MALLA REDDY ENGINEERING COLLEGE (Autonomous)

(An Autonomous institution approved by UGC and Affiliated to JNTUH Hyderabad,
Accredited by NAAC with 'A' Grade,Accredited by NBA,
Maisammaguda, dhulapally,(Post,Via Kompally),Secunderabad-500100 Ph:040-64634234)

Department of Computer Science and Engineering

Subject Name: Discrete Mathematics

Year and Semester: II-I

Regulations: MR-18

Syllabus

2018-19 Onwards (MR-18)	**MALLA REDDY ENGINEERING COLLEGE (Autonomous)**	**B.Tech. III Semester**		
Code: 80505	**Discrete Mathematics** (Common for CSE and IT)	**L**	**T**	**P**
Credits: 3		**3**	**-**	**-**

Prerequisites: NIL

Course Objectives:
This course provides the concepts of mathematical logic demonstrate predicate logic and Binary Relations among different variables, discuss different type of functions and concepts of Algebraic system and its properties. It also evaluates techniques of Combinatorics based on counting methods and analyzes the concepts of Generating functions to solve Recurrence equations.

MODULE I: Mathematical Logic [10 Periods]
Basic Logics-Statements and notations, Connectives, Well-formed formulas, Truth Tables, tautology.
Implications and Quantifiers-Equivalence implication, Normal forms, Quantifiers, Universal quantifiers.

MODULE II: Predicate Logic and Relations [10 Periods]

Predicate Logic- Predicative logic, Free and Bound variables, Rules of inference, Consistency, proof of contradiction, Proof of automatic Theorem.
Relation- Properties of Binary Relations, equivalence, transitive closure, compatibility and partial ordering relations, Lattices, Hasse diagram.

MODULE III: Functions and Algebraic Structures **[10 Periods]**

A: Functions-Inverse Function, Composition of functions, recursive Functions - Lattice and its Properties.

B: Algebraic structures-Algebraic systems Examples and general properties, Semi-groups and monoids, groups, sub-groups, homomorphism, Isomorphism, Lattice as POSET, Boolean algebra.

MODULE IV:Counting Techniques and Theorems **[09 Periods]**

Counting Techniques- Basis of counting, Combinations and Permutations with repetitions, Constrained repetitions

Counting Theorems- Binomial Coefficients, Binomial and Multinomial theorems, principles of Inclusion – Exclusion. Pigeon hole principle and its applications.

MODULE V: Generating functions and Recurrence Relation **[09 Periods]**

Generating Functions- Generating Functions, Function of Sequences, Calculating Coefficient of generating function.

Recurrence Relations- Recurrence relations, Solving recurrence relation by substitution and Generating functions. Method of Characteristics roots, solution of Non-homogeneous Recurrence Relations.

TEXTBOOKS:

1. J P Tremblay and R Manohar, "**Discrete Mathematics with applications to Computer Science**", Tata McGraw Hill.
2. J.L. Mott, A. Kandel, T.P.Baker "**Discrete Mathematics for Computer Scientists and Mathematicians**", PHI.

REFERENCES:

1. Kenneth H. Rosen, "**Discrete Mathematics and its Applications**", TMH, Fifth Edition.
2. Thomas Koshy, "**Discrete Mathematics with Applications**", Elsevier.
3. Grass Man and Trembley, "**Logic and Discrete Mathematics**", Pearson Education.
4. C L Liu, D P Nohapatra, "**Elements of Discrete Mathematics - A Computer Oriented Approach**", Tata McGraw Hill, Third Edition.

E –RESOURCES:

1. http://www.cse.iitd.ernet.in/~bagchi/courses/discrete-book/fullbook.pdf
2. http://www.medellin.unal.edu.co/~curmat/matdiscretas/doc/Epp.pdf
3. http://ndl.iitkgp.ac.in/document/yVCWqd6u7wgye1qwH9xY7xPG734QA9tMJN2ncqS12ZbN7pUSSlWCxSgPOZJEokyWJlxQLYsrFyeITA70W9C8Pg
4. http://nptel.ac.in/courses/106106094/

Course Outcomes:

At the end of the course, students will be able to

1. **Apply** the concepts of connectives and normal forms in real time applications.
2. **Summarize** predicate logic, relations and their operations.
3. **Describe** functions, algebraic systems, groups and Boolean algebra.
4. **Illustrate** practical applications of basic counting principles, permutations, combinations, and the pigeonhole methodology.
5. **Analyze** techniques of generating functions and recurrence relations.

Lecture Notes Module Wise

UNIT-I

Syallabus

Mathematical Logic : Statements and notations, Connectives, Well-formed formulas, Truth tables, Tautology, Equivalence implication, Normal forms, Quantifiers, Universal Quantifiers.

Mathematical Logic

Introduction :

The main aim of logic is to provide rules by which any particular argument or reasoning is validated (True or False).

Logic is concerned with all kinds of reasoning. For reasoning rules are required. These rules are called as ***rules of inference***.

These rules must be stated independent of any particular

- Argument
- Discipline &
- Language.

The rules of inference can be followed mechanically and independently to decide the validity of an argument. Natural languages are not precise enough to state these rules because they are ambiguous.

So, a formal language called as ***object language*** is necessary. In this language, the syntax is well defined, i.e., this language contains well-defined terms and well-specified uses of these terms.

To avoid ambiguity, symbols, which are clearly defined in object language, are used. Symbols are easy to write and manipulate. Since symbols are used, this logic is also called as ***symbolic logic***.

The study of object language requires the use of another language i.e., natural language English. The statements about the object language are made in English. So, the natural language English is known as ***meta language***.

Statements and notations:

The basic units of object language are ***primary, primitive or atomic statements***. A primitive statement is a declarative sentence, which cannot be further broken down or analyzed into simpler sentences.

Object language contains primary statements, which have one and only one of the two possible values called as ***truth values*** (True or False, T or F, 1 or 0). The assignment of truth values to these sentences is independent of the subject and context. Since, there are two possible truth values, this logic is sometimes called as ***two-valued logic***.

In the object language, the statements are of two types :

1> Primitive statements, denoted by upper-case letters (A, B, C, …, P, Q, …, Z).
2> Compound statements, obtained by joining primitive statements using certain symbols (connectives) and certain punctuation marks (parentheses).
These statements in object language are also known as ***propositions*** and the logic as ***propositional logic***.
Let us consider the following statements, to understand admissibility into object language.
1> India is a country.
2> Mumbai is the capital of China.
3> This statement is false.
4> 1 + 101 = 110.
5> Close the door.
6> Patna is an old city.
7> Man will reach Jupiter by 2020.
The statement (1) is true where as the statement (2) is false.
The statement (3) cannot be assigned a proper truth value.
The statement (4)'s truth value depends on the context.
The statement (5) is a command.
The statement (6) is considered true in some parts of the world and false in certain other parts.
The statement (7)'s truth value is known only after 2020.
Symbols are used to denote the atomic statements admitted in the object language.
While making a statement about the object, it is a practice to use the ***name*** of the object as shown below.
8> This table is big.
Here, the expression "this table" is used as a name of the object.
Now, let us consider the following example to understand the difference between the ***name of the object*** and ***name to a name***.
9> Ramu is a good boy
10> "Ramu" contains four letters.
In the statement (9) Ramu is the name of the object.
But in statement (10) "Ramu" is the name to a name. Thus to differentiate, the name to a name is enclosed between the quotation marks.
The above convention can be used to name the sentence also as shown below.
11> "Ramu is a good boy" contains "Ramu".
The above statement (11) is a statement about statement (9) and the word "Ramu". In the above example, the statement (9) is named by enclosing it in quotation marks.
The naming of the statements can also be done as shown below.
12> "Ramu is a good boy" is true
can be written as
12 a> (9) is true.
If a particular person or an object has more than one name, then the name of the object is given statement can be substituted by any other name without losing the meaning.
The above situation is analogous to the name-object concept exists in many programming languages.
Ex : During the function call, the actual parameters are associated with the formal parameters either by name or by value (Call-by-name or Call-by-value).
Usually, the upper-case letters A, B, …, P, Q, …, Z and subscripted upper-case letters A_1, A_2, … are used to represent statements in symbolic logic.
13> P : It is raining today.

In the above statement, “P” is a statement in symbolic logic corresponds to the statement in English “It is raining today”.

Connectives :

The compound, molecular, complex or composite statements are constructed from simple statements by using certain connecting words or expressions known as sentential connectives. There are several connectives (and, or, but etc.,) available in English, but they have variety of meanings based on the context i.e., ambiguity.

But, for the object language, the connectives should have definite meanings. Here, the connectives are represented using the symbols. These connectives define an algebra that satisfies a set of properties. These properties help us to perform some calculations by using the statements as objects.

The upper-case letters with or without subscripts also used to denote arbitrary statements. So, a statement “P” either denotes a particular statement or serves as a place holder for any other statement.

Now, a symbol denotes either a definite statement, called a ***constant*** or an arbitrary statement, called a ***variable***. The truth value of statement variable “P” can be determined only after assigning the sentence to it. Here, “P” is sometimes called as ***statement formula*** or simply ***statement***.

Let

P : It is raining today

Q : It is snowing.

Let “R” be a statement variable whose possible replacements are P and Q. As long as replacement is not specified, “R” remains as a variable without the truth value.

The common and popular connectives and their meaning in brief are tabulated.

S. No.	Name	Representation	Meaning
1	Negation	$\neg P$	“not P”
2	Conjunction	$P \land Q$	“P and Q”
3	Disjunction	P V Q	“P or Q (or both)”
4	Conditional	$P \rightarrow Q$	“if P then Q”
5	Biconditional	$P \leftrightarrow Q$	“P if and only if Q”

Negation :

The negation of a given statement can be obtained by placing the word not at a proper place or prefixing the statement with the phrase “It is not the case that”.

If “P” denotes a statement, its negation can be represented as $\neg$***P***, ***~P***, ***NOT P***, or ***P***. The truth table for logical negation is as shown below:

P	$\neg$***P***
T	F
F	T

Let us consider the statement,

P : Hyderabad is a city

Then $\neg$P represents

It is not the case that Hyderabad is a city or

Hyderabad is not a city.

A given statement in the object language may correspond to several statements in English because the given fact can be expressed in a number of ways.

Note : The negation is called a connective although it modifies a single statement. So, it is a unary operator.

Conjunction :

The conjunction of two statements P and Q is the statement ***P □ Q, P & Q, P AND Q, or P.Q***, read as "P and Q". The truth table for ***P □ Q*** is as given below :

P	***Q***	***P □ Q***
T	T	T
T	F	F
F	T	F
F	F	F

From the above table, it is observed that P □ Q will be T if both P and Q are T; otherwise it is F.

Ex 1 :

P : It is raining today.

Q : There are 20 tables in this room.

P □ Q : It is raining today and There are 20 tables in this room.

Ex 2 :

Translate the following statement into symbolic form :

Jack and Jill went up the hill

P : Jack went up the hill

Q : Jill went up the hill

P □ Q : Jack went up the hill and Jill went up the hill

Sometimes, the connective "and" is used in different sense. Then it cannot be translated by the symbol □.

Let us consider the following statements :

1> Roses are red and Violets are blue.

2> He opened the book and started to read.

3> Jack and Jill are cousins.

In statement (1) the conjunction " and" is used in the same sense as the symbol □.

In statement (2) the conjunction "and" is used in the sense "and then", so it cannot be represented using □.

In statement (3), the word "and" is not a conjunction.

Disjunction :

The disjunction of two statements P and Q is the statement ***P V Q, P || Q, P OR Q, or P + Q*** read as "P or Q". The truth table for P V Q is as given below :

P	***Q***	***P V Q***
T	T	T
T	F	T
F	T	T
F	F	F

From the above table, it is observed that P V Q will be F if both P and Q are F; otherwise it is T.

The word "or" in English is commonly used for both "inclusive or" and "exclusive or". So, the connective V is not always same as the word "or".

Now let us consider the following statements to understand the usage of "or" in English.

1> I shall watch the game on television or go to the game.

2> There is something wrong with the bulb or with the wiring.

3> Twenty or thirty animals were killed in the fire today.

In statement (1), the connective “or” is used as “exclusive or” because only one the possibilities exist, but not both.
In statement (2), the connective “or” is used as “inclusive or” because the reason could be bulb, or wire, or both.
In statement (3), the word “or” is used for indicating an approximate number of animals, and not as connective.
From the above observations it is clear that the connective V is used as “inclusive or”.
Ex :
P : It is raining today.
Q : There are 20 tables in this room.
P V Q : It is raining today or There are 20 tables in this room.
From the above example, it is observed that the statements involved in disjunction need have any relationship, because here we are concerned with the logical values and not with meaning.
Conditional :
Let P and Q be any two statements, then $P \rightarrow Q$, read as “If P, then Q” is called as a ***conditional statement***. The truth table for $P \rightarrow Q$ is as given below :

P	***Q***	**$P \rightarrow Q$**
T	T	T
T	F	F
F	T	T
F	F	T

From the above table, it is observed that the statement $P \rightarrow Q$ has the truth value F, if the first operand is T and the second operand is F; otherwise it is T.
In $P \rightarrow Q$, the statement P is called as antecedent and Q as consequent. There need not be any kind of relation between P and Q.
The statement $P \rightarrow Q$ is equivalent to ($\neg$P V Q).
Ex 1 :
Express in English the statement $P \rightarrow Q$ where
P : The sun is shining today
Q : 2 + 7 > 4.
If the sun is shining today, then 2 + 7 > 4.
A variety of conditional expressions used in English. The following forms can be appropriately translated by the symbol $\rightarrow$.
1> Q is necessary for P
2> P is sufficient for Q
3> Q if P
4> P only if Q
Ex 2 :
Convert the following sentence into symbolic form.
If either Ramu takes Mathematics or Raju takes Physics, then Ramana will take Chemistry.
The statements are denoted as
P : Ramu takes Mathematics.
Q : Raju takes Physics.
R : Ramana takes Chemistry.
Now, the given statement can be symbolized as
(P V Q) $\rightarrow$ R
Ex 3 :
Write the symbolic form for the given statement

The crop will be destroyed if there is a flood.
Can also be written as
If there is a flood, then the crop will be destroyed.
The statements are denoted as
C : The crop will be destroyed.
F : There is flood.
Now, the above statement can be symbolized as
F □ C
In English, there should be cause and effect relationship between the antecedent and consequent. But, in logic, no such kind of relationship is required because here it is concerned with truth values.

Bi-conditional :

Let P and Q be any two statements, then the statement P □ Q, P □ Q, or P Q read as “P if and only if Q”, abbreviated as “P iff Q” is called as bi-conditional statement.
The truth table for P □ Q is as given below :

P	***Q***	***P □ Q***
T	T	T
T	F	F
F	T	F
F	F	T

From the above table, it observed that the statement P □ Q has the truth value T whenever both P and Q have identical truth values; otherwise it is F.
The statement P □ Q is also translated as “P is necessary and sufficient for Q” and equivalent to (P □ Q) □ (Q □ P).

Other Connectives :

These connectives can be realized by the previous connectives. But by using these connectives the formulas become simpler. They are tabulated as :

S. No.	Name	Representation	Meaning
1	Exclusive Disjunction	P □ Q	“P Exclusive OR Q ”
2	NAND	P □ Q	“P NAND Q”
3	NOR	P □ Q	“P NOR Q”

Exclusive Disjunction or Exclusive OR :

Let P and Q be any two formulas, then the statement ***P V Q, P ^ Q, or P □ Q*** read as “P Exclusive OR Q”. It is equivalent to (P □ □Q) V (□P □ Q). The truth table is as given below :

P	***Q***	***P □ Q***
T	T	F
T	F	T
F	T	T
F	F	F

From the above table it observed that the statement P □ Q has truth value T whenever either P or Q, but not both, has the truth value T; otherwise it is false.

NAND :

Let P and Q be any two formulas, then the statement ***P □ Q, P | Q, or P NAND Q*** read as “P Nand Q”. It is equivalent to □(P □ Q) and (□P) ∨ (□Q). The truth table is as given below :

P Q P ↑ Q

T T F
T F T
F T T
F F T
From the above table it is observed that the statement P Q has the truth value F whenever both P and Q have the truth value T; otherwise it is T.

NOR :

Let P and Q be any two formulas, then the statement ***P Q, P Q, or P NOR Q*** read as “P Nor Q”. It is equivalent to (P V Q) and (P) (Q). The truth table is as given below :

P Q P Q
T T F
T F F
F T F
F F T
From the above table it is observed that the statement P Q has the truth value T whenever both P and Q have the truth value F; otherwise it is F.

Well-formed Formulas (Wffs) :

A statement formula is an expression, consisting a string of variables (upper-case letters with or without subscripts), parentheses, and connective symbols. The variables are called as the components of the statement formula.
A statement can be obtained from statement formula by replacing the variables by statements. The statement’s truth value depends upon the truth values of the statements used in replacing the variables.
A well-formed formula (wff) can be generated recursively by the following rules :
1> A statement variable standing alone is a well-formed formula.
2> If A is a well-formed formula, then A is a well-formed formula.
3> If A and B are well-formed formulas, then (A B), (A V B), (A B), (A B), (A B), (A B), and (A B) are well-formed formulas).
4> A string of symbols containing the statement variables, connectives, and parentheses is a well-formed formula iff it can be obtained by finitely many applications of the rules 1, 2, and 3.
Examples of well-formed formulas are :
(P Q) (P (P V Q)) (P (Q R)) (P (Q R))
The following are not well-formed formulas :
(P Q) (Q) Since Q is not wff.
(Q R
For the sake of simplicity and convenience, the outer parentheses are omitted.
P Q in the place of (P Q)
(P V Q) R in the place of ((P V Q) R).

Truth Tables :

The truth value of a statement formula for each possible combination of the truth values of the component statements, has to be determined. A table showing all such truth values is called the ***truth table*** of the formula.
If there are ***n*** distinct components in a statement formula, then there are ***2n*** possible combinations of truth values. So, there are ***2n*** rows in the truth table.
There are two methods for the construction of truth tables.
In the first method, all possible truth values of ***n*** components are entered in the first ***n*** columns.

Now, the components along with the connectives are placed one by one in the next columns.
In the second method, there will be separate columns for each component variable and the connective present in the statement formula. The bottom of the table contains sequence of step numbers to be followed for arriving the final step.

Ex 1 : Construct the truth table for the statement formula P V □Q.
Method 1:
P Q □Q P V □Q
T T F T
T F T T
F T F F
F F T T
Method 2:
P Q P V □ Q
T T T T F T
T F T T T F
F T F F F T
F F F T T F
Step Number 1 3 2 1
Ex 2: Construct the truth table for (P V Q) V □P.
Method 1 :
P Q P V Q □P (P V Q) V □P
T T T F T
T F T F T
F T T T T
F F F T T
Method 2 :
P Q (P V Q) V □ P
T T T T T T F T
T F T T F T F T
F T F T T T T F
F F F F F T T F
Step Number 1 2 1 3 2 1
Exercise : Construct the truth tables for the following formulas.
a> P □ □P
b> □(□P V □Q)
c> P □ (Q □ □P)
d> □P V Q
e> (P □ Q) □ (Q □ P)
f> (P □ □Q) V (□P □ Q)
g> □(P □ Q)
h> □P ∨ □Q
i> □(P V Q)
j> □P □ □Q
k> P □ (Q □ R)

Tautologies :
In the truth table, the truth value of the statement formula depends upon the truth values of the

statements substituted for the variables and it is listed in the last column.
The entries in the final column depend on the truth values of the statements assigned to the variables, rather than on the statements themselves.
So, the final column of the truth table for a given formula contains both T and F.
There are some formulas whose truth values are always T or always F irrespective of the truth values assigned to the variables.
Ex :
P V □P is always T
P □ □P is always F
A statement formula, which is ***true*** irrespective of the truth values of the statements which replace the variables in it is called a ***universally valid formula, tautology, logical truth or identically true***.
A statement formula, which is ***false*** irrespective of the truth values of the statements which replace the variables in it is called a ***contradiction, logical false or identically false***.
The straight forward method to determine whether a given formula is a tautology is to construct the truth table.
But, this process becomes tedious, when there are many distinct variables in the formulas. This is because n distinct variables require 2**n** rows in the truth table.
So, alternative methods are required to determine whether the given formula is a tautology, without the construction of its truth table.
Let A and B be two statement formulas, which are tautologies. Now, for all the truth values of A and B, A □ B will be always T. Thus A □ B is also tautology.
A formula A is called a substitution instance of another formula B, if A can be obtained from B by substituting formulas for some variables of B. Here, the same formula has to be substituted for each occurrence of the same variable.
Ex :
Let B : P □ (J □ P).
Substitute R □ S for P in B to get
A : (R □ S) □ (J □ (R □ S)).
So, A is a substitution instance of B.
But, (R □ S) □ (J □ P) is not a substitution instance of B, because P is in J □ P is not replace by R □ S.
Ex :
The substitution instances of P □ □Q are :
1> (R V □S) □ □(J V M)
2> (R □ □S) □ □(R □ □S)
3> (R □ □S) □ □P
4> Q □ □(P V □Q)

While constructing the substitution instances of a formula, substitutions are made for the atomic variable, but not for molecular.
Ex : So, P □ Q is not a substitution instance of P □ □R.
It is the fact that any substitution of a tautology is a tautology.
So, to determine whether the given formula is a tautology, find out whether the given formula is a substitution instance of a tautology.
Equivalence of Formulas :
Let A and B be two statement formulas.
Let P**1**, P**2**, …, P**n** denote all the variables appearing in both A and B.

Now determine the truth values of A and B for 2**n** possible combinations of truth values assigned to P**1**, P**2**, …, P**n**.
The statement formulas A and B are said to be ***equivalent*** if the truth value of A is equal to the truth value of B for each of the 2**n** possible combinations.
So, the final columns in the truth tables of A and B will be ***identical*** provided the variables and the assignment of truth values to the variables appear in the same order.
The equivalence of two formulas say A and B is represented as "A □ B" and read as "A is equivalent to B".
"A □ B" is a statement in meta language and not in the object language. The symbol "□" is not a connective but a symbol in the meta language.
Ex :
1> □ □P □ P
2> P V P □ P
3> (P □ □P) V Q □ Q
4> (P V □P) □ (Q V □Q)
The examples (3) and (4) illustrate that for the equivalence of two formulas, it is not necessary to assume that they both contain the same variables.
If two formulas are equivalent and a particular variable occurs in only one of them, then the truth value of this formula is independent of this variable.
In example (3), the truth value of (P □ □P) V Q is independent of the truth value of P.
In example (4), the truth values of (P V □P) and (Q V □Q) are each independent of P and Q.
Equivalence is a symmetric relation; i.e., "A is equivalent to B" is same as "B is equivalent to A".
Equivalence is a transitive relation; i.e., if A □ B and B □ C, then A □ C.
The basic way to find the equivalence of two formulas is :
- Construct their truth tables using all combinations of truth values associated with the variables appearing in both formulas.
- Compare the final columns of the truth tables.

Ex : Prove (P □ Q) □ (□P V Q)
The truth table is :
P Q P □ Q □P □P V Q (P □ Q) □ (□P V Q)
T T T F T T
T F F F F T
F T T T T T
F F T T T T
The following table shows some basic equivalence formulas
S. No. Disjunctions Conjunctions Laws
1 P V P □ P P □ P □ P Idempotent Law
2 (P V Q) V R □ P V (Q V R) (P □ Q) □ R □ P □ (Q □ R) Associative Law
3 P V Q □ Q V R P □ Q □ Q □ P Commutative Law
4 P V (Q □ R) □ (P V Q) □ (P V R) P □ (Q V R) □ (P □ Q) V (P □ R) Distributive Law
5 P V F □ P P □ T □ P
6 P V T □ T P □ F □ F
7 P V □P □ T P □ □P □ F
8 P V (P □ Q) □ P P □ (P V Q) □ P Absorption law
9 □(P V Q) □ □P □ □Q □(P □ Q) □ □P V □Q De Morgan's law
In the above table, pairs of equivalence formulas are arranged two to a line such as
A**1** □ B**1** A**2** □ B**2**.

For each pair A**1**, B**1** there is a corresponding pair A**2**, B**2** in which V is replaced by □, □ by V, T by F, and F by T.
A**1** and A**2**, B**1** and B**2** are said to be duals of each other.
In replacement process, any part of a statement formula, be it atomic or molecular, can be replaced by any other formula.
Note : Any part of a given formula that is to be replaced by another formula must be equivalent to that other formula, so that the result is equivalent to the original formula.
If any part or parts of a tautology is replaced by formulas that are equivalent to these parts, then again a tautology is obtained.
Ex : Show that P □ (Q □ R) □ P □ (□Q V R) □ (P □ Q) □ R.
P □ (Q □ R) □ P □ (□Q V R)
P □ (□Q V R) □ □P V (□Q V R)
□ (□P V □Q) V R
□ □(P □ Q) V R
□ (P □ Q) □ R

Ex : Show that (□P □ (□Q □ R)) V (Q □ R) V (P □ R) □ R.
(□P □ (□Q □ R)) V (Q □ R) V (P □ R) □ (□P □ (□Q □ R)) V ((Q V P) □ R)
□ ((□P □ □Q) □ R) V ((Q V P) □ R)
□ ((□P □ □Q) V (Q V P)) □ R
□ ((□P □ □Q) V (Q V P)) □ R
□ (□(P V Q) V (P V Q)) □ R
□ T □ R
□ R
Ex : Show that ((P V Q) □ □(□P □ (□Q V □R))) V (□P □ □Q) V (□P □ □R) is a tautology.
By applying De Morgan's laws, we get
□P □ □Q □ □(P V Q) □P □ □R □ □(P V R)
(□P □ □Q) V (□P □ □R) □ □(P V Q) V □(P V R)
□ □((P V Q) □ (P V R))
Also
□(□P □ (□Q V □R)) □ □(□P □ □(Q □ R))
□ P V (Q □ R)
□ (P V Q) □ (P V R)
(P V Q) □ ((P V Q) □ (P V R)) □ (P V Q) □ (P V R)
So, the given formula is equivalent to
((P V Q) □ (P V R)) V □((P V Q) □ (P V R))
The above formula is a substitution instance of P V □P i.e., Tautology.
Exercises : Show the following equivalences.
1> P □ (Q □ P) □ □P □ (P □ Q)
2> P □ (Q V R) □ (P □ Q) V (P □ R)
3> (P □ Q) □ (R □ Q) □ (P V R) □ Q
4> □(P □ Q) □ (P V Q) □ □(P □ Q)
Tautological Implications :
Except the conditional, the connectives □, V and □ are symmetric in the sense that P □ Q □ Q □ P, P V Q □ Q V P and P □ Q □ Q □ P.
For any statement formula P □ Q, the statement formula Q □ P is called its ***converse***, □P □ □Q is called its ***inverse***, and □Q □ □P is called its ***contra-positive***.
P □ Q □ □Q □ □P Q □ P □ □P □ □Q

A statement A is said to tautologically imply a statement B if and only if A □ B is a tautology, denoted as A □ B, read as "A implies B".

Normal Forms :
Let A(P**1**, P**2**, …, P**n**) be a statement formula where P**1**, P**2**, …, P**n** are the atomic variables.
The truth table for A consists of truth values for A, for all possible assignments of the truth values to P**1**, P**2**, …, P**n**.
If the formula A have
- The truth value T for all possible combinations, then it is called as ***identically true or tautology***.
- The truth value F for all possible combinations, then it is called as ***identically false or contradiction***.
- The truth value T for at least one combination, then it is called as ***satisfiable***.
A ***decision problem*** is a problem of determining, whether the given formula is a tautology, contradiction or at least satisfiable in a finite number of steps.
The straight forward approach for determination is by constructing the truth tables. But, it is tedious and complicate when there are many component variables in the given formula.
The alternative procedure is the reduce the given formula to ***normal forms***.
There are four types of normal forms :
1> Disjunctive Normal Form (DNF).
2> Conjunctive Normal Form (CNF).
3> Principal Disjunctive Normal Form (PDNF).
4> Principal Conjunctive Normal Form (PCNF).
Disjunctive Normal Form (DNF) :
Let us use the term ***"product"*** in place of ***"conjunction"*** and ***"sum"*** in place of ***"disjunction"***.
A product of the variables and their negations in a formula is called as ***elementary product***.
Ex : P, □P □ Q, □Q □ P □ □P, □Q □ P
A sum of the variables and their negations in a formula is called as ***elementary sum***.
Ex : P, □P V Q, □Q V P V □P, □Q V P
Any part of an elementary sum or product which is itself an elementary sum or product is called as a ***factor*** of the original elementary sum or product.
Ex : In □Q V P V □P, □Q V P and P V □P are factors.
In □Q □ P □ □P, □Q □ P and P □ □P are factors.
The following statements hold for elementary sums and products.
□ A necessary and sufficient condition for an elementary product to be identically false is that it contains at least one pair of factors in which one is the negation of the other.
□ A necessary and sufficient condition for an elementary sum to be identically true is that it contains at least one pair of factors in which one is the negation of the other.
A formula which is equivalent to a given formula and consists of a sum of elementary products is called as a ***disjunctive normal form*** of the given formula.

The procedure to obtain disjunctive normal form of a given formula is as given below :
1> The connectives □, □, □, □, and □, have to be replaced with equivalent formulas containing □, V and □.
2> Now, apply De Morgan's laws wherever applicable.
3> Next, apply distributive laws wherever applicable to obtain the required normal form.
Ex : Obtain the disjunctive normal form for P □ (P □ Q).

P □ (P □ Q) □ P □ (□P V Q)
□ (P □ □P) V (P □ Q)
Ex : Obtain the disjunctive normal form for R □ S.
R □ S □ (R □ S) □ (S □ R)
□ (□R V S) □ (□S V R)
□ ((□R V S) □ □S) V ((□R V S) □ R)
□ ((□R □ □S) V (S □ □S)) V ((□R □ R) V (S □ R))
□ ((□R □ □S) V F) V ((F V (S □ R))
□ (R □ S) V (□R □ □S)
Ex : Obtain the disjunctive normal form for □(P V Q) □ (P □ Q).
Apply the above equivalence taking R as □(P V Q) and S as (P □ Q)
□(P V Q) □ (P □ Q) □ (□(P V Q) □ (P □ Q)) V (□□(P V Q) □ □(P □ Q))
□ (□P □ □Q □ P □ Q) V ((P V Q) □ (□P V □Q))
□ (□P □ □Q □ P □ Q) V (P □ □P) V (P □ □Q) V (Q □ □P) V (Q □ □Q)
The disjunctive normal form of a given formula is not unique. Different disjunctive normal forms can be obtained for a given formula by applying distributive laws in different ways.
Ex : P V (Q □ R) □ (P V Q) □ (P V R)
□ (P □ P) V (P □ R) V (Q □ P) V (Q □ R)
Note: A given formula is identically false if every elementary product appearing in its disjunctive normal form is identically false. Every elementary product is of either (P □ □P) or (P □ F) form.

Conjunctive Normal Form (CNF) :

Let us use the term ***"product"*** in place of ***"conjunction"*** and ***"sum"*** in place of ***"disjunction"***.
A product of the variables and their negations in a formula is called as ***elementary product***.
Ex : P, □P □ Q, □Q □ P □ □P, □Q □ P
A sum of the variables and their negations in a formula is called as ***elementary sum***.
Ex : P, □P V Q, □Q V P V □P, □Q V P
Any part of an elementary sum or product which is itself an elementary sum or product is called as a ***factor*** of the original elementary sum or product.
Ex : In □Q V P V □P, □Q V P and P V □P are factors.
In □Q □ P □ □P, □Q □ P and P □ □P are factors.

The following statements hold for elementary sums and products.
□ A necessary and sufficient condition for an elementary product to be identically false is that it contains at least one pair of factors in which one is the negation of the other.
□ A necessary and sufficient condition for an elementary sum to be identically true is that it contains at least one pair of factors in which one is the negation of the other.
A ***conjunctive normal form*** of a given formula is equivalent to it and consists of a product of elementary sums.
The procedure to obtain conjunctive normal form of a given formula is as given below :
1> The connectives □, □, □, □, and □, have to be replaced with equivalent formulas containing □, V and □.
2> Now, apply De Morgan's laws wherever applicable.
3> Next, apply distributive laws wherever applicable to obtain the required normal form.
Ex : Obtain the conjunctive normal form for P □ (P □ Q).
P □ (P □ Q) □ P □ (□P V Q)
Ex : Obtain the conjunctive normal form for □(P V Q) □ (P □ Q).
□(P V Q) □ (P □ Q) □ (□(P V Q) □ (P □ Q)) □ ((P □ Q) □ □(P V Q))

□ ((P V Q) V (P □ Q)) □ (□(P □ Q) V □(P V Q))
□ ((P V Q V P) □ (P V Q V Q)) □ ((□P V □Q) V (□P □ □Q))
□ (P V Q V P) □ (P V Q V Q) □ (□P V □Q V □P) □ (□P V □Q V □Q)
The conjunctive normal form of a given formula is not unique. Different conjunctive normal forms can be obtained for a given formula by applying distributive laws in different ways.
Note : A given formula is identically true if every elementary sum appearing in its conjunctive normal form is identically true. Every elementary sum is of either (P V □P) or (P V T) form.
Ex : Show that the formula Q V (P □ □Q) V (□P □ □Q) is a tautology.
Let us obtain the conjunctive normal form for the given formula
Q V (P □ □Q) V (□P □ □Q) □ Q V ((P V □P) □ □Q)
□ (Q V (P V □P)) □ (Q V □Q)
□ (Q V P V □P) □ (Q V □Q)
Since, each of the elementary sum is a tautology, the given formula is also a tautology.
Principal Disjunctive Normal Form (PDNF) :
For a given formula, there are different disjunctive normal forms. Even though they all are equivalent, it is necessary to obtain a unique normal form for a given formula.
Let P and Q be two statement variables.
Let us construct all possible formulas consisting of conjunctions of P or its negation and conjunctions of Q or its negation.
None of the formulas should contain both a variable and its negation.
Any formula obtained by commuting the formulas in the conjunction is not included in the list.
For example, both P □ Q and Q □ P cannot be included.

For two variables P and Q, there are 2**2** such formulas in an order as
P □ Q P □ □Q □P □ Q □P □ □Q
These formulas are called **minterms or Boolean conjunctions** of P and Q.
Now let us construct the truth table consisting of all minterms.

P	***Q***	***P □ Q***	***P □ □Q***	***□P □ Q***	***□P □ □Q***
T	T	T	F	F	F
T	F	F	T	F	F
F	T	F	F	T	F
F	F	F	F	F	T

From the above table, it is clear that no two minterms are equivalent and each minterm has the truth value T for exactly one combination.
For every truth value T in the truth table of the given formula, select the minterm which also has the value T for the same combination of the truth values of P and Q.
The disjunctions of these minterms will be equivalent to the given formula.
Ex :
P □ Q □ (P □ Q) V (□P □ Q) V (□P □ □Q)
P V Q □ (P □ Q) V (P □ □Q) V (□P □ Q)
□(P □ Q) □ (P □ □Q) V (□P □ Q) V (□P □ □Q)
The minterms are written down by including the variable if its truth value is T and its negation if the value is F.
From the above examples, it is observed that the number of minterms appearing in the normal form is equal to the number of entries with T in the truth table of the given formula.
Principal disjunctive normal form is equivalent to the given formula and consists of disjunctions of minterms only. This form is also called as ***sum-of-products canonical form***.

It is possible to define the minterms for three or more variables. The minterms for the three variables P, Q, and R are :
P □ Q □ R P □ Q □ □R P □ □Q □ R P □ □Q □ □R
□P □ Q □ R □P □ Q □ □R □P □ □Q □ R □P □ □Q □ □R
Obtaining the principal disjunctive normal form with the help of truth table is tedious because of 2**n** rows and 2**n** minterms; where n is the number of components in the given formula.
So, an alternative procedure is required and it is :
1> The connectives □, □, □, □, and □, have to be replaced with equivalent formulas containing □, V and □.
2> Now, apply De Morgan's laws wherever applicable.
3> Next, apply distributive laws wherever applicable to obtain the required normal form.
4> Any elementary product which is contradiction is dropped.
5> Minterms are obtained in the disjunctions by introducing the missing factors.
6> Identical minterms appearing in the disjunctions are deleted.

Ex : Obtain the principal disjunctive normal form of □P V Q.
□P V Q □ (□P □ (Q V □Q)) V (Q □ (P V □P)) (A □ T □ A)
□ (□P □ Q) V (□P □ □Q) V (Q □ P) V (Q □ □P) (Distributive Law)
□ (□P □ Q) V (□P □ □Q) V (P □ Q)
Ex : Obtain the principal disjunctive normal form of (P □ Q) V (□P □ R) V (Q □ R).
(P □ Q) V (□P □ R) V (Q □ R) □ (P □ Q □ (R V □R)) V (□P □ R □ (Q V □Q)) V (Q □ R □ (P V □P))
□ (P □ Q □ R) V (P □ Q □ □R) V (□P □ Q □ R) V (□P □ □Q □ R)
Ex : Show that the following are equivalent formulas.
a> P V (P □ Q) □ P
b> P V (□P □ Q) □ P V Q
Obtain the principal disjunctive normal form of each formula and compare these normal forms.
a> P V (P □ Q) □ (P □ (Q V □Q)) V (P □ Q)
□ (P □ Q) V (P □ □Q)
P □ (P □ (Q V □Q))
□ (P □ Q) V (P □ □Q)
b> P V (□P □ Q) □ (P □ (Q V □Q)) V (□P □ Q)
□ (P □ Q) V (P □ □Q) V (□P □ Q)
P V Q □ (P □ (Q V □Q)) V (Q □ (P V □P))
□ (P □ Q) V (P □ □Q) V (□P □ Q)
Ex : Obtain the principal disjunctive normal form of
P □ ((P □ Q) □ □(□Q V □P))
Principal Conjunctive Normal Form (PCNF) :
For a given formula, there are different conjunctive normal forms. Even though they all are equivalent, it is necessary to obtain a unique normal form for a given formula.
Let P and Q be two statement variables.
Let us construct all possible formulas consisting of disjunctions of P or its negation and disjunctions of Q or its negation.
None of the formulas should contain both a variable and its negation.
Any formula obtained by commuting the formulas in the disjunction is not included in the list.
For example, both P V Q and Q V P cannot be included.
For two variables P and Q, there are 2**2** such formulas in an order as
P V Q P V □Q □P V Q □P V □Q
These formulas are called **maxterms or Boolean disjunctions** of P and Q.

Thus the maxterms are duals of minterms.

Now let us construct the truth table consisting of all maxterms.

P Q P V Q P V □Q □P V Q □P V □Q
T T T T T F
T F T T F T
F T T F T T
F F F T T T

From the above table, it is clear that no two maxterms are equivalent and each maxterm has the truth value F for exactly one combination.
For every truth value F in the truth table of the given formula, select the minterm which also has the value F for the same combination of the truth values of P and Q.
The conjunctions of these maxterms will be equivalent to the given formula.

Ex :
P □ Q □ (□P V Q)
P □ Q □ (P V □Q) □ (□P V Q) □ (□P V □Q)
□(P V Q) □ (P V □Q) □ (□P V Q) □ (□P V □Q)

The maxterms are written down by including the variable if its truth value is F and its negation if the value is T.
From the above examples, it is observed that the number of maxterms appearing in the normal form is equal to the number of entries with F in the truth table of the given formula.
Principal conjunctive normal form is equivalent to the given formula and consists of conjunctions of maxterms only. This form is also called as ***product-of-sums canonical form***.
It is possible to define the maxterms for three or more variables. The maxterms for the three variables P, Q, and R are :
P V Q V R P V Q V □R P V □Q V R P V □Q V □R
□P V Q V R □P V Q V □R □P V □Q V R □P V □Q V □R
Obtaining the principal conjunctive normal form with the help of truth table is tedious because of 2**n** rows and 2**n** maxterms; where n is the number of components in the given formula.
So, an alternative procedure is required and it is :
1> The connectives □, □, □, □, and □, have to be replaced with equivalent formulas containing □, V and □.
2> Now, apply De Morgan's laws wherever applicable.
3> Next, apply distributive laws wherever applicable to obtain the required normal form.
4> Any elementary sum which is tautology is dropped.
5> Maxterms are obtained in the conjunctions by introducing the missing factors.
6> Identical maxterms appearing in the conjunctions are deleted.
Ex : Obtain the principal conjunctive normal form of the formula S given by
(□P □ R) □ (Q □ P)

UNIT-II

Syllabus

Predicates : Predicate logic, Free & Bound variables, Rules of inference, Consistency, proof of contradiction, Automatic Theorem Proving.

Predicate Calculus

Introduction:
In symbolic logic,
- Limited to statements, statement variables, and statement formulas.
- Atomic statements are the basic units and analysis of them was not permitted.
- Only compound statements were analyzed by studying the forms, i.e., the connections between the constituent atomic statements.

In symbolic logic, it is not possible to express that any two atomic statements have some features in common. For this purpose, the **concept of a predicate** in an atomic statement is introduced.

The logic based on the analysis of predicates in any statement is called **predicate logic**. **Predicates :**

Let us consider the following two statements:

1> Ramu is a bachelor. 2>
Raju is a bachelor.

Using symbolic logic, the above statements are represented using two symbols as : P : Ramu is a
bachelor.
Q : Raju is a bachelor.

But, these symbols do not reveal the common feature, viz., both are statements about two different individuals who are bachelors.

So, some symbol is to be introduced to denote "is a bachelor" and a method to join it with the symbols denoting the names of the individuals.
Now, any individual's being a bachelor can be represented. The part "is a bachelor" is called predicate.

A predicate is symbolized by a upper-case letter and the names of the individuals or objects by lower-case letters. Every predicate describes something about one or more objects.

Let us denote the predicate "is a bachelor" symbolically by the predicate letter B, "Ramu" by m, and "Raju" by j. Now, the statements (1) and (2) can be written as B(m) and B(j) respectively.

In general, a statement of type "s is P", where P is a predicate and s is the subject can be denoted by P(s).

Let us consider the following statements All
human beings are mortal.
Ramana is a human being.
From the above two statements, we can conclude that Ramana is a
mortal.
This type of conclusion is true, but not possible with symbolic logic.

But, this can be achieved by separating the part "are mortal" from the part "All human beings", then it is possible to consider any particular human being.

For a predicate letter, there must be at least one name of an object associated with the predicate.
The number of names associated with the predicate letter is represented as a superscript to the predicate letter.
A predicate requiring m (m > 0) names is called an m-place predicate. For
example, B in (1) and (2) is a 1-place predicate.
"L : is greater than" is a 2-place predicate.

Let R denote the predicate "is red" and let p denote "This painting". Then the
statement
3> This painting is red can
be symbolized by R(p).

The connectives □, □, V, □, □, □, □, and □, described earlier can now be used to form compound statements.

For example, "Ramu is a bachelor and this painting is red" can be written as B(m) □ R(p). Now, let us

consider the statements involving the names of two objects.
4> Ramu is taller than Raju
5> Jammu is to the north of India.

Here, the predicates "is taller than" and "is to the north of" are 2-place predicates, because names of two objects are required.

Let the letter T denotes "is taller than", r_1 denotes Ramu and r_2 denotes Raju, then the statement (4) is represented as $T(r_1, r_2)$.

Let the letter N denotes "is to the north of", j denotes Jammu and i denotes India, then the statement (5) is represented as N(j, i).

The examples of 3-place and 4-place predicates are : 6> Raju
sits between Ramu and Ramana.
7> Ramu and Raju played tennis against Ramana and Suri.

An n-place predicate requires ***n*** names of objects to be inserted in fixed positions in order to obtain a statement.
Let S be an n-place predicate letter and $a_1, a_2, \ldots, a_n$ are the names of objects, then $S(a_1, a_2, \ldots, a_n)$ is
a statement.

The Statement Function, Variables, and Quantifiers :

Let M be the predicate “is a mortal”, r the name “Ramu”, d the name “Delhi” and s the name “A shirt”. Then H(r), H(d), and H(s) all denote statements and they have a common form.

Let H(x) denotes “x is a mortal”, then H(r), H(d), H(s) and others having the same form can be obtained from H(x), by replacing x by an appropriate name.

Note : H(x) is not a statement, but it results in a statement when x is replaced by the name of an object. So, the letter x is used here as a placeholder.

So, the lower-case letters are used to represent the names of the objects as well as **object variables**.

A **simple statement function** of one variable is defined to be an expression consisting of a predicate symbol and an individual variable.

A function becomes a statement when a variable is replaced by the name of any object.

This statement is called a **substitution instance** of the statement function and is a formula of statement calculus.

The compound statement functions are formed by combining one or more simple statement functions using logical connectives.

Ex : Let M(x) denote “x is a man” and H(x) denote “x is a mortal”. The compound functions are M(x) □ H(x), M(x) □ H(x), M(x) V □H(x).

The idea of having a single variable in statement formula can be extended for two or more variables.

Ex : Let G denote the predicate “is taller than”, then 1> G(x, y) : x is taller than y

Let both x and y are replaced with r_1 for “Ramu” and r_2 for “Raju”, we obtain the statements as $G(r_1, r_2)$: Ramu is taller than Raju.

$G(r_2, r_1)$: Raju is taller than Ramu.

It is possible to form statement functions of two variables by using statement functions of one variable as

M(x) : x is a man

H(y) : y is a mortal

Now

M(x) □ H(y) : x is a man and y is a mortal.

But it is not possible to write every statement function of two variables using statement functions of one variable.

One way of obtaining the statements from statement functions is by replacing the variables by the names of objects.

But, there is an alternative method to obtain the statements from statement functions. To understand this, let us consider some familiar equations in elementary algebra.

2> $X + 2 = 5$

3> $X^2 + 1 = 0$

4> $(X - 1) * (X - ½) = 0$

5> $X^2 - 1 = (X - 1) * (X + 1)$

In the above equations, the universe of the variable X is the set of real numbers, complex numbers or integers.

In statement (2), if X is replaced by a real number or an integer then a statement is obtained. The resulting statement is true when 3 or 3.0 is substituted for X, and for other values it is false.

In statement (3), substitution of no real number or integer can make it true. But two substitution instances with complex numbers give true statements.

In statement (4), the universe of X is assumed to be real numbers. Here, either X = 1.0 and/or 0.5 produces a true statement when substituted.

In the case of statement (5), any number substituted for X results as true statement. Therefore, we may say that

6> For any number X, $X^2 - 1 = (X - 1) * (X + 1)$

Here, (6) is a statement and not a statement function even though the variable x appears in it. This is because of the addition of the phrase "For any number X".
In (6), the variable X need not be replaced by any name to obtain a statement.

Occasionally, when a statement involves an equality, a distinction is made by using the symbol □ instead of the equality sign to show that it is a statement. Here, (6) would be written as $X^2 - 1$ □ $(X - 1) * (X + 1)$

Let us consider the following statements, in which each one is a statement about the elements of a certain set.

7> All men are mortal. 8>
Every apple is red.
9> Any integer is either positive or negative.

The above statements can be rewritten as

7a> For all x, if x is a man, then x is a mortal 8a>
For all x, if x is an apple, then x is red
9a> For all x, if x is an integer, then x is either positive or negative.

The phrase "For all x" is denoted by the symbol "(□x)" or by (x). This symbol is placed before the statement function, which contains the phrase "for all".

M(x) : x is a man	H(x) : x is a mortal
A(x) : x is an apple	R(x) : x is red
N(x) : x is an integer	P(x) : x is either positive or negative

Now, (7a), (8a), and (9a) can be written as

7b> (□x) (M(x) □ H(x)) or (x) (M(x) □ H(x))
8b> (□x) (A(x) □ R(x)) or (x) (A(x) □ R(x))
9b> (□x) (N(x) □ P(x)) or (x) (N(x) □ P(x))

The symbols (x) or (□x) are called as **universal quantifiers**. The quantification symbol is "()" or "(□)" and contains the variable, to be quantified.
Universal quantifier is used to translate expressions such as "for all", "every", and "for any". So, any statement function of one variable can be quantified to obtain a statement.

Thus (x) M(x) is a statement that can be translated as : 10>

For all x, x is a man

10a> For every x, x is a man

10b> Everything is a man

Obtaining the truth value of a statement involving universal quantifier by using the truth values of the statement function is not possible because of the following two reasons :

1> Statement functions do not have truth values. The statements are obtained by substituting the variables with the names of the objects and they have a truth value.

2> Using substitutions, in most of the cases, infinite number of statements are produced.

Note : The particular variable appearing in the statements involving universal quantifier is not important because the statements remain unchanged if x is replaced by y.

Thus (x) (M(x) □ H(x)) and (y) (M(y) □ H(y)) are equivalent.

In some cases, it may be necessary to use more than one universal quantifier in a statement.

Ex : "For any x and any y, if x is taller than y, then it is not true that y is taller than x" Let T(x, y) : x is taller than y.

Now the above statement can be symbolized as (x) (y)

(T(x, y) □ □T(y, x))

There is another quantifier, which is used to symbolize the expressions such as "for some", "there is at least one", or "there exists some".

Let us consider the following statements.

11> There exists a man 12>

Some men are clever

13> Some real numbers are rational

The statement (11) can also be expressed as 11a> There exists an x such that x is a man

11b> There is at least one x such that x is a man

The statement (12) can also be expressed as

12a> There exists an x such that x is a man and x is clever

12b> There exists at least one x such that x is a man and x is clever.

Similarly, the statement (13) can also be expressed as

13a> There exists an x such that x is a real number and x is rational

13b> There exists at least one x such that x is a real number and x is rational

The symbol "(□x)", called the existential quantifier, used to symbolize expressions such as "there is at least one x such that" or "there exist an x such that" or "for some x".

Now, the above statements can be symbolized as: M(x) :

x is a man

C(x) : x is clever

R_1(x) : x is a real number

$R_2(x)$: x is rational

11c> (□x) (M(x))
12c> (□x) (M(x) □ C(x))
13c> (□x) (R_1(x) □ R_2(x))

Predicate Formulas :
In propositional logic, the upper-case letters are used to denote definite statements and also as placeholders for the statements i.e., variables.

In Predicate logic, the upper-case letters are used to denote predicates. A **superscript n** is used to indicate it as an **n-place predicate**.

Usually, the above notation is not followed, because an n-place predicate is followed by n object or individual variables denoted by lower-case letters.
Ex : $P(x_1, x_2, \ldots, x_n)$

In predicate calculus, $P(x_1, x_2, \ldots, x_n)$ is an **atomic formula**. It also includes atomic formulas with **n = 0** as special cases. The examples of atomic formulas are :
R Q(x) P(x, y) A(x, y, z) B(y, b, z) K(p, q)

A well-formed formula (wff) of predicate calculus is obtained by using the following rules : 1> An atomic formula is a well-formed formula.
2> If A is a well-formed formula, then □A is a well-formed formula.
3> If A and B are well-formed formulas, then (A □ B), (A V B), (A □ B), (A □ B), (A □ B), (A □ B), and (A □ B) are also well-formed formulas.
4> If A is a well-formed formula and x is any variable, then (x)A and (□x)A are well-formed formulas.
5>Only those formulas obtained by using rules (1) to (4) are well-formed formulas.

Note : Since only well-formed formulas are used, the term "formula" is used for "well-formed formula".

Free and Bound Variables :
If a formula containing a part of the form (x)P(x) or (□x)P(x), then this part is called an x-bound part of the formula.
Any occurrence of x in an x-bound part of a formula is called a **bound occurrence** of x.
Any occurrence of x or of any variable that is not a bound occurrence is called a **free occurrence**.

The formula P(x) either in (x)P(x) or (□x)P(x) is called as the **scope of the quantifier**. So, scope of the quantifier is the formula immediately following the quantifier.
If the scope is an atomic formula, parentheses are not needed; otherwise parentheses are needed.

Let us consider the following examples :
(1) (x)P(x, y)
In (1), P(x, y) is the scope of the quantifier, both occurrences of x are bound occurrences and the occurrence of y is free occurrence
(2) (x)(P(x) □ Q(x))
In (2), P(x) □ Q(x) is the scope of the quantifier and all occurrences of x arebound.

(3) (x)(P(x) □ (□y)R(x, y))

In (3), the scope of (x) is P(x) □ (□y)R(x, y) and the scope of (□y)is R(x, y). All occurrences of both x and y are bound.

(4) (x)(P(x) □ R(x)) V (x)(P(x) □ Q(x))

In (4), the scope of the first quantifier is P(x) □ R(x) and the scope of second quantifier is P(x) □ Q(x). All occurrences of x arebound.

(5) (□x)(P(x) □ Q(x))

In (5), the scope of (□x) is P(x) □ Q(x) and all occurrences of x are bound. (6)

(□x)P(x) □ Q(x)

In (6), the scope of (□x) is P(x) and both occurrences of x are bound. The occurrence of x in Q(x) is free.

In the bound occurrence of a variable, the letter used to represent the variable is not important. Any other letter can be used to represent the variable without affecting the formula, provided that letter is not used elsewhere in the formula.

Ex : (x)P(x, y) and (z)P(z, y) are same.

The bound occurrence of a variable cannot be substituted by a constant; the free occurrence of a variable can be substituted.

Ex : (x)P(x) □ Q(a) is a substitution instance of (x)P(x) □ Q(y) (x)P(z) □ Q(a) is not a substitution instance of (x)P(x) □ Q(y)

Ex 1 : Let

P(x) : x is a person.
F(x, y) : x is the father of y. M(x,
y) : x is the mother of y.

Write the predicate “x is the father of the mother of y”.

Solution :

Let z be a person and mother of y. The above predicate is symbolized as : (□x) (P(z)
□ F(x, z) □ M(z, y))

Ex 2 : Symbolize the expression “All the world loves a lover”.

Solution :

The above expression means “Everybody loves a lover”.

Let

P(x) : x is a person
L(x) : x is a lover R(x,
y) : x loves y.

The above expression is symbolized as :

(x)(P(x) □ (y)(P(y) □ L(y) □ R(x, y)))

The Theory of Inference :

Propositional Logic :

The main function of logic is to provide rules of inference or principles of reasoning. The theory based on these is known as inference theory.

In this conclusion is inferred from certain premises.

If a conclusion is derived from a set of premises by using accepted rules of reasoning, then this process of derivation is called **a deduction or a formal proof**.

In mathematics, the proofs are informal, i.e., many steps in the derivation are either omitted or considered to be understood.

But in a formal proof, every rule of inference used at any stage in the derivation is acknowledged.

The rules of inference are criteria for determining the validity of an argument.

These rules are stated in terms of the forms of the statements (premises and conclusions) involved rather than in terms of the actual statements or their truth values.

So, these rules are given in terms of statement formulas rather than any specific statements.

For any argument, a conclusion is admitted to be true provided,

- The premises (assumptions, axioms, hypotheses) are accepted as true
- The reasoning used to derive a conclusion from the premises follow accepted rules of logical inference.

This type of argument is called as sound argument.

Any conclusion arrived by following rules of inference is called a valid conclusion, and the argument is called a valid argument.

Validity Using Truth Tables :

Let A and B be two statement formulas.

Now, "**B logically follow from A**" or "**B is a valid conclusion (consequence) from premise A**" iff A □ B is a tautology, i.e., A □ B.

So, from a set of premises $\{H_1, H_2, \ldots, H_m\}$ a conclusion C follows iff H_1 □ H_2 □ … □ H_m □ C (1)

In the straight forward method, truth tables are used to determine whether a conclusion logically follows from the given premises.

Let $P_1, P_2, \ldots, P_n$ be all atomic variables appearing in the premises $H_1, H_2, \ldots, H_m$ and the conclusion C.

For all possible combinations of truth values assigned to $P_1, P_2, \ldots, P_n$, the truth values of $H_1, H_2, \ldots, H_m$ and C are tabulated.

Now, check for all rows in which all $H_1, H_2, \ldots, H_m$ have the value T. If for every such row, C also has the value T, then (1) holds.

Alternatively, for every row in which C has the value F, at least one of the values of $H_1, H_2, \ldots, H_m$ is F, then (1) holds.

Ex : Determine whether the conclusion C follows logically from the premises H_1 and H_2.

a> H_1 : P □ Q	H_2 : P	C : Q
b> H_1 : P □ Q	H_2 : □P	C : Q
c> H_1 : P □ Q	H_2 : □(P □ Q)	C : □P
d> H_1 : □P	H_2 : P □ Q	C : □(P □ Q)
e> H_1 : P □ Q	H_2 : Q	C : P

The solution for (a) is :

P	**Q**	**P □ Q**
T	T	T
T	F	F
F	T	T
F	F	T

The truth table method becomes tedious when the number of atomic variables present in all the formulas representing the premises and conclusion is large.
The alternative method is using inference theory.

Rules of Inference :
To find out whether the given formula is a valid consequence of a given set of premises, the knowledge of the following is required.

- Rules of inference
- Implications
- Equivalences

There are two rules of inference, called as rules **P** and **T**.
Rule P : A premise may be introduced at any point in the derivation.
Rule T : A formula S may be introduced in a derivation if S is tautologically implied by and/or equivalent to any one or more of the preceding formulas in the derivation.

The list of implications are :

Notation	**Implication**	**Rule**
I_1	P □ Q □ P	Simplification
I_2	P □ Q □ Q	Simplification
I_3	P □ P V Q	Addition
I_4	Q □ P V Q	Addition
I_5	□P □ P □ Q	
I_6	Q □ P □ Q	
I_7	□(P □ Q) □ P	
I_8	□(P □ Q) □ □Q	
I_9	P, Q □ P □ Q	
I**10**	□P, P V Q □ Q	Disjunctive Syllogism
I**11**	P, P □ Q □ Q	Modus Ponens
I**12**	□Q, P □ Q □ □P	Modus Ponens
I**13**	P □ Q, Q □ R □ P □ R	Hypothetical Syllogism
I**14**	P V Q, P □ R, Q □ R □ R	Dilemma

The list of equivalences are :

Notation	Equivalence	Rule
E_1	□□P □ P	Double Negation
E_2	P □ Q □ Q □ P	Commutative Law
E_3	P V Q □ Q V P	Commutative Law
E_4	(P □ Q) □ R □ P □ (Q □ R)	Associative Law
E_5	(P V Q) V R □ P V (Q V R)	Associative Law
E_6	P □ (Q V R) □ (P □ Q) V (P □ R)	Distributive Law
E_7	P V (Q □ R) □ (P V Q) □ (P VR)	Distributive Law
E_8	□(P □ Q) □ □P V □Q	De Morgan's Law
E_9	□(P V Q) □ □P □ □Q	De Morgan's Law
E_{10}	P V P □ P	Idempotent Law
E_{11}	P □ P □ P	Idempotent Law
E_{12}	R V (P □ □P) □ R	
E_{13}	R □ (P V □P) □ R	
E_{14}	R V (P V □P) □ T	
E_{15}	R □ (P □ □P) □ F	
E_{16}	P □ Q □ □P V Q	
E_{17}	□(P □ Q) □ P □ □Q	
E_{18}	P □ Q □ □Q □ □P	
E_{19}	P □ (Q □ R) □ (P □ Q) □ R	
E_{20}	□(P □ Q) □ P □ □Q	
E_{21}	P □ Q □ (P □ Q) □ (Q □ P)	
E_{22}	P □ Q □ (P □ Q) V (□P □ □Q)	

Ex 1: Demonstrate that R is a valid inference from the premises P □ Q, Q □ R, and P.

Solution :

{1}	(1)	P □ Q	Rule P
{2}	(2)	P	Rule P
{1, 2}	(3)	Q	Rule T, (1), (2), and I_{11}.
{4}	(4)	Q □ R	Rule P
{1, 2, 4}	(5)	R	Rule T, (3), (4) and I_{11}.

In the above solution,

- The first column contains the set of numbers for each line, which show the premises on which the formula in the line depends.

- The second column of numbers designates the formula as well as the line of derivation.

- In the last column, P or T represents the rule of inference, followed by a comment showing from which formulas, implications and equivalences that particular formula is obtained.

Ex 2 : Show that R V S logically follows from the premises C V D, (C V D) □ □H, □H □ (A □ □B), and (A □ □B) □ (R V S).

Solution :

{1}	(1)	(C V D) □ □H	P
{2}	(2)	□H □ (A □ □B)	P
{1, 2}	(3)	(C V D) □ (A □ □B)	T, (1), (2), and I_{13}
{4}	(4)	(A □ □B) □ (R V S)	P
{1, 2, 4}	(5)	(C V D) □ (R V S)	T, (3), (4), and I_{13}
{6}	(6)	C V D	P
{1, 2, 4, 6}	(7)	R V S	T, (5), (6), and I_{11}

Ex 3 : Show that S V R is tautologically implied by (P V Q) □ (P □ R) □ (Q □ S).

Solution :

{1}	(1)	P V Q	P
{1}	(2)	□P □ Q	T, (1), E_1, and E_{16}
{3}	(3)	Q □ S	P
{1, 3}	(4)	□P □ S	T, (2), (3), and I_{13}
{1, 3}	(5)	□S □ P	T, (4), E_{18}, and E_1
{6}	(6)	P □ R	P
{1, 3, 6}	(7)	□S □ R	T, (5), (6), and I_{13}
{1, 3, 6}	(8)	S V R	T, (7), E_{16}, and E_1

Ex 4 : Show that R □ (P V Q) is a valid conclusion from the premises P V Q, Q □ R, P □ M, and □M.

Solution :

{1}	(1)	P □ M	P
{2}	(2)	□M	P
{1, 2}	(3)	□P	T, (1), (2), and I_{12}
{4}	(4)	P V Q	P
{1, 2, 4}	(5)	Q	T, (3), (4), and I_{10}
{6}	(6)	Q □ R	P
{1, 2, 4, 6}	(7)	R	T, (5), (6), and I_{11}
{1, 2, 4, 6}	(8)	R □ (P V Q)	T, (4), (7) and I_9

Ex 5 : Show I_{12} : □Q, P □ Q □ □P

Solution :

{1}	(1)	P □ Q	P
{1}	(2)	□Q □ □P	T, (1), and E_{18}
{3}	(3)	□Q	P
{1, 3}	(4)	□P	T, (2), (3), and I_{11}

Now, let us introduce a third inference rule, known as **rule CP** or **rule of conditional proof**.

Rule CP : If S can be derived from R and a set of premises, then R □ S can be derived from the same set of premises alone.

Actually, rule CP follows from the equivalence E_{19}, which states that (P □ R)
□ S □ P □ (R □ S)
In the above equivalence,
Let P denote the conjunction of the set of premises and let R be any formula.
Here, if R is included as an additional premise and S is derived from P □ R, then R □ S can be derived from the premises P alone.

Rule CP is also called as the deduction theorem and generally used if the conclusion is of the form R □ S.
Here, R is taken as an additional premise and S is derived from the given premises and R. **Ex 6:** Show

that R □ S can be derived from the premises P □ (Q □ S), □R V P, and Q. **Solution :** To derive R □ S,

first R is included as an additional premise and S is derived.

{1}	(1)	$\neg$R V P	P
{2}	(2)	R	P (assumed premise)
{1, 2}	(3)	P	T, (1), (2), and I_{10}
{4}	(4)	$P \rightarrow (Q \rightarrow S)$	P
{1, 2, 4}	(5)	$Q \rightarrow S$	T, (3), (4), and I_{11}
{6}	(6)	Q	P
{1, 2, 4, 6}	(7)	S	T, (5), (6), and I_{11}
{1, 4, 6}	(8)	$R \rightarrow S$	CP

From the above examples, it is observed that a derivation consists of a sequence of formulas, and each formula in the sequence is either a premise or tautologically implied by formulas appearing before.

If one can determine in a finite number of steps whether an argument is valid, then the decision problem for validity is solvable.

Ex 7: "If there was a ball game, then travelling was difficult. If they arrived on time, then travelling was not difficult. They arrived on time. Therefore, there was no ball game".
Show that these statements constitute a valid argument.

Solution:

Let us symbolize the atomic statements involved.

P : There was a ball game. Q :
Travelling was difficult. R :
They arrived on time.

The above English statements are symbolized as :
Premises : P □ Q, R □ □Q and R.
Conclusion : □P

Ex 8: If A works hard, then either B or C will enjoy themselves. If B enjoys himself, then A will not work hard. If D enjoys himself, then C will not. Therefore, if A works hard, D will not enjoy himself.

Solution:

Let us symbolize the atomic statements involved.

A : A works hard.
B : B will enjoy himself. C
: C will enjoy himself. D :
D will enjoy himself.

The above English statements are symbolized as :
Premises : A □ (B V C), B □ □A and D □ □C. Conclusion :
A □ □D.

Consistency of Premises and Indirect Method of Proof :

A set of formulas H_1, H_2, …, H_m is said to be consistent if their conjunction has the truth value T for some assignment of the truth values to the atomic variables appearing in H_1, H_2, …, H_m.

If, for every assignment of truth values to the atomic variables, at least one of the formulas H_1, H_2, …, H_m is false, then their conjunction is identically false. So, they are inconsistent.

If a set of formulas H_1, H_2, …, H_m is inconsistent then their conjunction implies a contradiction, i.e.,

H_1 □ H_2 □ … □ H_m □ R □ □R, where R is a formula.

This notation of inconsistency is used in a procedure called **proof by contradiction** or **indirect method of proof**.

In this method, to show that a conclusion C follows logically from the premises H_1, H_2, …, H_m, we assume that C is false and consider □C as an additional premise.

If the new set of premises is inconsistent, then the assumption that □C is true does not hold simultaneously with H_1 □ H_2 □ … □ H_m being true.

Therefore C is true whenever H_1 □ H_2 □ … □ H_m is true. Thus, C follows logically from the premises H_1, H_2, …, H_m.

Ex 1: Show that □(P □ Q) follows from □P □ □Q.

Solution: Let us introduce □□(P □ Q) as an additional premise and show that this additional premise leads to a contradiction.

{1}	(1)	□□(P □ Q)	P (Assumed)
{1}	(2)	P □ Q	T, (1), and E_1
{1}	(3)	P	T, (2), and I_1
{4}	(4)	□P □ □Q	P
{4}	(5)	□P	T, (4), and I_1
{1, 4}	(6)	P □ □P	T, (3), (4), and I_9

Ex 2: Show that the following premises are inconsistent.

1>If Raghu misses many classes through illness, then he fails high school. 2> If Raghu fails high school, then he is uneducated.
3> If Raghu reads a lot of books, then he is not uneducated.
4> Raghu misses many classes through illness and reads a lot of books.

Solution :

Let us symbolize the atomic statements involved.

M : Raghu misses many classes through illness F :
Raghu fails high school
R : Raghu reads a lot of books U :
Raghu is uneducated

So, the premises are M □ F, F □ U, R □ □U, and M □ R.

{1}	(1)	M □ F	P
{2}	(2)	F □ U	P
{1, 2}	(3)	M □ U	T, (1), (2), and I_{13}
{4}	(4)	R □ □U	P
{4}	(5)	U □ □R	T, (4), and E_{18}
{1, 2, 4}	(6)	M □ □R	T, (3), (5), and I_{13}
{1, 2, 4}	(7)	□M V □R	T, (6), E_{16}
{1, 2, 4}	(8)	□(M □ R)	T, (7), E_8
{9}	(9)	M □ R	P
{1, 2, 4, 9}	(10)	(M □ R) □ □(M □ R)	T, (8), (9), and I_9

Proof by contradiction is convenient in some cases.

It can always be eliminated by a conditional proof (CP). Observe that P □ (Q □ □Q) □ □P (1)

In the proof by contradiction, we show

$H_1, H_2, \dots, H_m$ □ C

by showing

$H_1, H_2, \dots, H_m$, □C □ R □ □R (2)

Now (2) can be converted to the following by using rule CP $H_1, H_2, \dots, H_m$ □ □C □ (R □ □R) (3)

From (3), (1) and E1, we can obtain

$H_1, H_2, \dots, H_m$ □ C, the required derivation.

Automatic Theorem Proving :

The shortcomings of procedures used in the process of derivation are :

- **Rule P** allows the introduction of a premise at any point in the derivation. But, it does not suggest either the premise or the step at which it should be introduced.

- **Rule T** allows the introduction of any formula, obtained from the previous formulas. But, there is neither definite choice of such formula nor any guidance for the use of any particular implication and/or equivalence.

- **Rule CP** does not tell anything about the stages at which an antecedent has to be introduced as an assumed premise. It does not indicate the stage, at which it is again incorporated into the conditional.

Because of the above disadvantages, the process of derivation requires skill, experience and intelligence to make the right decision at every step.
So, the process can not carried out mechanically.

Hence, a new set of rules and procedure is required to construct each step of derivation in a specified manner and finally to show whether the conclusion follows from the given premises.

The new formulation is based on the work of **Hao Wang** and consists of :

- 10 rules,
- An axiom schema,
- Rules of well-formed sequents and formulas.

1. ***Variables:*** The upper-case letters A, B, …, P, Q, …, Z are used as statement variables and also as statement formulas.

2. ***Connectives:*** The connectives, □, □, V, □, and □ appear in the formulas with same order of precedence. The concept of well-formed formulas is the same.

3. ***String of Formulas:*** A string of formulas is defined as follows :
   a. Any formula is a string of formulas.
   b. If □ and □ are strings of formulas, then □, □ and □, □ are strings of formulas.
   c. Only those strings which are obtained by steps (a) and (b) are strings of formulas, with the exception of the empty string, which is also a string of formulas.

Note: The order in which the formulas appear in any string is not important.

4. ***Sequents:*** If □ and □ are strings of formulas, then □ $\overset{c}{\rightarrow}$ □ is called a sequent in which □ is called as antecedent and □ as consequent of the sequent.

   A sequent □ $\overset{c}{\rightarrow}$ □ is true iff either at least one of the formulas of the antecedent is false or at least one of the formulas of the consequent is true.
   Thus A, B, C $\overset{c}{\rightarrow}$ D, E, F is true iff A □ B □ C □ D V E V F is true. So, the symbol $\overset{c}{\rightarrow}$ is a generalization of the connective □ to strings of formulas.

Similarly, the symbol ⇒ applied to strings of formulas as a generalization of the symbol □. c c

Thus A □ B means "A implies B" or "A □ B is a tautology" and □ ⇒ □ means □ → □ istrue

Sometimes, the sequents may have empty strings of formulas as antecedent or consequent. The empty antecedent is interpreted as logical constant “true” or **T** and empty consequent as the logical constant “false” or **F**.

5. ***Axiom Schema:*** Let □ and □ are strings of formulas, each formula containing a variable only,

   then the sequent □ $\to_c$ □ is an axiom iff □ and □ have at least one variable in common.

   Ex : A, B, C $\to_c$ P, B, R, where A, B, C, P, and R are variables, is an axiom.

   Note : If □ → □ is an axiom then □ ⇒ □.

6. ***Theorem:*** The following sequents are theorems of the system.
   a. Every axiom is a theorem.
   b. If a sequent □ is a theorem and a sequent □ results from □ by using one of the 10 rules given below, then □ is also a theorem.
   c. Sequents obtained by (a) and (b) are the only theorems.

7. ***Rules:*** The following rules are used to combine formulas within strings by introducing connectives. For each of the connective there are two rules :
   a. The introduction of the connective in the antecedent.
   b. The introduction of the connective in the consequent.

   In the description of these rules,
   □, □, γ, … are strings of formulas, and
   X and Y are formulas to which the connectives are applied.

Antecedent Rules:

Rule □□ : If □, □ ⇒ X, γ, then □, □X, □ ⇒ γ.

Rule □□ : If X, Y, □, □ ⇒ γ, then □, X □ Y, □ ⇒ γ.

Rule V□ : If X, □, □ ⇒ γ and Y, □, □ ⇒ γ, then □, X V Y, □ ⇒ γ.

Rule □□ : If Y, □, □ ⇒ γ and □, □ ⇒ X, γ, then □, X □ Y, □ ⇒ γ.

Rule □□ : If X, Y, □, □ ⇒ γ and □, □ ⇒ X, Y, γ, then □, X ⇄ Y, □ ⇒ γ.

Consequent Rules:

Rule □□ : If X, □ ⇒ □, γ, then □ ⇒ □, □X, γ.

Rule □□ : If □ ⇒ X, □, γ and □ ⇒ Y, □, γ, then □ ⇒ □, X □ Y, γ.

Rule □V : If □ ⇒ X, Y, □, γ, then □ ⇒ □, X V Y, γ.

Rule □□ : If X, □ ⇒ Y, □, γ, then □ ⇒ □, X □ Y, γ.

Rule □⇄ : If X, □ ⇒ Y, □, γ and Y, □ ⇒ X, □, γ, then □ ⇒ □, X □ Y, γ.

The above method means showing

$H_1, H_2, \ldots, H_m$ □ C (1)

Another way of stating (1) is

H_1 □ (H_2 □ (H_3 □ … (H_m □ C) …)) (2)

is a tautology.

The new formulation is premise-free, so that in order to show that C follows from $H_1, H_2, \ldots, H_m$, we establish that

$\overset{c}{\rightarrow}$ H_1 □ (H_2 □ (H_3 □ … (H_m □ C) …)) (3)

is a theorem.
We must show that

$\overset{c}{\Rightarrow}$ H_1 □ (H_2 □ (H_3 □ … (H_m □ C) …)) (4)

New procedure involves showing (3) to be a theorem. For this, (4) is assumed as true and then show that this assumption is or is not justified.
This can be accomplished by working backward from (4), using the rules and showing that (4) holds if some simple sequent is a theorem.
Note : A simple sequent is a sequent in which some connective is eliminated in one of the formulas appearing in the antecedent or the consequent.

Working backward continues till the simple possible sequents i.e., those which do not have any connectives are obtained.
If these sequents are axioms then our assumption of (4) is justified.
If at least one of the simplest sequents is not an axiom, then the assumption of (4) is not justified and C does not follow from $H_1, H_2, \ldots, H_m$.

Ex 1: Show that P V Q follows from P.
Solution: We need to show that

(1) $\overset{c}{\Rightarrow}$ P □ (P V Q)

(1) If (2) P $\overset{c}{\Rightarrow}$ P V Q (□□)

(2) If (3) P $\overset{c}{\Rightarrow}$ P, Q (□V)

The connective □ is eliminated in (1) by using the rule □□ and the resulting P $\overset{c}{\Rightarrow}$ P V Q is named as (2). Similarly (3) is obtained from (2) by using the rule □V. Finally (3) is a theorem, because it is an axiom.

Note: "(1) if (2)" means "if (2) then (1)" or "(1) holds if (2)".

c

The actual derivation is reversal of these steps in which (3) is an axiom that leads to ⇒ P □ (P V Q) as shown

c

a> P ⇒ P, Q   Axiom

c

b> P ⇒ P V Q   Rule (□V), (a)

c

c> ⇒ P □ (P V Q)   Rule (□□), (b)

UNIT-III

Syllabus

Relations: Properties of Binary Relations, equivalence, transitive closure, compatibility and partial ordering relations, Hasse diagram.
Functions: Inverse Function, Composition of functions, recursive Functions, Lattices and its Properties

Relations

Introduction:

In real life, the word "relation" suggest relations such as the relation of father to son, mother to son, brother to sister, etc.
In arithmetic, the relations are such as "greater than", "less than", or "equality" between two real numbers.
In computer science, a relation is the act of comparing objects which are related to one another. Computer performs different tasks based on the result of comparison.

Here, only binary relations between a pair of objects are considered. A relation between two objects can be defined by listing the two objects as an ordered pair.
A set of all such ordered pairs, in each of which the first member has some definite relationship to the second, describes a particular ***relationship***.

Definition: Any set of ordered pairs defines a ***binary relation***.

A binary relation simply called as a relation. A particular ordered pair, say <x, y> □ R, where R is a relation, can be expressed by writing xRy, read as "x is in relation R to y".

In mathematics, relations are denoted by special symbols rather than by upper-case letters.

< = {<x, y>□ x, y are real numbers and x < y}

The relation of father to his child can be described by a set F, of ordered pairs. In ordered pair, first member is the name of the father and the second is of child.

F = {<x, y> □ x is the father of y}

The definition of relation permits any set of ordered pairs to define a relation. Let us consider the set S given by

S = {<2, 4>, <1, 3>, <□, 6>, <□, □>}

can be treated as a relation, but it is not much useful.

Let R denote the set of real numbers. Then

$Q = \{<x^2, x> \ \square\ x \ \square\ R\}$ defines the relation of the square of a real number.

Definition: Let S be a binary relation.

The set D(S) of all objects x such that for some y, $<x, y> \ \square\ S$ is called the ***domain*** of S. $D(S) = \{x \ \square\ (\square y) (<x, y> \ \square\ S)\}$

The set R(S) of all objects y such that for some x, $<x, y> \ \square\ S$ is called the ***range*** of S. $R(S) = \{y \ \square\ (\square x) (<x, y> \ \square\ S)\}$

Ex: For relation S = {<2, 4>, <1, 3>, <□, 6>, <□, □>}

D(S) = {2, 1, □, □}and R(S) = {4, 3, 6, □}.

Let A and B be any two sets. A subset of the cartesian product A x B defines a relation, C. For C, D(C) ⊆ A and R(C) ⊆ B. The relation C is said to be from A to B.

If B = A, then C is said to be a relation from A to A. C is called a relation in A. Any relation in A is a subset of A x A.

The set A x A defines a relation in A and is called a ***universal relation*** in A. The empty set which is also a subset of A x A is called a ***void relation*** in A.

Since relation is defined as a set of ordered pairs, the usual operations of sets are applied to relations also.

Let R and S denote two relations. x (R

∩ S) y □ x R y ∧ x S y x

(R ∪ S) y □ x R y ∨ x S y x

(R – S) y □ x R y ∧ x S y x

(~ R) y □ x R y

Ex: Let X = {1, 2, 3, 4}. If

R = {<x, y> □ x □ X ∧ y □ X ∧ ((x – y) is an integral nonzero multiple of 2)} S = {<x, y> □ x □ X ∧ y □ X ∧ ((x – y) is an integral nonzero multiple of 3)}

Find R ∪ S, R ∩ S, and R - S.

Solution:

R = {<1, 3>, <3, 1>, <2, 4>, <4, 2>}

S = {<1, 4>, <4, 1>}

R ∪ S = {<1, 3>, <3, 1>, <2, 4>, <4, 2>, <1, 4>, <4, 1>}

R ∩ S = ϕ

R – S = {<1, 3>, <3, 1>, <2, 4>, <4, 2>}

Ex: Let A = {1, 2, 3} and B = {1, 2, 3, 4}. The relations R_1 = {<1, 1>, <2, 2>, <3, 3>} and

R_2 = {<1, 1>, <1, 2>, <1, 3>, <1, 4>}. Find $R_1 \cup R_2$, $R_1 \cap R_2$, $R_1 - R_2$, and $R_2 - R_1$.

Solution:

$R_1 \cup R_2$ = {<1, 1>, <2, 2>, <3, 3>, <1, 2>, <1, 3>, <1, 4>},

$R_1 \cap R_2$ = {<1, 1>},

$R_1 - R_2$ = {<2, 2>, <3, 3>},

$R_2 - R_1$ = {<1, 2>, <1, 3>, <1, 4>}.

Properties of Binary Relations :

Reflexive: A binary relation R in a set A is ***reflexive*** if, for every x □ A, x R x, i.e., <x, x> □ R. R is reflexive in A □ (x) (x □ A □ x R x)

Ex: Let A = {1, 2, 3, 4}

R = {<1, 1>, <2, 2>, <3, 3>, <4, 4>}

Note: The relations □, □ and = are reflexive in the set of real numbers.

Symmetric: A relation R in a set A is ***symmetric*** if, for every x and y in A, whenever x R y, then y R x.

R is symmetric in A □ (x) (y) (x □ A ⋀ y □ A ⋀ x R y □ y R x)

Ex: Let A = {1, 2, 3, 4}

R = {<1, 2>, <2, 1>, <2, 3>, <3, 2>, <3, 4>, <4, 3>, <4, 4>}

Note: The relation = is symmetric in the set of real numbers.

Transitive: A relation in a set A is ***transitive*** if, for every x, y, and z in A, whenever x R y and y R z, then x R z.

R is transitive in A □ (x) (y) (z) (x □ A ⋀ y □ A ⋀ z □ A ⋀ x R y ⋀ y R z □ x R z)

Ex: Let A = {1, 2, 3, 4}

R = {<1, 2>, <2, 3>, <1, 3>, <2, 3>, <3, 4>, <2, 4>}

Note: The relations □, <, >, □, and = are transitive in the set of real numbers.

Irreflexive: A relation R in a set A is ***irreflexive*** if, for every x □ A, <x, x> □ R. R is irreflexive in A □ (x) (x □ A □ x R x)

Ex: Let A = {1, 2, 3, 4}

R = {<1, 2>, <1, 3>, <2, 3>, <4, 3>, <3, 4>, <2, 4>, <1, 4>}

Note: The relations < and > are irreflexive.

Antisymmetric: A relation R in a set A is ***antisymmetric*** if, for every x and y in A, whenever x R y and y R x, then x = y.

R is antisymmetric in A □ (x) (y) (x □ A ⋀ y □ A ⋀ x R y ⋀ y R x □ x = y)

Ex: Let A = {1, 2, 3, 4}

R = {<1, 2>, <1, 3>, <2, 3>, <4, 4>, <3, 4>, <2, 4>, <1, 4>}

Transitive Closure: A relation R' is the transitive closure of a relation R iff a> R' is transitive

b> R ⊆ R'

c> For any relation R", if R ⊆ R" and R" is transitive, then R' ⊆ R", i.e., R' is the smallest relation that satisfies (a) and (b).

The transitive closure of a relation R is denoted as t(R).

Ex : Let S = {1, 2, 3} and R = {<1, 1>, <1, 2>, <1, 3>, <2, 3>, <3, 1>.

Find the transitive closure of R.

Solution:

<2, 3> □ R ⋀ <3, 1> □ R □ <2, 1> □ R'

<3, 1> □ R ⋀ <1, 2> □ R □ <3, 2> □ R'

<3, 1> □ R ⋀ <1, 3> □ R □ <3, 3> □ R'

<2, 1> □ R' ⋀ <1, 2> □ R □ <2, 2> □ R'

□ R' ={<1, 1>, <1, 2>, <1, 3>, <2, 1>, <2, 2>, <2, 3>, <3, 1>, <3, 2>, <3, 3>}

Representing Relations using Matrices:

A relation between finite sets can be represented using a zero–one matrix. Let R be a relation from A = {a1, a2, …, am} to B = {b1, b2, …, bn}.

The relation R can be represented by the matrix $M_{\mathbf{R}} = [m_{ij}]$, where

$$m_{ij} = \begin{cases} 1 \text{ if } (a_i, b_j) \square R \\ 0 \text{ if } (a_i, b_j) \square R \end{cases}$$

The zero–one matrix representing R has a 1 as its (i, j) entry when a_i is related to b_j, and a 0 in this position if a_i is not related to b_j .
The resulting matrix is of size m X n.

Ex: Let A = {1, 2, 3} and B = {1, 2}. R be the relation from A to B containing <a, b>, if a □ A, b □ B, and a > b. What is the matrix representing R if $a_1 = 1$, $a_2 = 2$, and $a_3 = 3$, and $b_1 = 1$ and $b_2 = 2$?

Solution:

Here, R = {<2, 1>, <3, 1>, <3, 2>}, the matrix for R is

$$M_R = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 1 & 1 \end{bmatrix}$$

Ex: Let A = {a1, a2, a3} and B = {b1, b2, b3, b4, b5}. Which ordered pairs are in the relation R represented by the matrix

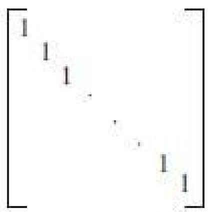
$$M_R = \begin{bmatrix} \mathbf{0} & \mathbf{1} & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix} ?$$

Solution:

$R = \{\langle a_1, b_2\rangle, \langle a_2, b_1\rangle, \langle a_2, b_3\rangle, \langle a_2, b_4\rangle, \langle a_3, b_1\rangle, \langle a_3, b_3\rangle, \langle a_3, b_5\rangle\}$

The matrix of a relation on a set, which is a square matrix, can be used to determine whether the relation has certain properties.

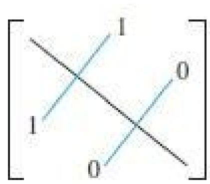
A relation R on A is ***reflexive*** if $\langle a, a\rangle \in R$ whenever $a \in A$. Thus, R is reflexive iff $\langle a_i, a_i\rangle \in R$ for $i = 1, 2, \ldots, n$.
Hence, R is reflexive iff $m_{ij} = 1$, for $i = 1, 2, \ldots, n$.
In other words, R is reflexive if all the elements on the main diagonal of M_R are equal to 1.
Note: The other elements can be either 0 or 1. The
form of the matrix for a reflexive relation is:



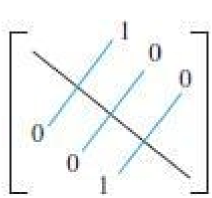
The relation R is ***symmetric*** if $\langle a, b\rangle \in R$ implies that $\langle b, a\rangle \in R$.
So, the relation R on the set $A = \{a_1, a_2, \ldots, a_n\}$ is symmetric iff $\langle a_j, a_i\rangle \in R$ whenever $\langle a_i, a_j\rangle \in R$.

In terms of the entries of M_R, R is symmetric iff $m_{ji} = 1$ whenever $m_{ij} = 1$. This also means $m_{ji} = 0$ whenever $m_{ij} = 0$.
So, R is symmetric iff $m_{ij} = m_{ji}$, for all pairs of integers i and j with $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, n$.
The relation R is symmetric iff $M_R = (M_R)^T$

The form of the matrix for a symmetric relation is:



The relation R is antisymmetric iff $\langle a, b\rangle \in R$ and $\langle b, a\rangle \in R$ imply that $a = b$.
So, the matrix of an antisymmetric relation has the property that if $m_{ij} = 1$ with $i \neq j$, then $m_{ji} = 0$. In other words, either $m_{ij} = 0$ or $m_{ji} = 0$ when $i \neq j$.
The form of the matrix for an antisymmetric relation is:



Ex: Let the relation R on a set A is represented by the matrix

1 1 0
[1 1 1]

0 1 1

Is R reflexive, symmetric, and/or antisymmetric?

Solution:

Since all the diagonal elements are 1, the relation R is reflexive. Since $a_{ij} = a_{ji}$, the relation R is symmetric.

The relation is antisymmetric.

The matrices also can be used to represent the union and the intersection of two relations. Let R and S are relations on a set A represented by the matrices M_R and M_S , respectively.

The matrix representing the union of these relations has a 1 in the positions where either M_R or M_S has a 1.

The matrix representing the intersection of these relations has a 1 in the positions where both M_R and M_S have a 1.

$\therefore\ M_{R \cup S} = M_R \vee M_S$ and $M_{R \cap S} = M_R \wedge M_S$

Ex: Let the relations R and S on a set A are represented by the matrices

$$M_R = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \text{ and } M_S = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

What are the matrices representing $R \cup S$ and $R \cap S$?

Solution:

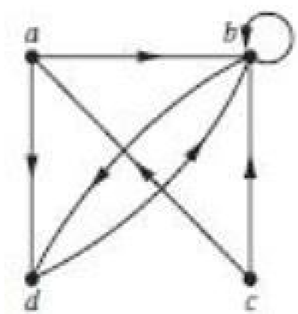
The resulting matrices of these relations are

$$M_{R \cup S} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix} \text{ and } M_{R \cap S} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Representing Relations using Digraphs:

A directed graph, or digraph, consists of a non-empty set V of vertices (or nodes) together with a set E of ordered pairs of elements of V called edges (or arcs).
The vertex ***a*** is called the initial vertex of the edge <a, b>, and the vertex ***b*** is called the terminal vertex of this edge.
An edge of the form <a, a> is represented using an arc from the vertex ***a*** back to itself. Such an edge is called a **loop**.

Ex: The directed graph with vertices a, b, c, and d, and edges <a, b>, <a, d>, <b, b>, <b, d>, <c, a>, <c, b>, and <d, b> is as shown below.



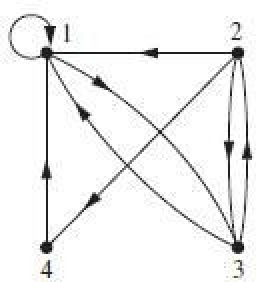
Directed graphs give a visual display of information about relations.
The relations from a set A to a set B can be represented by a directed graph where there is a vertex for each element of A and a vertex for each element of B
The relation R on a set A is represented by the directed graph that has the elements of A as its vertices and the ordered pairs <a, b>, where <a, b> □ R, as edges.

Ex: Find the directed graph of the relation

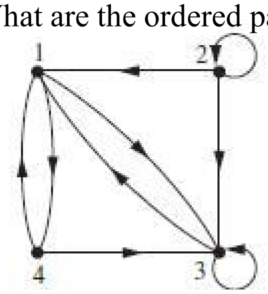
R = {<1, 1>, <1, 3>, <2, 1>, <2, 3>, <2, 4>, <3, 1>, <3, 2>, <4, 1>}

on the set {1, 2, 3, 4}.

Solution:



Ex: What are the ordered pairs in the relation R represented by the directed graph shown below?



Solution: The ordered pairs <x, y> in the relation are

R = {<1, 3>, <1, 4>, <2, 1>, <2, 2>, <2, 3>, <3, 1>, <3, 3>, <4, 1>, <4, 3>}.

Each of these pairs corresponds to an edge of the directed graph, with <2, 2> and <3, 3> corresponding to loops.

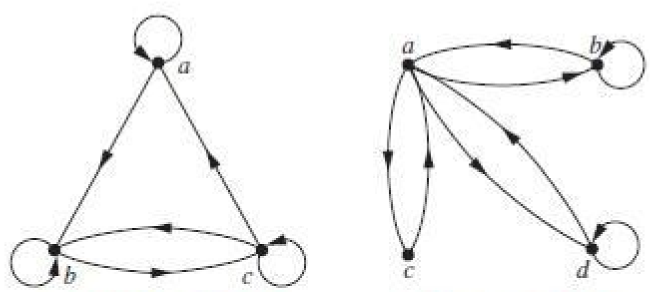
The directed graph representing a relation can be used to determine whether the relation has various properties. A relation is ***reflexive*** iff there is a loop at every vertex of the directed graph, so that every ordered pair of the form <x, x> occurs in the relation.

A relation is ***symmetric*** iff for every edge between distinct vertices in its digraph there is an edge in the opposite direction, so that <y, x> is in the relation whenever <x, y> is in the relation.

A relation is ***antisymmetric*** iff there are never two edges in opposite directions between distinct vertices.

A relation is ***transitive*** iff whenever there is an edge from a vertex x to a vertex y and an edge from a vertex y to a vertex z, there is an edge from x to z.

Ex: Determine whether the relations for the directed graphs shown below are reflexive, symmetric, antisymmetric, and/or transitive.



(a) Directed graph of R (b) Directed graph of S

Solution:

Because there are loops at every vertex of the directed graph of R, it is reflexive.

R is neither symmetric nor antisymmetric because there is an edge from a to b but not one from b to a, but there are edges in both directions connecting b and c.

Finally, R is not transitive because there is an edge from a to b and an edge from b to c, but no edge from a to c.

Because loops are not present at all the vertices of the directed graph of S, this relation is not reflexive.

It is symmetric and not antisymmetric, because every edge between distinct vertices is accompanied by an edge in the opposite direction.

The relation S is not transitive, because <c, a> and <a, b> belong to S, but <c, b> does not belong to S.

Equivalence Relation: A relation R in a set X is called an ***equivalence relation*** if it is reflexive, symmetric, and transitive.

If R is an equivalence relation in a set X, then D(R), the domain of R, is X itself. Therefore R will be called a relation on X.

Two elements a and b that are related by an equivalence relation are called equivalent. The notation a ~ b is often used to denote that a and b are equivalent elements with respect to a particular equivalence relation.

Ex: Let R be the relation on the set of real numbers such that a R b iff a − b is an integer. Is R an equivalence relation?

Solution:

Because a −a = 0 is an integer for all real numbers a, a R a for all real numbers a. Hence, R is reflexive.
Now suppose that a R b, then a −b is an integer, so b−a is also an integer. It follows that R is symmetric.
If a R b and b R c, then a −b and b−c are integers. Therefore, a −c = (a −b) + (b−c) is also an integer. Hence, a R c. Thus, R is transitive.
Consequently, R is an equivalence relation.

Let R be an equivalence relation on a set A. For any x □ A, the set $[x]_R \subseteq A$ given by $[x]_R$ = {y □ y □ A ∧ x R y}
is called an ***R-equivalence class*** generated by x □ A. Sometimes $[x]_R$ is also written as x/R.

Ex: Let Z be the set of integers and R be the relation called "congruence modulo 3" defined by R = {<x, y> □ x □ Z ∧ y □ Z ∧ (x – y) is divisible by 3}
Determine the equivalence classes generated by the elements of Z.

Solution: The equivalence classes are :

$[0]_R = \{\ldots, -6, -3, 0, 3, 6, \ldots\}$
$[1]_R = \{\ldots, -5, -2, 1, 4, 7, \ldots\}$
$[2]_R = \{\ldots, -4, -1, 2, 5, 8, \ldots\}$
$Z/R = \{[0]_R, [1]_R, [2]_R\}$

Compatibility Relation: A relation R in A is said to be a ***compatibility relation*** if it is reflexive and symmetric.

Two elements a and b that are related by a compatibility relation are called compatible. The notation a ≈ b is often used to denote that a and b are compatible elements with respect to a particular compatibility relation.

Partial Ordering:

A relation R on a set S is called a partial ordering or partial order if it is reflexive, antisymmetric, and transitive. A set S together with a partial ordering R is called a partially ordered set, or poset, and is denoted by (S, R). Members of S are called elements of the poset.

Ex: Show that the "greater than or equal" relation (≥) is a partial ordering on the set of integers.

Solution:

Since a ≥ a for every integer a, ≥ is reflexive.
If a ≥ b and b ≥ a, then a = b. Hence, ≥ is antisymmetric. Finally, ≥ is transitive because a ≥ b and b ≥ c imply that a ≥ c.
It follows that ≥ is a partial ordering on the set of integers and (Z, ≥) is a poset.

Ex: Show that the inclusion relation ⊆ is a partial ordering on the power set of a set S.

Solution:

Since A ⊆ A whenever A is a subset of S, ⊆ isreflexive.
It is antisymmetric because A ⊆ B and B ⊆ A imply that A = B. Finally, ⊆ is transitive, because A ⊆ B and B ⊆ C imply that A ⊆ C. Hence, ⊆ is a partial ordering on P(S), and (P(S), ⊆) is a poset.

In different posets different symbols are used for a partial ordering. So, the symbol ≤ is commonly used to denote the relation in any poset, not just the "less than or equals" relation.

The elements a and b of a poset (S, ≤) are called ***comparable*** if either a ≤ b or b ≤ a. When a and b are elements of S such that neither a ≤ b nor b ≤ a, a and b are called ***incomparable***.

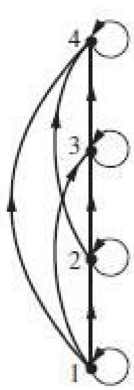
If (S, ≤) is a poset and every two elements of S are comparable, S is called a ***totally ordered or linearly ordered set***, and ≤ is called a ***total order*** or a ***linear order***. A totally ordered set is also called a ***chain***.

(S, ≤) is a ***well-ordered set*** if it is a poset such that ≤ is a total ordering and every nonempty subset of S has a least element.

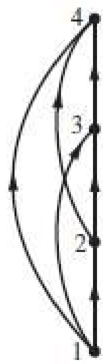
Hasse Diagrams:

Many edges in the directed graph for a finite poset do not have to be shown because they must be present.

Ex: Let us consider the directed graph for the partial ordering {<a, b> | a ≤ b} on the set {1, 2, 3, 4}, as shown below.



Because this relation is a partial ordering, it is reflexive, and its directed graph has loops at all vertices.
So, there is no need to show these loops because they must be present. The modified figure without the loops is:



Because a partial ordering is transitive, there is no need to show those edges that must be present because of transitivity.
Here, the edges (1, 3), (1, 4), and (2, 4) need not shown because they must be present.
If we assume that all edges are pointed “upward” (as they are drawn in the figure), no need to show the directions of the edges.
The resulting figure is:

eneral, a finite poset (S, ≤) can be represented using this procedure: Start with the directed graph for this relation.
Since a partial ordering is reflexive, a loop (a, a) is present at every vertex a. Remove these oops.
Next, remove all edges that must be in the partial ordering because of the presence of other edges and transitivity, i.e., remove all edges (x, y) for which there is an element z ∈ S such hat x □ z and z □ x.
Finally, arrange each edge so that its initial vertex is below its terminal vertex. Remove all he arrows on the directed edges, because all edges point “upward” toward their terminal vertex.

resulting diagram is called the **Hasse diagram** of (S, □), named after the twentieth-century man mathematician Helmut Hasse, who made extensive use of them.

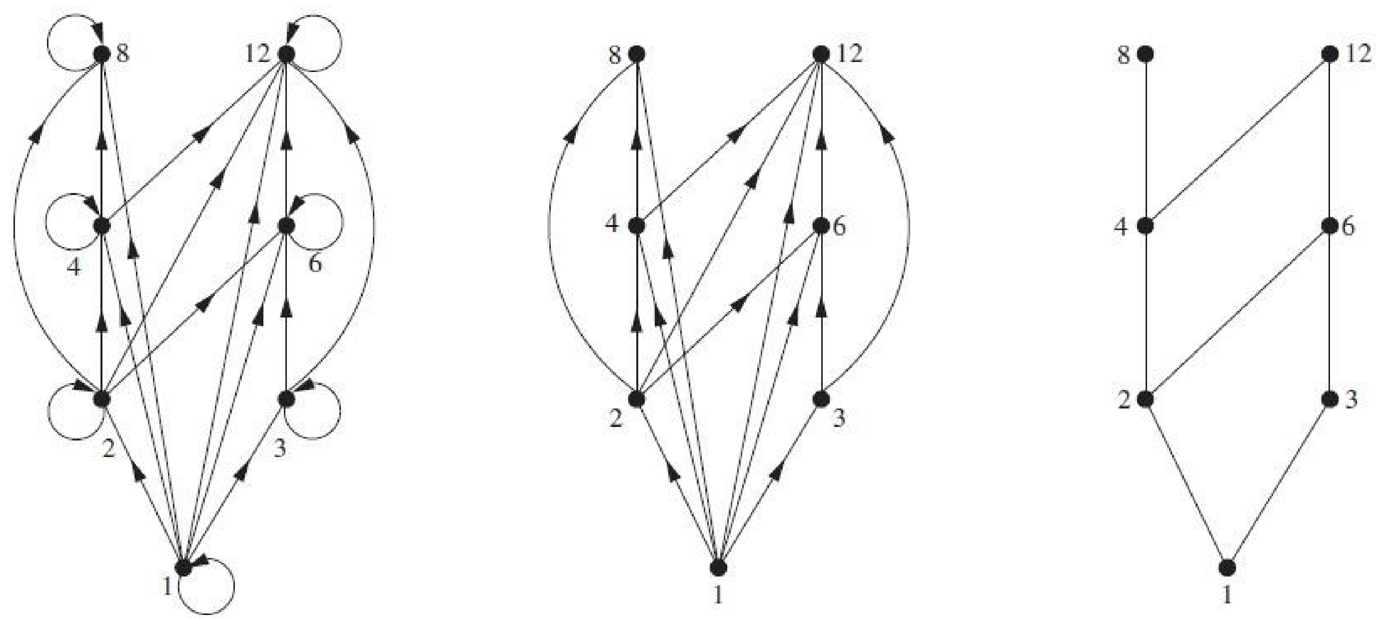
Let (S, □) be a poset. We say that an element y □ S **covers** an element x □ S if x □ y and there is no element z □ S such that x □ z□ y.
The set of pairs <x, y> such that y covers x is called the **covering relation** of (S, □).

From the description of the Hasse diagram of a poset, it is observed that the upward pointing edges of the Hasse diagram corresponding to the pairs in the covering relation of (S, □).

Ex: Draw the Hasse diagram representing the partial ordering {(a, b) | a divides b} on the set {1, 2, 3, 4, 6, 8, 12}.

Solution:

Begin with the digraph for this partial order, as shown in Figure (a). Remove all loops, as shown in Figure (b).
Then delete all the edges implied by the transitive property. These are (1, 4), (1, 6), (1, 8), (1, 12), (2, 8), (2, 12), and (3, 12).
Arrange all edges to point upward, and delete all arrows to obtain the Hasse diagram. The resulting Hasse diagram is shown in Figure (c).



(a) (b) (c)

Elements of posets that have certain extremal properties are important for many applications. An element of a

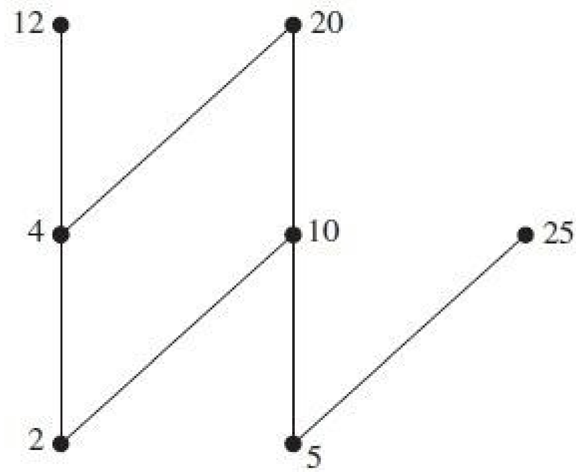
poset is called ***maximal*** if it is not less than any element of the poset. That is, ***a*** is **maximal** in the poset (S, □) if there is no b □ S such that a □ b.

Similarly, an element of a poset is called ***minimal*** if it is not greater than any element of the poset. That is, ***a*** is **minimal** if there is no element b □ S such that b □ a.

Maximal and minimal elements are the “top” and “bottom” elements in the Hasse diagram.

Ex: Which elements of the poset ({2, 4, 5, 10, 12, 20, 25}, |) are maximal, and which are minimal?

Solution: The Hasse diagram shown below for this poset shows that the maximal elements are 12, 20, and 25, and the minimal elements are 2 and 5.



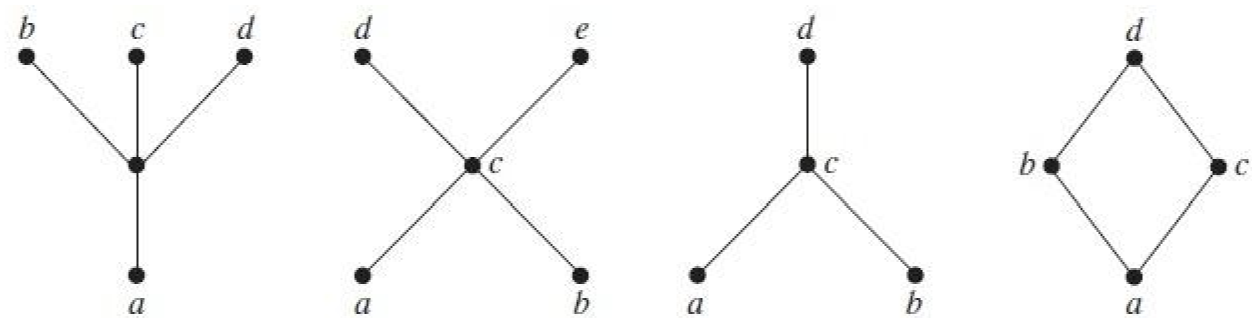
Note: A poset can have more than one maximal element and more than one minimal element.

The element of a poset which is greater than every other element, is called the ***greatest element***, i.e., ***a*** is the **greatest element** of the poset (S, □) if b □ a for all b □ S.

The greatest element is unique when it exists.

Similarly, an element is called the ***least element*** if it is less than all the other elements in the poset. That is, ***a*** is the **least element** of (S, □) if a □ b for all b □ S.

Ex: Determine whether the posets represented by each of the Hasse diagrams shown below have a greatest element and a least element.



(a) (b) (c) (d)

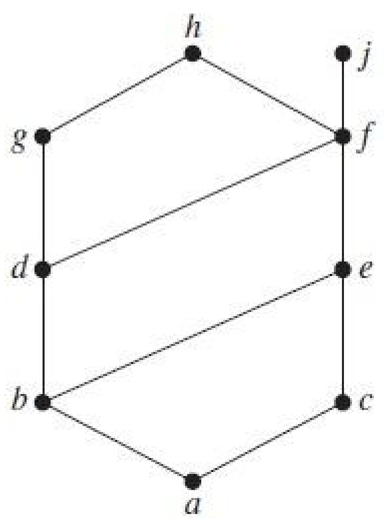
Solution: The least element of the poset with Hasse diagram (a) is a. This poset has no greatest element.
The poset with Hasse diagram (b) has neither a least nor a greatest element.
The poset with Hasse diagram (c) has no least element. Its greatest element is d.

Sometimes, it is required to find an element that is greater than or equal to all the elements in a subset A of a poset (S, $\preceq$).

If u is an element of S such that a $\preceq$ u for all elements a $\in$ A, then u is called an **upper bound** of A.

Similarly, there may be an element less than or equal to all the elements in A. If l is an element of S such that l $\preceq$ a, for all elements a $\in$ A, then l is called a **lower bound** of A.

Ex: Find the lower and upper bounds of the subsets {a, b, c}, {j, h}, and {a, c, d, f } in the poset with the Hasse diagram shown below.



Solution:

The upper bounds of {a, b, c} are d, e, f, j, g, and h, and its only lower bound is a. There are no upper bounds of {j, h}, and its lower bounds are a, b, c, d, e, f, and g. The upper bounds of {a, c, d, f} are g , h, and j , and its lower bound is a.

The element x is called the **least upper bound** of the subset A if x is an upper bound that is less than every other upper bound of A.

Similarly, the element y is called the **greatest lower bound** of A if y is a lower bound of A and z $\preceq$ y whenever z is a lower bound of A.

Note: The greatest lower bound and least upper bound of a subset A are denoted by glb(A) and lub(A), respectively.

A partially ordered set in which every pair of elements has both a least upper bound and a greatest lower bound is called a **lattice**.

Functions

Introduction:

Function: Let A and B be any two sets. A relation **f** from A to B is called a ***function*** if for every x□A there is a unique y□B such that <x, y>□ f.

Terms such as "transformation", "map" (or "mapping"), "correspondence", and "operation" are used as synonyms for "function".

Ex: Let A = {1, 5, P, □), B = {2, 5, 7, q, □} and f = {<1, 2>, <5, 7>, <P, q>, <□, □>. Here, D_f = A, R_f = {2, 7, q, □}, and f(1) = 2, f(5) = 7, f(P) = q, f(□) = □.

If f: X □ Y and A ⊆ X, then f ⋂ (A X Y) is a function A □ Y called as the ***restriction*** of f to A and is sometimes written as f/A. If g is a ***restriction*** of f, then f is called the ***extension*** of g.

A mapping of f: X □ Y is called ***onto*** (surjective, a surjection) if the range R_f = Y; otherwise it is called ***into***.

A mapping of f: X □ Y is called ***one-to-one*** (injective, or 1-1) if distinct elements of X are mapped into distinct elements of Y.
In other words, f is one-to-one if x1 ≠ x2 ⇒ f(x1) ≠ f(x2).

A mapping of f: X □ Y is called ***one-to-one onto*** (bijective) if it is both one-to-one and onto. Such mapping is also called a one-to-one correspondence between X and Y.

Composition of Functions:

Let f: X □ Y and g: Y □ Z be two functions. The composite relation g ∘ f such that g ∘ f =
{<x, z> □ (x □ X) ⋀ (z □ Z) □ (□y) (y □ Y □ y = f(x) □ z = g(y))}
is called the ***composition*** of functions or ***relative product*** of functions f and g. Precisely g ∘ f is called the ***left composition*** of g with f.

Ex: Let X = {1, 2, 3}, Y = {p, q} and Z = {a, b}. Also let f: X □ Y, be f = {<1, p>, <2, p>, <3, q>} and g: Y □ Z be given by g = {<p, b>, <q, b>}. Find g ∘ f.

Solution:

g ∘ f = {<1, b>, <2, b>, <3, b>}

Inverse Functions:

The converse of a relation R from X to Y is a relation Ř from Y to X such that
<y, x> □ Ř □ <x, y> □ R.
So, the ordered pairs of Ř are obtained from those of R by simply interchanging themembers.

But, in the case of functions the same may not be possible in all the cases. If g: X □ Y is a function, then ğ may not be a function.

Ex: Let X = {1, 2, 3}, Y = {p, q, r} and g: X □ Y be given by g = {<1, p>, <2, q>, <3, q>}. Then ğ = {<p, 1>, <q, 2>, <q, 3>} and ğ is not a function.

A mapping I_x: X □ X is called an ***identity map*** if I_x = {<x, x>□ x □ X}.

Note: For any function g: X □ X, the functions g ° I_x and I_x ° g are both equal to g. If f: X □ Y

is invertible, then f^{-1} ° f = I_x and f ° f^{-1} = I_y.

Let f: X □ Y and g: Y □ X. The function g is equal to f^{-1} only if g ° f = I_x and f ° g = I_y.

Ex: Show that the functions f(x) = x^3 and g(x) = $x^{1/3}$ for x □ R are inverses of one another. Solution:

Since (f ° g) (x) = f($x^{1/3}$) = x = I_x, (g ° f)

(x) = g(x^3) = x = I_x.

So f = g^{-1} and g = f^{-1}.

UNIT-IV

Before we discuss permutations we are going to have a look at what the words combination means and permutation. A Waldorf salad is a mix of among other things celeriac, walnuts and lettuce. It doesn't matter in what order we add our ingredients but if we have a combination to our padlock that is 4-5-6 then the order is extremely important.

If the order doesn't matter then we have a combination, if the order does matter then we have a permutation. One could say that a permutation is an ordered combination.

The number of permutations of n objects taken r at a time is determined by the following formula:

P(n,r)=n!(n−r)!P(n,r)=n!(n−r)!

Example

A code have 4 digits in a specific order, the digits are between 0-9. How many different permutations are there if one digit may only be used once?

A four digit code could be anything between 0000 to 9999, hence there are 10,000 combinations if every digit could be used more than one time but since we are told in the question that one digit only may be used once it limits our number of combinations. In order to determine the correct number of permutations we simply plug in our values into our formula:

P(n,r)=10!(10−4)!=10⋅9⋅8⋅7⋅6⋅5⋅4⋅3⋅2⋅16⋅5⋅4⋅3⋅2⋅1=5040P(n,r)=10!(10−4)!=10⋅9⋅8⋅7⋅6⋅5⋅4⋅3⋅2⋅16⋅5⋅4⋅3⋅2⋅1=5040

In our example the order of the digits were important, if the order didn't matter we would have what is the definition of a combination. The number of combinations of n objects taken r at a time is determined by the following formula:

C(n,r)=n!(n−r)!r!

Multinomial theorem, in algebra, a generalization of the binomial theorem to more than two variables. In statistics, the corresponding multinomial series appears in the multinomial distribution, which is a generalization of the binomial distribution. The multinomial theorem provides a formula for expanding an expression such as $(x_1 + x_2 +\cdots+ x_k)^n$ for integer values of n. In particular, the expansion is given by

$$(x_1 + x_2 + \cdots + x_k)^n = \sum_{n_1, n_2, \ldots, n_k \geq 0} \frac{n!}{n_1! n_2! \cdots n_k!} x_1^{n_1} x_2^{n_2} \cdots x_k^{n_k},$$

where $n_1 + n_2 +\cdots+ n_k = n$ and $n!$ is the factorial notation for $1 \times 2 \times 3 \times\cdots\times n$.

For example, the expansion of $(x_1 + x_2 + x_3)^3$ is $x_1^3 + 3x_1^2x_2 + 3x_1^2x_3 + 3x_1x_2^2 + 3x_1x_3^2 + 6x_1x_2x_3 + x_2^3 + 3x_2^2x_3 +$

$3x_2x_3^2 + x_3^3$.
Let $|A|$ denote the cardinal number of set A, then it follows immediately that

$$|A \cup B| = |A| + |B| - |A \cap B|, \tag{1}$$

where $\cup$ denotes union, and $\cap$ denotes intersection. The more general statement

$$\left|\bigcup_{i=1}^{N} E_i\right| \leq \sum_{i=1}^{N} |E_i|, \tag{2}$$

also holds, and is known as Boole's inequality or one of the Bonferroni inequalities.
This formula can be generalized in the following beautiful manner. Let $\mathcal{A} = \{A_i\}_{i=1}^{p}$ be a *p*-system of S consisting of sets A_1, ..., A_p, then

$$\begin{aligned}\left|A_1 \cup A_2 \cup \ldots \cup A_p\right| = \sum_{1 \leq i \leq p} |A_i| - \sum_{1 \leq i_1 < i_2 \leq p} \left|A_{i_1} \cap A_{i_2}\right| + \\ \sum_{1 \leq i_1 < i_2 < i_3 \leq p} \left|A_{i_1} \cap A_{i_2} \cap A_{i_3}\right| - \ldots + (-1)^{p-1} \left|A_1 \cap A_2 \cap \cdots \cap A_p\right|,\end{aligned} \tag{3}$$

where the sums are taken over *k*-subsets of $\mathcal{A}$. This formula holds for infinite sets S as well as finite sets (Comtet 1974, p. 177).
The principle of inclusion-exclusion was used by Nicholas Bernoulli to solve the recontres problem of finding the number of derangements (Bhatnagar 1995, p. 8).
For example, for the three subsets $A_1 = \{2, 3, 7, 9, 10\}$, $A_2 = \{1, 2, 3, 9\}$, and $A_3 = \{2, 4, 9, 10\}$ of $S = \{1, 2, \ldots, 10\}$, the following table summarizes the terms appearing the sum.

#	term	set	length
1	A_1	$\{2, 3, 7, 9, 10\}$	5
	A_2	$\{1, 2, 3, 9\}$	4
	A_3	$\{2, 4, 9, 10\}$	4
2	$A_1 \cap A_2$	$\{2, 3, 9\}$	3
	$A_1 \cap A_3$	$\{2, 9, 10\}$	3
	$A_2 \cap A_3$	$\{2, 9\}$	2
3	$A_1 \cap A_2 \cap A_3$	$\{2, 9\}$	2

$|A_1 \cup A_2 \cup A_3|$ is therefore equal to $(5 + 4 + 4) - (3 + 3 + 2) + 2 = 7$, corresponding to the seven elements $A_1 \cup A_2 \cup A_3 = \{1, 2, 3, 4, 7, 9, 10\}$

UNIT-V

Generating Function

Generating functions are used to represent sequences efficiently by coding the terms of a sequence as coefficients of powers of a variable (say) in a formal power series.

Now with the formal definition done, we can take a minute to discuss *why* should we learn this concept.
This concept can be applied to solve many problems in mathematics. There is a huge chunk of mathematics dealing with just generating functions.

- It can be used to solve various kinds of Counting problems easily.

- It can be used to solve recurrence relations by translating the relation in terms of *sequence* to a problem about *functions*.
- It can be used to prove combinatorial identities.

In simple words generating functions can be used to translate problems about *sequences* to problems about *functions* which are comparatively easy to solve using maneuvers.

There is an extremely powerful tool in discrete mathematics used to manipulate sequences called the generating function. The idea is this: instead of an infinite sequence (for example: 2,3,5,8,12,…2,3,5,8,12,…) we look at a single function which encodes the sequence. But not a function which gives the nnth term as output. Instead, a function whose power series (like from calculus) "displays" the terms of the sequence. So for example, we would look at the power series 2+3x+5x2+8x3+12x4+⋯2+3x+5x2+8x3+12x4+⋯ which displays the sequence 2,3,5,8,12,…2,3,5,8,12,… as coefficients.

An infinite power series is simply an infinite sum of terms of the form cnxncnxn were cncn is some constant. So we might write a power series like this:

∞∑k=0ckxk.∑k=0∞ckxk.

or expanded like this

c0+c1x+c2x2+c3x3+c4x4+c5x5+⋯.c0+c1x+c2x2+c3x3+c4x4+c5x5+⋯.

When viewed in the context of generating functions, we call such a power series a *generating series*. The generating series generates the sequence

c0,c1,c2,c3,c4,c5,….c0,c1,c2,c3,c4,c5,….

In other words, the sequence generated by a generating series is simply the sequence of *coefficients* of the infinite polynomial.

Example5.1.1

What sequence is represented by the generating series 3+8x2+x3+x57+100x6+⋯?3+8x2+x3+x57+100x6+⋯?

Solution

Now you might very naturally ask why we would do such a thing. One reason is that encoding a sequence with a power series helps us keep track of which term is which in the sequence. For example, if we write the sequence 1,3,4,6,9,…,24,41,…1,3,4,6,9,…,24,41,… it is impossible to determine which term 2424 is (even if we agreed that the first term was supposed to be a0a0). However, if we wrote the generating series instead, we would have 1+3x+4x2+6x3+9x4+⋯+24x17+41x18+⋯.1+3x+4x2+6x3+9x4+⋯+24x17+41x18+⋯. Now it is clear that 24 is the 17th term of the sequence (that is, a17=24a17=24). Of course to get this benefit we could have displayed our sequence in any number of ways, perhaps $\boxed{1}0\boxed{3}1\boxed{4}2\boxed{6}3\boxed{9}4\cdots\boxed{24}17\boxed{41}18\cdots$,1031426394⋯24174118⋯, but we do not do this. The reason is that the generating series looks like an ordinary power series (although we are interpreting it differently) so we can do things with it that we ordinarily do with power series such as write down what it converges to.

For example, from calculus we know that the power series 1+x+x22+x36+x424+⋯+xnn!+⋯1+x+x22+x36+x424+⋯+xnn!+⋯ converges to the function ex.ex. So we can use exex as a way of talking about the sequence of coefficients of the power series for ex.ex. When we write down a nice compact function which has an infinite power series that we view as a generating series, then we call that function a *generating function*. In this example, we would say

1,1,12,16,124,…,1n!,… has generating function ex1,1,12,16,124,…,1n!,… has generating function example

Recurrence Relation

A recurrence relation is an equation that recursively defines a sequence where the next term is a function of the previous terms (Expressing F_n as some combination of F_i with $i<n$).

Example − Fibonacci series − $F_n = F_{n-1} + F_{n-2}$, Tower of Hanoi − $F_n = 2F_{n-1} + 1$

Linear Recurrence Relations

A linear recurrence equation of degree k or order k is a recurrence equation which is in the format $x_n = A_1 x_{n-1} + A_2 x_{n-1} + A_3 x_{n-1} + \dots A_k x_{n-k}$ (A_n is a constant and $A_k \neq 0$) on a sequence of numbers as a first-degree polynomial

How to solve linear recurrence relation

Suppose, a two ordered linear recurrence relation is − $F_n = AF_{n-1} + BF_{n-2}$ where A and B are real numbers.

The characteristic equation for the above recurrence relation is −

$$x^2 - Ax - B = 0$$

Three cases may occur while finding the roots −

Case 1 − If this equation factors as $(x - x_1)(x - x_1) = 0$ and it produces two distinct real roots x_1 and x_2, then $F_n = ax_1^n + bx_2^n$ is the solution. [Here, a and b are constants]

Case 2 − If this equation factors as $(x - x_1)^2 = 0$ and it produces single real root x_1, then $F_n = ax_1^n + bnx_1^n$ is the solution.

Case 3 − If the equation produces two distinct complex roots, x_1 and x_2 in polar form $x_1 = r\angle\theta$ and $x_2 = r\angle(-\theta)$, then $F_n = r^n(a\cos(n\theta) + b\sin(n\theta))$ is the solution.

Problem 1

Solve the recurrence relation $F_n = 5F_{n-1} - 6F_{n-2}$ where $F_0 = 1$ and $F_1 = 4$

Solution

The characteristic equation of the recurrence relation is −

$$x^2 - 5x + 6 = 0,$$

So, $(x - 3)(x - 2) = 0$

Hence, the roots are −

$x_1 = 3$ and $x_2 = 2$

The roots are real and distinct. So, this is in the form of case 1

Hence, the solution is −

$$F_n = ax_1^n + bx_2^n$$

Here, $F_n = a3^n + b2^n$ (As $x_1 = 3$ and $x_2 = 2$)

Therefore,

$1 = F_0 = a3^0 + b2^0 = a + b$

$4 = F_1 = a3^1 + b2^1 = 3a + 2b$

Solving these two equations, we get $a = 2$ and $b = -1$

Hence, the final solution is −

Fn=2.3n+(−1).2n=2.3n−2nFn=2.3n+(−1).2n=2.3n−2n

Problem 2

Solve the recurrence relation − Fn=10Fn−1−25Fn−2Fn=10Fn−1−25Fn−2 where F0=3F0=3 and F1=17F1=17

Solution

The characteristic equation of the recurrence relation is −

x2−10x−25=0x2−10x−25=0

So (x−5)2=0(x−5)2=0
Hence, there is single real root x1=5x1=5

As there is single real valued root, this is in the form of case 2

Hence, the solution is −

Fn=axn1+bnxn1Fn=ax1n+bnx1n
3=F0=a.50+(b)(0.5)0=a3=F0=a.50+(b)(0.5)0=a
17=F1=a.51+b.1.51=5a+5b17=F1=a.51+b.1.51=5a+5b
Solving these two equations, we get a=3a=3 and b=2/5b=2/5
Hence, the final solution is − Fn=3.5n+(2/5).n.2nFn=3.5n+(2/5).n.2n

Problem 3

Solve the recurrence relation Fn=2Fn−1−2Fn−2Fn=2Fn−1−2Fn−2 where F0=1F0=1 and F1=3F1=3

Solution

The characteristic equation of the recurrence relation is −

x2−2x−2=0x2−2x−2=0

Hence, the roots are −

x1=1+ix1=1+i and x2=1−ix2=1−i

In polar form,

x1=r∠θx1=r∠θ and x2=r∠(−θ),x2=r∠(−θ), where r=2–√r=2 and θ=π4θ=π4

The roots are imaginary. So, this is in the form of case 3.

Hence, the solution is −

Fn=(2–√)n(acos(n.⊓/4)+bsin(n.⊓/4))Fn=(2)n(acos(n.⊓/4)+bsin(n.⊓/4))
1=F0=(2–√)0(acos(0.⊓/4)+bsin(0.⊓/4))=a1=F0=(2)0(acos(0.⊓/4)+bsin(0.⊓/4))=a
3=F1=(2–√)1(acos(1.⊓/4)+bsin(1.⊓/4))=2–√(a/2–√+b/2–√)3=F1=(2)1(acos(1.⊓/4)+bsin(1.⊓/4))=2(a/2+b/2)
Solving these two equations we get a=1a=1 and b=2b=2

Hence, the final solution is −

Fn=(2–√)n(cos(n.π/4)+2sin(n.π/4))Fn=(2)n(cos(n.π/4)+2sin(n.π/4))

Non-Homogeneous Recurrence Relation and Particular Solutions

A recurrence relation is called non-homogeneous if it is in the form

Fn=AFn−1+BFn−2+f(n)Fn=AFn−1+BFn−2+f(n) where f(n)≠0f(n)≠0

Its associated homogeneous recurrence relation is Fn=AFn–1+BFn−2Fn=AFn–1+BFn−2
The solution (an)(an) of a non-homogeneous recurrence relation has two parts.
First part is the solution (ah)(ah) of the associated homogeneous recurrence relation and the second part is the particular solution (at)(at).

an=ah+atan=ah+at

Solution to the first part is done using the procedures discussed in the previous section.

To find the particular solution, we find an appropriate trial solution.

Let f(n)=cxnf(n)=cxn ; let x2=Ax+Bx2=Ax+B be the characteristic equation of the associated homogeneous recurrence relation and let x1x1 and x2x2 be its roots.

- If x≠x1x≠x1 and x≠x2x≠x2, then at=Axnat=Axn
- If x=x1x=x1, x≠x2x≠x2, then at=Anxnat=Anxn
- If x=x1=x2x=x1=x2, then at=An2xnat=An2xn

Example

Let a non-homogeneous recurrence relation be Fn=AFn–1+BFn−2+f(n)Fn=AFn–1+BFn−2+f(n) with characteristic roots x1=2x1=2 and x2=5x2=5. Trial solutions for different possible values of f(n)f(n) are as follows −

f(n)	Trial solutions
4	A
5.2^n	$An2^n$
8.5^n	$An5^n$
4^n	$A4^n$
$2n^2+3n+1$	An^2+Bn+C

Problem

Solve the recurrence relation Fn=3Fn−1+10Fn−2+7.5nFn=3Fn−1+10Fn−2+7.5n where F0=4F0=4 and F1=3F1=3

Solution

This is a linear non-homogeneous relation, where the associated homogeneous equation is Fn=3Fn−1+10Fn−2Fn=3Fn−1+10Fn−2 and f(n)=7.5nf(n)=7.5n

The characteristic equation of its associated homogeneous relation is −

x2−3x−10=0x2−3x−10=0

Or, (x−5)(x+2)=0(x−5)(x+2)=0
Or, x1=5x1=5 and x2=−2x2=−2

Hence ah=a.5n+b.(−2)nah=a.5n+b.(−2)n , where a and b are constants.
Since f(n)=7.5nf(n)=7.5n, i.e. of the form c.xnc.xn, a reasonable trial solution of at will be AnxnAnxn
at=Anxn=An5nat=Anxn=An5n

After putting the solution in the recurrence relation, we get −

An5n=3A(n–1)5n−1+10A(n–2)5n−2+7.5nAn5n=3A(n–1)5n−1+10A(n–2)5n−2+7.5n
Dividing both sides by 5n−25n−2, we get
An52=3A(n−1)5+10A(n−2)50+7.52An52=3A(n−1)5+10A(n−2)50+7.52
Or, 25An=15An−15A+10An−20A+17525An=15An−15A+10An−20A+175
Or, 35A=17535A=175
Or, A=5A=5
So, Fn=An5n=5n5n=n5n+1Fn=An5n=5n5n=n5n+1

The solution of the recurrence relation can be written as −

Fn=ah+atFn=ah+at
=a.5n+b.(−2)n+n5n+1=a.5n+b.(−2)n+n5n+1
Putting values of F0=4F0=4 and F1=3F1=3, in the above equation, we get a=−2a=−2 and b=6b=6

Hence, the solution is −

Fn=n5n+1+6.(−2)n−2.5nFn=n5n+1+6.(−2)n−2.5n
Generating Functions

Generating Functions represents sequences where each term of a sequence is expressed as a coefficient of a variable x in a formal power series.

Mathematically, for an infinite sequence, say a0,a1,a2,…,ak,…,a0,a1,a2,…,ak,…, the generating function will be −

Gx=a0+a1x+a2x2+⋯+akxk+⋯=∑k=0∞akxkGx=a0+a1x+a2x2+⋯+akxk+⋯=∑k=0∞akxk

Some Areas of Application

Generating functions can be used for the following purposes −

- For solving a variety of counting problems. For example, the number of ways to make change for a Rs. 100 note with the notes of denominations Rs.1, Rs.2, Rs.5, Rs.10, Rs.20 and Rs.50
- For solving recurrence relations
- For proving some of the combinatorial identities
- For finding asymptotic formulae for terms of sequences

Problem 1

What are the generating functions for the sequences {ak}{ak} with ak=2ak=2 and ak=3kak=3k?

Solution

When ak=2ak=2, generating function, G(x)=∑∞k=02xk=2+2x+2x2+2x3+…G(x)=∑k=0∞2xk=2+2x+2x2+2x3+…
When ak=3k,G(x)=∑∞k=03kxk=0+3x+6x2+9x3+……ak=3k,G(x)=∑k=0∞3kxk=0+3x+6x2+9x3+……

Problem 2

What is the generating function of the infinite series; 1,1,1,1,…1,1,1,1,…?

Solution

Here, ak=1ak=1, for 0≤k≤∞0≤k≤∞
Hence, G(x)=1+x+x2+x3+…⋯=1(1−x)G(x)=1+x+x2+x3+…⋯=1(1−x)

Some Useful Generating Functions

- For ak=ak,G(x)=∑∞k=0akxk=1+ax+a2x2+……⋯=1/(1−ax)ak=ak,G(x)=∑k=0∞akxk=1+ax+a2x2+……⋯ =1/(1−ax)
- For ak=(k+1),G(x)=∑∞k=0(k+1)xk=1+2x+3x2……⋯=1(1−x)2ak=(k+1),G(x)=∑k=0∞(k+1)xk=1+2x+3x 2……⋯=1(1−x)2
- For ak=cnk,G(x)=∑∞k=0cnkxk=1+cn1x+cn2x2+……⋯+x2=(1+x)nak=ckn,G(x)=∑k=0∞cknxk=1+c1nx+ c2nx2+……⋯+x2=(1+x)n
- For ak=1k!,G(x)=∑∞k=0xkk!=1+x+x22!+x33!……⋯=ex

Module Wise Important Questions

MODULE – I

1. Define DNF , CNF & Find CNF for ~(P ∨ Q) ↔ (P^Q)?
2. Define PCNF, PDNF & Find PCNF for (P^Q) ∨ (~P^Q)?
3. Define Tautology .Show that the formula Q ∨ (P^~Q) ∨ (~P^~Q) is a Tautology using truth table.
4. Explain briefly different connectives. Construct the truth table for (P ^~Q) V (~P ^~ Q)?
5. Explain different quantifiers with examples?
6. Write down the following statements in symbolic logic
   a) Some integers are divisible by 5
   b) All real numbers are complex numbers
   c) Every real number is rational or irrational but not both

MODULE –II

1. Show that the following premises are inconsistent. The premises are E → S, S → H, A →~H and E ^ A.
2. Explain the three rules of inference. Demonstrate that R is a valid inference from the premises P→ Q, Q → R, and P
3. Write about automatic theorem proving.
4. Define equivalence relation. Let S = {1,2,3,4} and R = {(1,1),(1,2),(2,1),(2,2), (3,4),(4,3),(3,3),(4,4)}. Find the given relation is equivalence or not.
5. Discuss about Hasse diagram. Let X={2,3,6,12,24,36}and the relation ≤ be such that x≤y if x divides y then draw the required Hasse diagrams
6. Explain different properties of a binary relation.

MODULE –III

1. Define different types of functions with examples.
2. Discuss about Inverse function and composition function with examples.
3. Define Recursive functions. Show that f(x,y)=x*y is a primitive recursive function?

4. Define Algebraic structure? On the set Q of all rational numbers, the operation * is defined by a*b= a + b - ab. show that under this operation forms a commutative monoid.
5. Explain about Group with example? Let G be the set of all non Zero real numbers and let a*b=1/2 ab. show that <G,*> is an Abelian group.
6. Define the below terms.
   i) Semi group ii) Monoid iii) Sub Group iv) Properties of Binary Operations

Module –IV

1. Define permutation and find the number of permutations of the letters of the word SUCCESS.
2. Define combination and a certain question paper contains two parts A and B each containing 4 questions. How many different ways a student can answer 5 questions by selecting atleast two questions from each part?
3. State the binominal and multi nominal theorems? Find the coefficient of xyz^5 in the expansion of $(x + y + z)^7$
4. Among the students in a hostel,12 students study Maths (A), 20 Physic (B),20 Chemistry (C),8 Biology (D), there are 5 students for A and B,7 are for A and C, 4 are A and D,16 are B and C, are B and D,3 students for A,C,D finally, there are 2 who study all of these subjects. Furthermore, there are 71 students who do not study any of these subjects. find the total number of students in hostel.
5. State the pigeonhole principle? prove that if 40 dictionaries in a library contain in a total87,428 pages, then at least one of the dictionaries must have at least 2186.
6. In How many ways can 20 similar books be placed on 5 different shelves.

Module –V

1. Solve the recurrence relation $a_{n+1}=4a_n$ for $n\geq 0$, given that $a_0=3$.
2. Solve the recurrence relation $a_n + a_{n-1}-6a_{n-2}=0$ for $n\geq 2$ given that $a_0 =-1$, a1=8.
3. Solve the recurrence relation $2a_{n+3} = a_{n+2}+2a_{n+1}-a_n$, for $n\geq 0$ given that $a_0 =1$, a1=1, $a_2=2$.
4. Solve the recurrence relation $a_n + 4a_{n-1}+4a_{n-2}=8$ for $n \geq 2$, and $a_0=1$, $a_1=2$.
5. Find a generating function for the recurrence relation $a_n + a_{n-1}- 6a_{n-2}=0$ for $n\geq 2$, given $a_0=-1, a_1=8$.
6. Find the generating function for the recurrence relation $a_{n+1} - a_n=3^n$ for $n \geq 0$, and $a_0=1$ hence the solve relations.