert given context free grammar into CNF. (Since we have studied this part earlier, so we will use this approach. There are some alternative approaches are available for GNF conversion)

Step 2: If CFG contain left recursions, then remove them. (A production of context free grammar is said to have left recursion if the leftmost variable of its RHS is same as variable of its LHS. E.g. $S \rightarrow Sa$ )

Space for learners:

Space for learners:

Step 3: Finally convert the production rules into GNF.

Example: Consider the following grammar –

S → XA|BB

B → b|SB

X → b

A → a

Step 1: We need to convert the grammar into CNF but every productions of the grammar are in CNF. So, lets move to step 2

Step 2: There is no left recursion in the grammar, so we can convert this grammar to GNF.

Step 3: We need to check for the productions that are not in GNF, then we will convert it one by one.

The production rule B->SB is not in GNF-

S → XA|BB

B → b|XAB|BBB

X → b

A → a

So, we substituted S -> XA|BB in production rule B->SB.

The production rules S->XA and B->XAB is not in GNF -

S → bA|BB

B → b|bAB|BBB

X → b

A → a

So, we substituted X->b in production rules S->XA and B->XAB.

Now, B->BBB production is a left recursive production, we need to remove that-

S → bA|BB

B → bC|bABC

$C \rightarrow BBC|\ \varepsilon$

$X \rightarrow b$

$A \rightarrow a$

We got another problem, because C-> ε is a null production, after removing this -

$S \rightarrow bA|BB$

$B \rightarrow bC|bABC|b|bAB$

$C \rightarrow BBC|BB$

$X \rightarrow b$

$A \rightarrow a$

The production rules S->BB is not in GNF -

$S \rightarrow bA|\ bCB|bABCB|bB|bABB$

$B \rightarrow bC|bABC|b|bAB$

$C \rightarrow BBC|BB$

$X \rightarrow b$

$A \rightarrow a$

So, we substituted $B \rightarrow bC|bABC|b|bAB$ in production rules S->BB.

The production rules C->BB is not in GNF-

$S \rightarrow bA|\ bCB|bABCB|bB|bABB$

$B \rightarrow bC|bABC|b|bAB$

$C \rightarrow BBC$

$C \rightarrow bCB|bABCB|bB|bABB$

$X \rightarrow b$

$A \rightarrow a$

So, we substituted $B \rightarrow bC|bABC|b|bAB$ in production rules C->BB.

The production rules C->BBC is not in GNF -

$S \rightarrow bA|\ bCB|bABCB|bB|bABB$

$B \rightarrow bC|bABC|b|bAB$

Space for learners:

$C \rightarrow bCBC|bABCBC|bBC|bABBC$

$C \rightarrow bCB|bABCB|bB|bABB$

$X \rightarrow b$

$A \rightarrow a$

So, we substituted $B \rightarrow bC|bABC|b|bAB$ in production rules C->BBC.

Now, finally every productions of this grammar are in greibach normal form (GNF).

## 4.5 Pumping Lemma of Context Free Languages

Just like regular language's pumping lemma, We can use pumping lemma for Context free languages to check a language is context free or not. Unlike regular languages, in the case of CFL pumping lemma, we break its strings into five parts and pump second and fourth substring. Pumping lemma for Context free languages is -

For every context free languages L, there exists a number n such that for every string z in L, we can write z = uvwxy, where-

1. $|vwx| \leq n$
2. $|vx| \geq 1$
3. For every $i \geq 0$, the string $uv^iwx^iy$ is in L.

For example, suppose a language $L = \{a^nb^nc^n \mid n \geq 0\}$, we need to check the language is CFL or not.

We have studied in the regular languages, whenever we are applying pumping lemmas, we try to show a contradiction which implies this language is not belongs to this class.

Lets there exists a positive integer number n, lets our string is $a^nb^nc^n$. Lets divide it into five parts such that z = uvwxy, considering pumping lemma's conditions $|vwx| \leq n$ and $|vx| \geq 1$

$u=a^n$, $v=b^{n/3}$, $w=b^{n/3}$, $x=b^{n/3}$, $y= c^n$

Now, for i=0,

$uv^iwx^iy = uv^0wx^0y = a^n(b^{n/3})^0b^{n/3}(b^{n/3})^0c^n$

$= a^n\, b^{n/3}c^n \notin L$

Space for learners:

So, it's a simple contradiction, Hence the language is not a context free language.

Space for learners:

**CHECK YOUR PROGRESS**

Question 5: What is CNF?

Question 6: What is GNF?

Question 7: State the pumping lemma of context free languages?

## 4.6 SUMMING UP

- Pushdown Automata (PDA) is a way of constructing context free grammar just like finite automata for regular grammar. Only difference is extra memory element stack here, stack allows PDA to recognize some complex languages that is why A PDA is more powerful than finite automata (FA).
- By using PDA state diagram or instantaneous description we can construct PDA for any context free language.
- A Context free Grammar G is in Chomsky Normal Form where every productions are like --- $A \rightarrow BC$, $A \rightarrow a$, where a is a terminal and A,B,C are non-terminals.
- If the productions are like - $A \rightarrow a\alpha$, where a is a terminal and $\alpha$ consists of any number of non-terminals, then it is said to be is in Greibach Normal Form.
- There are steps to convert a grammar into its CNF, GNF.
- Pumping lemma for context free languges, which is quite similar to the regular language's pumping lemma and it's an easy way to check whether a language is context free or not.

## 4.7 ANSWERS TO CHECK YOUR PROGRESSES

**1.** Just as we design DFA for a regular grammar, pushdown automata (PDA) is a way to implement a context free grammar. Pushdown Automata are new type of computational model, which is like finite automata but have an extra memory component called stack. Formally, a PDA can be defined by 7-tuple ($Q$, $\sum$, $\Gamma$, $\delta$, $q_0$ ,$Z_0$, $F$), where –

- Q is the finite number of states
- $\sum$ is the finite set of input symbols, the alphabet
- $\Gamma$ is the finite set of stack symbols, symbols which are allowed to push/pop into the stack
- $q_0$ is the initial state of PDA
- $Z_0$ is the initial stack symbol
- F is the set of final states
- $\delta$ is a transition function: $Q \times \{\Sigma \cup \in\} \times \Gamma \rightarrow Q \times \Gamma^*$ .

**2.** Transition function of PDA defines the mappings of state to state, which is denoted by $\delta$ that implies a PDA will read input symbol and stack symbol (top of the stack) and move to a new state and change the symbol of stack and mathematically written as $Q \times \{\Sigma \cup \in\} \times \Gamma \rightarrow Q \times \Gamma^*$, where Q is the finite number of states, $\sum$ is the input alphabet and $\Gamma$ is the finite set of stack symbols of PDA.

**3.**Yes, we constructed PDA using state diagrams because it's an easy way to construct a PDA. We can construct PDA by using instantaneous description and turnstile symbol as well, where we need to write each and every moves of your PDA. By seeing these moves, one can easily understand the working principle of designed PDA.

**4.**Refer section no. 4.3.3.

**5.**A Context free Grammar G is in Chomsky Normal Form where every production is either of the form:$A \rightarrow BC$, $A \rightarrow a$

Where a is a terminal and A,B,C are non terminalsand If $\varepsilon$ is in the language, then we will allow the rule $S \rightarrow \varepsilon$.

***Space for learners:***

**6.** Context free Grammar G is in Greibach Normal Form(GNF) where every production is of the form: $A \rightarrow a\alpha$

Where a is a terminal and $\alpha$ consists of any number of non terminals and If $\varepsilon$ is in the language, then we will allow the rule $S \rightarrow \varepsilon$.

**7.**Pumping lemma of Context free language is –

For every context free languages L, there exists a number n such that for every string z in L, we can write z = uvwxy, where-

1. $|vwx| \leq n$
2. $|vx| \geq 1$

For every $i \geq 0$, the string $uv^iwx^iy$ is in L.

## 4.8 POSSIBLE QUESTIONS

1. Define Pushdown Automata (PDA).
2. Why Pushdown Automata is more powerful as compared to finite automata?
3. What are the normal forms in CFG?
4. State the pumping lemma for context free languages?
5. Design a PDA for the language L = {w: w has same number of 0's and 1's}.
6. Design a PDA for accepting a language $\{0^n1^m0^n \mid m, n>=1\}$.
7. Convert the following context free grammar into its CNF:

   S->a

   S->aZ

   Z->a

8. Convert the following context free grammar into its CNF:

   $S \rightarrow aXbX$

   $X \rightarrow aY \mid bY \mid \varepsilon$

   $Y \rightarrow X \mid c$
9. Convert the following context free grammar into its GNF:

   S -> BA

Space for learners:

B -> b | SB

A -> a

10. Using Pumping lemma, check L = {ww | w∈{0,1}*} is context free or not.

## 4.9 REFERENCES AND READING SUGGESTIONS


- Introduction to Automata Theory,Languages and Computation, *John E Hopcroft ,Matwani& Jeffery D. Ullman*
- Introduction to Languages and the Theory of Computation, *John C. Martin*
- Elements of the Theory of Computation, *Lewis & Papadimitriou*
- Examples- https://www.geeksforgeeks.org/
- Javatpoint - https://www.javatpoint.com/


Space for learners: