

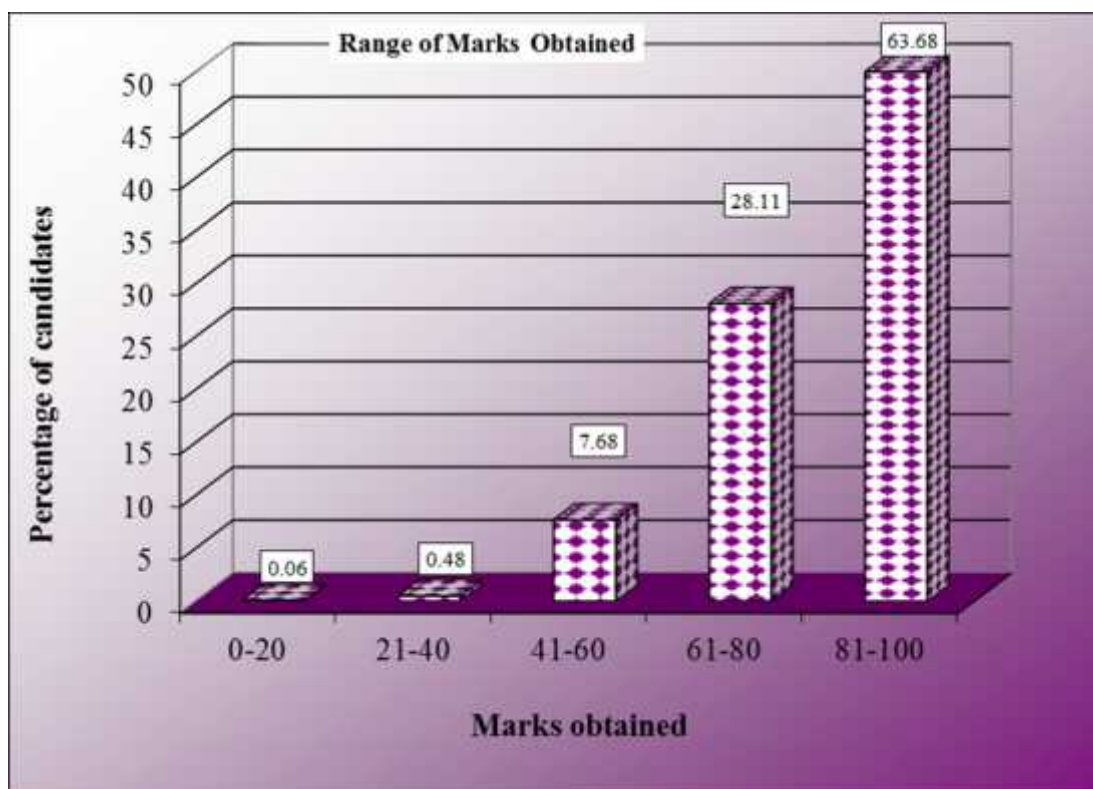
# COMPUTER SCIENCE

## A. STATISTICS AT A GLANCE

Total number of students taking the examination	18,018
Highest marks obtained	100
Lowest marks obtained	1
Mean marks obtained	83.68

### Percentage of candidates according to marks obtained

	Mark Range				
	<i>0-20</i>	<i>21-40</i>	<i>41-60</i>	<i>61-80</i>	<i>81-100</i>
Number of candidates	10	87	1383	5065	11473
Percentage of candidates	0.06	0.48	7.68	28.11	63.68
Cumulative Number	10	97	1480	6545	18018
Cumulative Percentage	0.06	0.54	8.21	36.32	100



## B. ANALYSIS OF PERFORMANCE

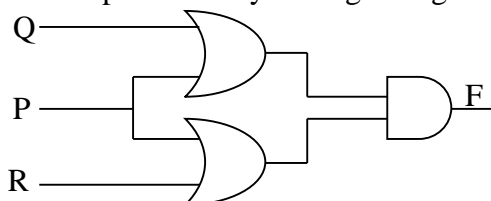
### PART I

Answer **all** questions.

**While answering questions in this Part, indicate briefly your working and reasoning, wherever required.**

#### Question 1

- (a) From the logic circuit diagram given below, find the output 'F' and simplify it. [2]  
Also, state the law represented by the logic diagram.



- (b) Write the truth table for a 2-input conjunction and disjunction in a proposition. [2]  
(c) Find the complement of  $XY'Z + XY + YZ'$  [2]  
(d) Convert the following expression into its canonical POS form : [2]

$$F(A,B) = (A + B) \cdot A'$$

- (e) Minimise the following Boolean expression using the Karnaugh map: [2]

$$F(A,B,C) = \bar{A}\bar{B}\bar{C} + \bar{A}BC + A\bar{B}\bar{C} + ABC$$

#### Comments of Examiners

- (a) This part was well answered by most of the candidates. Some candidates did not mention the law, but wrote the expression correctly. Some wrote only the intermediate expression.  
(b) Some candidates were confused with the terms 'conjunction' and 'disjunction' and interchanged their truth tables.  
(c) Most of the candidates answered this part correctly. A few simplified the expression, without finding the complement. Some only reduced the expression.  
(d) Some candidates were not clear with the POS form of expression and wrote the SOP expression. Actual meaning and features of canonical expression were not clear to many candidates.  
(e) Almost all candidates answered this part correctly except for a few who did not use the k-map for reducing but used the laws instead. K-Map plotting was not correct in some cases.

#### Suggestions for teachers

- Practice should be given to candidates in deriving an expression from a logic circuit and also recognizing the law used. Step by step solving of a circuit should be practiced.
- Proposition logic should be taught using all the terms that are required, especially the terms conjunction, disjunction, negation, implications and bi conditional.
- More practice should be given to students to complement a Boolean expression using de Morgan's law. Each and every law must be practiced with examples. All related terms and symbols used for each gate should be explained.

- More practice should be given in solving SOP and POS type of Boolean expressions. Both canonical and cardinal forms should be practiced.
- More practice should be given in minimizing Boolean expressions using k-maps. 3 variable k-maps should also be practiced for short answers.

## MARKING SCHEME

### Question 1.

(a)  $(P+Q)(P+R) = P+Q.R$  Distributive Law

(b)

A	B	Conjunction
0	0	0
0	1	0
1	0	0
1	1	1

A	B	Disjunction
0	0	0
0	1	1
1	0	1
1	1	1

(c)  $(XY'Z + XY + YZ')'$

$$(X' + Y + Z') \cdot (X' + Y') \cdot (Y' + Z)$$

**OR**  $(X' + Y + Z') \cdot (Y' + X'Z)$

(d)  $(A+B)A'$

$$= (A+B)(A' + B.B')$$

$$= (A+B)(A' + B)(A' + B')$$

(e)

	B'C'	B'C	BC	BC'
A'	0 0	1 0	3 1	2 1
A	4 0	5 0	7 1	6 1

$$\text{Quad} (m_2 + m_3 + m_6 + m_7) = B$$

## Question 2

- (a) State *two* advantages of using the *concept of inheritance* in Java. [2]
- (b) An array AR [ -4 .... 6, -2....12 ] , stores elements in **Row Major Wise**, with the address AR[2][3] as 4142. If each element requires 2 bytes of storage, find the Base address. [2]
- (c) State the sequence of traversing a binary tree in: [2]
- (i) preorder
- (ii) postorder
- (d) Convert the following infix expression into its postfix form: [2]
- $$( A / B + C ) * ( D / ( E - F ) )$$
- (e) State the difference between the functions *int nextInt( )* and *boolean hasNextInt( )*. [2]

### Comments of Examiners

- (a) A number of candidates were not aware of the advantages of inheritance. Some used examples while others explained in their own words. A few candidates wrote the definition instead of the advantages.
- (b) Candidates who had the knowledge of the formula to find the address of a cell in an array could solve and get the correct answer. Others gave vague answers. Some calculated using memory cell diagram.
- (c) In parts (i) and (ii), some candidates explained with the help of a diagram. A few used the term vertex or an operator to represent root.
- (d) Some candidates wrote the correct answer without showing the working. Others applied the postfix correctly, but could not derive the final answer.
- (e) Several candidates wrote the functions with syntax while others wrote in their own language.

### Suggestions for teachers

- OOP's concepts must be explained with advantages and examples. Real world examples may be used to explain these terms.
- Practice should be given in understanding the formula using row major and column major and making the subject of formula that is required in the question.
- Students should be given more practice in binary trees and various terms used.
- Examples need to be practiced with conversion of Infix to Postfix notation, the order of precedence; the Polish stack method should be explained clearly.

## MARKING SCHEME

### Question 2.

- (a) (i) Reusability (ii) Transitive nature.
- (b) Address of  $AR[i,j] = B + W [ Nc ( i - Lr ) + ( j - Lc ) ]$   
Address of  $AR[i,j] = 4142$   
 $B = ?$   
 $W = 2$   
 $Nc = Uc - Lc + 1 = 12 - (-2) + 1 = 15$   
 $i = 2 \quad j = 3$   
 $Lr = -4 \quad Lc = -2$   
 $4142 = B + 2 [ 15 ( 2 - (-4) ) + ( 3 - (-2) ) ]$   
 $4142 = B + 2 [ 90 + 5 ]$   
 $B = 4142 - 190 = 3952$
- (c) (i) root – left – right (+LR) (ii) left – right – root (LR+)
- (d)  $( A / B + C ) * ( D / ( E - F ) )$   
 $= ( AB / +C ) * ( D / ( EF - ) )$   
 $= ( AB / C + ) * ( DEF - / )$   
 $= AB / C + DEF - / *$
- (e) `nextInt()` Returns next token as an integer  
`hasNextInt ( )` Returns true if the input has another token and that token can be interpreted as an integer.

### Question 3

- (a) The following functions are part of some class:

```
void fun1(char s[ ],int x)
{
    System.out.println(s);
    char temp;
    if(x<s.length/2)
    {
        temp=s[x];
        s[x]=s[s.length-x-1];
        s[s.length-x-1]=temp;
        fun1(s,x+1);
    }
}
```

```

void fun2(String n)
{
    char c[ ]=new char[n.length( )];
    for(int i=0;i<c.length; i++)
        c[i]=n.charAt(i);
    fun1(c,0);
}

```

- (i) What will be the output of **fun1( )** when the value of s[ ]={ 'J','U','N','E' } and x=1? [2]
- (ii) What will be the output of **fun2( )** when the value of n ="SCROLL" ? [2]
- (iii) State in one line what does the function **fun1( )** do apart from recursion. [1]

- (b) The following is a function of some class which sorts an integer array a[ ] in ascending order using *selection sort* technique. There are some places in the code marked by **?1?**, **?2?**, **?3?**, **?4?**, **?5?** which may be replaced by a statement/expression so that the function works properly:

```

void selectsort(int [ ]a)
{
    int i,j,t,min,minpos;
    for( i=0;i<?1?;i++)
    {
        min=a[i];
        minpos = i;
        for(j=?2?;j<a.length;j++)
        {
            if(min>a[j])
            {
                ?3? = j;
                min = ?4?;
            }
        }
        t=a[minpos];
        a[minpos]=a[i];
        a[i]= ?5?;
    }

    for(int k=0;k<a.length;k++)
        System.out.println(a[k]);
}

```

- |       |  |     |
|-------|--|-----|
| (i)   | What is the expression or statement at ?1? | [1] |
| (ii)  | What is the expression or statement at ?2? | [1] |
| (iii) | What is the expression or statement at ?3? | [1] |
| (iv)  | What is the expression or statement at ?4? | [1] |
| (v)   | What is the expression or statement at ?5? | [1] |

### Comments of Examiners

- (a) Only some candidates gave the correct answer. The concept of nested functions was not clear to many candidates. Some candidates were not clear regarding the concept of recursive technique. Others did not show the working / dry run. Some were confused with the printing of the character array 's' as it did not have any subscript. A few wrote 'error' while many mentioned 'address at zero index' as output. In some cases vertical output was given instead of horizontal. Several candidates printed only the first and the last character.
- (b) Most of the candidates answered well. Multiple correct options were available and most of the candidates answered correctly. Some were confused with length and length ( ).

### Suggestions for teachers

- Various techniques relating to problem solving using recursion should be explained to students. Output programs must be explained using diagrams for each function call. More practice should be given in character array printing. Only character array prints the whole array, whereas the other type of arrays print the address at zero index if subscript is not mentioned.
- Practice should be given in program using conditions / looping and other output related programs. Show the dry run/ working of program and emphasize that working is necessary to get full credit. Practice must be given in standard programs.

### **MARKING SCHEME**

#### **Question 3.**

- |     |     |      |      |        |       |                                 |
|-----|-----|------|------|--------|-------|---------------------------------|
| (a) | (i) | JUNE | (ii) | SCROLL | (iii) | It reverses the character array |
|     |     | JNUE |      | LCROLS |       | Swaps, Interchanges etc.        |
|     |     |      |      | LLROCS |       |                                 |
|     |     |      |      | LLORCS |       |                                 |

(b)	(i)	a.length OR a.length-1
	(ii)	i+1 Or i OR minpos + 1 OR minpos
	(iii)	minpos
	(iv)	a[j]
	(v)	t

## PART – II

Answer **seven** questions in this part, choosing **three** questions from Section A, **two** from Section B and **two** from Section C.

### SECTION - A

Answer any **three** questions.

#### Question 4

- (a) Given the Boolean function  $F(A, B, C, D) = \sum (0, 1, 2, 3, 5, 6, 7, 10, 13, 14, 15)$
- Reduce the above expression by using 4-variable Karnaugh map, showing the various groups (i.e. octal, quads and pairs). [4]
  - Draw the logic gate diagram for the reduced expression. Assume that the variables and their complements are available as inputs. [1]
- (b) Given the Boolean function  $P(A, B, C, D) = \prod (0, 1, 2, 3, 5, 6, 7, 10, 13, 14, 15)$
- Reduce the above expression by using 4-variable Karnaugh map, showing the various groups (i.e. octal, quads and pairs). [4]
  - Draw the logic gate diagram for the reduced expression. Assume that the variables and their complements are available as inputs. [1]

#### Comments of Examiners

- (a) (i) Several candidates made errors in place value and putting variables in K-Map. In some cases, the groups were reduced by laws. A number of candidates drew the K-Map incorrectly. Many candidates included the redundant group in the final expression.
- (ii) Some candidates drew the logic circuit using universal gates while some others drew vague diagrams.
- (b) (i) A number of candidates were not able to draw the K-Map for the POS expression correctly. For many candidates the “Map rolling” concept was not very clear. Some converted the canonical form to cardinal form and then reduced it.

#### Suggestions for teachers

- Emphasize on arranging the variables in proper order and the importance of cell values corresponding with the variables. Explain clearly how the groups are framed and reduced. Redundant groups are not to be included in the final reduced expression.
- More practice should be given in drawing logic circuits using basic gates and also with universal gates.



- (ii) Some candidates drew the logic circuit using universal gates while some others drew vague diagrams.

– Make students reduce POS and SOP expressions using K-Map simultaneously. Students should be told not to include the redundant group in the final expression.

## MARKING SCHEME

### Question 4.

- (a) (i)  $F(A,B,C,D) = \sum (0, 1, 2, 3, 5, 6, 7, 10, 13, 14, 15)$

	$C'D'$	$C'D$	$CD$	$CD'$
$A'B'$	0 1	1 1	3 1	2 1
$A'B$	4 0	5 1	7 1	6 1
$AB$	12 0	13 1	15 1	14 1
$AB'$	8 0	9 0	11 0	10 1

There are three quads :

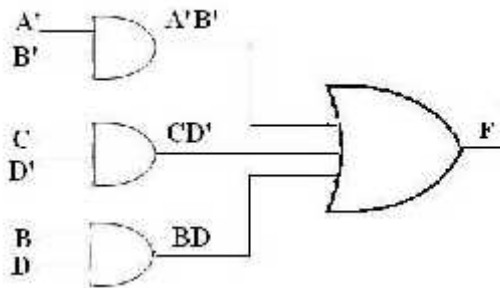
$$\text{Quad 1 } (m_0 + m_1 + m_2 + m_3) = A'B'$$

$$\text{Quad 2 } (m_2 + m_6 + m_{10} + m_{14}) = CD'$$

$$\text{Quad 3 } (m_5 + m_7 + m_{13} + m_{15}) = BD$$

$$\text{Hence } F(A, B, C, D) = A'B' + CD' + BD$$

- (ii)



(b) (i)  $P(A,B,C,D) = (0, 1, 2, 3, 5, 6, 7, 10, 13, 14, 15)$

	$C+D$	$C+D'$	$C'+D'$	$C'+D$
$A+B$	0 0	1 0	3 0	2 0
$A+B'$	4 1	5 0	7 0	6 0
$A'+B'$	12 1	13 0	15 0	14 0
$A'+B$	8 1	9 1	11 1	10 0

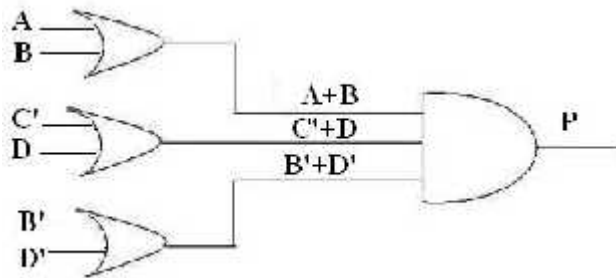
There are three quads :

Quad 1 :  $(M_0 M_1 M_2 M_3) = A+B$       Quad 2 :  $(M_2 M_6 M_{10} M_{14}) = C' + D$

Quad 2 :  $(M_5 M_7 M_{13} M_{15}) = B' + D'$

Hence  $P(A,B,C,D) = (A+B) \cdot (C' + D) \cdot (B' + D')$

(ii)



### Question 5

A school intends to select candidates for the Inter-School Athletic Meet, as per the criteria given below:

- The candidate is from the Senior School and has participated in an Inter-School Athletic Meet earlier.

**OR**

- The candidate is not from the Senior School, but the height is between 5 ft. and 6 ft. and weight is between 50 kg. and 60 kg.

**OR**

- The candidate is from the senior school and has height between 5 ft. and 6 ft., but the weight is not between 50 kg. and 60 kg.

The inputs are:

INPUTS	
<b>S</b>	Student is from the Senior School
<b>W</b>	Weight is between 50 kg. and 60 kg.
<b>H</b>	Height is between 5 ft. and 6 ft.
<b>A</b>	Taken part in Inter-School Athletic Meet earlier

( In all of the above cases 1 indicates yes and 0 indicates no)

Output: **X** - Denotes the selection criteria [1 indicates selected and 0 indicates rejected in all cases.]

(a) Draw the truth table for the inputs and outputs given above and write the **SOP** expression for  $X(S, W, H, A)$ . [5]

(b) Reduce  $X(S, W, H, A)$  using Karnaugh map. [5]

Draw the logic gate diagram for the reduced **SOP** expression for  $X(S, W, H, A)$  using AND and OR gate. You may use gates with two or more inputs. Assume that the variable and their complements are available as inputs.

#### Comments of Examiners

- (a) Most of the candidates answered this part well. Some candidates did not mention the final expression. A few were confused with the SOP expression and took the output with 0's instead of 1's. Some took 0's as outputs but wrote the minterms instead of maxterms. Careless mistakes were noticed in several cases.
- (b) Some candidates were not able to draw the K-Map for the SOP expression correctly. For a number of candidates, the "Map rolling" concept was not very clear. Some converted the canonical form to cardinal form and then reduced it. A few candidates replaced the variable names.

#### Suggestions for teachers

- Students should be told to read the question carefully and answer accordingly so that no part is left unanswered.
- More practice should be given in deriving SOP and POS expression from any given truth table (i.e. Minterms and Maxterms).
- Make students reduce POS and SOP expressions using K-Map simultaneously. Students should be told not to include the redundant group in the final expression. More practice must be given in marking the groups.

## MARKING SCHEME

### Question 5.

(a)

S	W	H	A	X (OUTPUT)
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

SOP Expression:  $X(S, W, H, A) = \Sigma(6, 7, 9, 10, 11, 13, 15)$

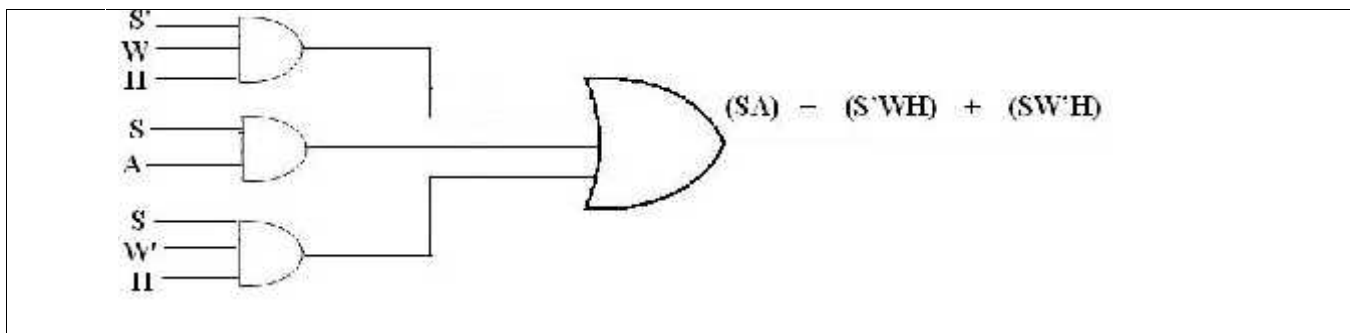
(b)

	H'A'	H'A	HA	HA'
S'W'	0 0	1 0	3 0	2 0
S'W	4 0	5 0	7 1	6 1
SW	12 0	13 1	15 1	14 0
SW'	8 0	9 1	11 1	10 1

Quad ( $m_9 + m_{11} + m_{13} + m_{15}$ ) = SA ; Pair 1 ( $m_6 + m_7$ ) = S'WH

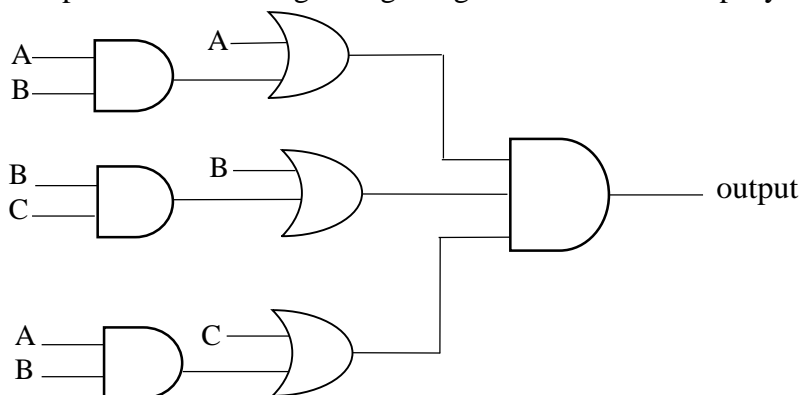
Pair 2 ( $m_{10} + m_{11}$ ) = SW'H

$X(S, W, H, A) = SA + S'WH + SW'H$



### Question 6

- (a) With the help of a logic diagram and a truth table explain a *Decimal to Binary encoder*. [4]
- (b) Derive a Boolean expression for the logic diagram given below and simplify it. [3]



- (c) Reduce the following expression using Boolean laws: [3]

$$F(A, B, C, D) = (A' + C) (A' + C') (A' + B + C'D)$$

### Comments of Examiners

- (a) Some candidates drew the AND gate instead of OR gate. Some drew the octal / hexadecimal encoder instead of decimal encoder. However, most of the candidates answered this part correctly. The working / explanation was shown in some cases which was not required.
- (b) Most of the candidates answered this part well except for a few who were not able to simplify the expression.
- (c) Most of the candidates attempted this part correctly barring a few who gave vague reduction.

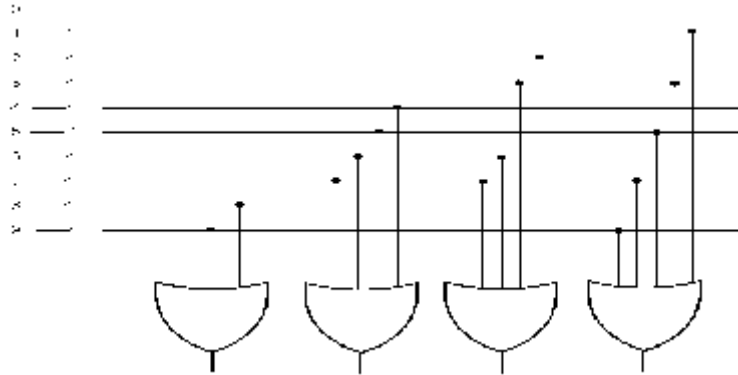
#### Suggestions for teachers

- Practice should be given in drawing the encoders along with the expression, truth table and logic circuit / diagram.
- More practice should be given in deriving expressions from a circuit diagram and simplifying it. Laws must be implemented for minimizing expressions.
- All related laws of Boolean algebra should be practiced for reducing expressions.

## MARKING SCHEME

### Question 6.

(a)



Truth table for decimal to binary encoder :

Decimal number	A	B	C	D
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

(b) Output:  $(A + AB) (B + BC) (C + AB)$

$$(A) \cdot (B) (C + AB)$$

$$ABC + ABAB$$

$$ABC + AB$$

$$AB (C + 1) = AB$$

(c)  $(A' + C) (A' + C') (A' + B + C'D)$

$$A' (1 + C) (1 + C') (A' + B + C'D)$$

$$A' (A' + B + C'D)$$

$$A' A' + A' B + A' C'D$$

$$A' + A' B + A' C'D$$

$$A (1 + 1B + 1C'D)$$

$$A' (1 + B + C'D)$$

$$A' (1)$$

$$= A'$$

### Question 7

(a) Differentiate between XNOR and XOR gates. Draw the truth table and logic diagrams of 3 input XNOR gate. [4]

(b) Differentiate between a *proposition* and *wff*. [2]

(c) Define *Half Adder*. Construct the truth table and a logic diagram of a Half Adder. [4]

#### Comments of Examiners

- (a) Some candidates were confused between XNOR and XOR gates and interchanged the answers. Some were confused with the 3 input XNOR truth table.
- (b) A number of candidates gave the full form of *wff*. Some candidates wrote the same definition for both.
- (c) Several candidates drew the block diagram instead of the logic circuit diagram. In some cases, the definition was not clear and vague answers were given.

#### Suggestions for teachers

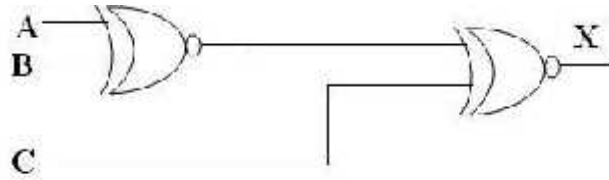
- More practice should be given for understanding the various gates (AND, OR, NOT, XOR, XNOR, NAND and NOR). Students should be told that in odd number of inputs, XOR and XNOR outputs are same and in even number of inputs XOR is the complement of XNOR.
- More practice should be given in the chapter on Propositional logic and the various terms used must be explained with examples.
- Application of logic gates should be explained along with definition, truth table, logic diagram and expressions.

### MARKING SCHEME

#### Question 7.

(a)	<b>XNOR</b>	<b>XOR</b>
	1. Logically equivalent to an inverted XOR, i.e., XOR gate followed by a NOT gate	1. Logically equivalent to an exclusive OR. i.e. if both inputs are 1 then output is 0 (reverse of inclusive OR)
	2. Produces 1 (high) as output when the input combination has even number of 0s	2. Produces 1 (high) as output when the inputs combination has odd number of 1s.

Logic Diagram of a 3 input XNOR gate :



Truth table for 3 input XNOR gate :

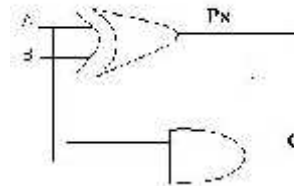
A	B	C	X
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Expression :  $X = A \odot B \odot C$

- (b) Proposition: A statement that can have only two possible answer (True or false)  
wff: It is a proposition which is satisfiable or valid.
- (c) Half adders are combinational circuits which adds two input binary bits and output two binary bits.

Truth table of half adder :

A	B	Partial Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



Partial sum (Ps) =  $(A+B) \cdot (A'+B') = A \oplus B$

Carry (C) =  $(A+B) \cdot (A+B') \cdot (A'+B) = AB$



## SECTION – B

*Answer any **two** questions.*

*Each program should be written in such a way that it clearly depicts the logic of the problem.*

*This can be achieved by using mnemonic names and comments in the program.*

(Flowcharts and Algorithms are **not** required.)

**The programs must be written in Java.**

### Question 8

A class **Mixer** has been defined to merge two sorted integer arrays in ascending order. [10]  
Some of the members of the class are given below:

**Class name** : **Mixer**

#### **Data members/instance variables:**

int arr[ ] : to store the elements of an array

int n : to store the size of the array

#### **Member functions:**

Mixer(int nn) : constructor to assign n=nn

void accept( ) : to accept the elements of the array in ascending order without any duplicates

Mixer mix(Mixer A) : to merge the current object array elements with the parameterized array elements and return the resultant object

void display( ) : to display the elements of the array

Specify the class **Mixer**, giving details of the **constructor(int)**, **void accept( )**, **Mixer mix(Mixer)** and **void display( )**. Define the **main( )** function to create an object and call the function accordingly to enable the task.

#### Comments of Examiners

Common errors made by candidates in answering this question were as follows: (i) wrong declaration of array; (ii) significance of data members were not clear; (iii) passing objects to the function and accessing its members by the dot operator; (iv) creating a new local object and returning the object. Some candidates sorted the array in accept ( ).

In some cases, 3 different arrays were created instead of creating objects. A few candidates used the same object to store the answer without creating a local object. Some were confused with merging or mixing the two arrays.

#### Suggestions for teachers

- Knowledge of handling/ passing objects to functions must be explained in detail. It must be emphasized that only the mentioned technique should be written. More practice should be given in writing the main function with each and every program. Practice should be given in both single and double dimension programs.

## MARKING SCHEME

### Question 8.

```
import java.util.*;
class Mixer
{ int arr[ ];
  int n;
  static Scanner sc=new Scanner(System.in);
  Mixer(int nn)
  {   n=nn;
      arr=new int[n];
  }
  void accept( )
  {   System.out.println("Enter  "+ n+ " elements in ascending order");
      for(int i=0;i<n;i++)
          arr[i]=sc.nextInt();
  }
  Mixer mix( Mixer A)
  {   Mixer B=new Mixer(n+A.n);
      int x=0,y=0,z=0;
      while(x<n && y<A.n)
      {   if (arr[x] > A.arr[y])
          {   B.arr[z]=A.arr[y];
              y++;
          }
          else
          {   B.arr[y]=arr[x];
              x++;
          }
          z++;
      }
      while(x<n)
          B.arr[z++]=arr[x++];
      while(y<A.n)
          B.arr[z++]=A.arr[y++];
      return B;
  }
  void display()
  {   for(int i=0;i<n;i++)
      System.out.println(arr[i]);
  }
  static void main()
  {   Mixer P=new Mixer(4);
      Mixer Q=new Mixer(3);
      Mixer R=new Mixer(7);
      P.accept();
      Q.accept();
```

```
OR      Mixer mix ( Mixer A)
        {   Mixer B=new Mixer (n+A.n);
            int x=0;
            for(int i=0 ;i<n ;i++)
                B.arr[x++]=arr[i];
            for(int j =0 ;j<A.n ;j++)
                B.arr[x++]=A.arr[j] ;
            return B ;
        }
```

```

    R=P.mix(Q);
    R.display();
}
}

```

### Question 9

A class **SeriesSum** is designed to calculate the sum of the following series:

$$\text{Sum} = \frac{x^2}{1!} + \frac{x^4}{3!} + \frac{x^6}{5!} + \dots + \frac{x^n}{(n-1)!}$$

Some of the members of the class are given below:

**Class name** : **SeriesSum**

**Data members/instance variables:**

x	: to store an integer number
n	: to store number of terms
sum	: double variable to store the sum of the series

**Member functions:**

SeriesSum(int xx, int nn)	: constructor to assign x=xx and n=nn
double findfact(int m)	to return the factorial of m using <b>recursive technique</b> .
double findpower(int x, int y)	: to return x raised to the power of y using <b>recursive technique</b> .
void calculate( )	: to calculate the sum of the series by invoking the recursive functions respectively
void display( )	: to display the sum of the series

- (a) Specify the class **SeriesSum**, giving details of the **constructor(int, int)**, **double findfact(int)**, **double findpower(int, int)**, **void calculate( )** and **void display( )**. Define the **main( )** function to create an object and call the functions accordingly to enable the task. [8]
- (b) State the *two* differences between *iteration* and *recursion*. [2]

### Comments of Examiners

- (a) A number of candidates answered this part correctly. Common errors made were: the concept of recursion was not clear to some candidates; some used the inbuilt mathematical functions; some wrote the function findfact() and findpower() without using the recursive technique. Several candidates did not invoke the functions findfact() and findpower() to calculate the sum of the series. In some cases, 'return' statement was missing. The other function including the constructor was well answered. Some candidates did not write the main() function.
- (b) In some cases, only one difference was given by candidates.

### Suggestions for teachers

- More practice should be given in solving programs using recursive techniques. Knowledge of base case and recursive case should be given to the students for every program using recursive technique. Practice should be given in invoking function within another function.
- More examples must be given to explain the difference between iteration and recursion using memory blocks and time factor.

### **MARKING SCHEME**

#### **Question 9.**

```
(a) class SeriesSum
{
    int x,n;
    double sum;
    SeriesSum(int xx,int nn)
    {   x=xx;
        n=nn;
        sum=0.0;
    }
    double findfact(int a)
    {   return (a<2)? 1:a*findfact(a-1);
    }
    double findpower(int a, int b)
    {   return (b==0)? 1:a*findpower(a,b-1);
    }
    void calculate()
    {   for(int i=2;i<=n;i+=2)
        sum += findpower(x,i)/findfact(i-1);
    }
    void display()
    {   System.out.println("sum="+ sum);
    }
    static void main()
    {   SeriesSum obj = new SeriesSum(3,8);
        obj.calculate();
        obj.display();
    }
}
```

```
}  
}
```

- (b) Iteration: Fast process and less memory.  
Recursion: Slow process and more memory.

### Question 10

A sequence of fibonacci strings is generated as follows:

[10]

$S_0 = \text{"a"}, S_1 = \text{"b"}, S_n = S_{(n-1)} + S_{(n-2)}$  where '+' denotes concatenation. Thus the sequence is:  
a, b, ba, bab, babba, babbabab, ..... n terms.

Design a class **FiboString** to generate fibonacci strings. Some of the members of the class are given below:

**Class name** : **FiboString**

**Data members/instance variables:**

x : to store the first string  
y : to store the second string  
z : to store the concatenation of the previous two strings  
n : to store the number of terms

**Member functions/methods:**

FiboString( ) : constructor to assign x="a", y="b" and z="ba"  
void accept( ) : to accept the number of terms 'n'  
void generate( ) : to generate and print the fibonacci strings. The sum of ('+' ie concatenation) first two strings is the third string. Eg. "a" is first string, "b" is second string then the third will be "ba", and fourth will be "bab" and so on.

Specify the class **FiboString**, giving details of the **constructor( )**, **void accept( )** and **void generate( )**. Define the **main( )** function to create an object and call the functions accordingly to enable the task.

### Comments of Examiners

Different methods / logic were used to concatenate the words and swap them. The loop was not proper in some of the cases along with its swapping. Interchanging and concatenating was not done properly in many cases. Several candidates did the entire program without answering the generate() function. The main() function and constructor were not answered by some candidates.

#### Suggestion for teachers

- Different methods /logic should be adopted so that wider exposure to string manipulation related programs is given to students.
- Knowledge of constructors to initialize a string and other data members should be given.

### **MARKING SCHEME**

#### **Question 10.**

```
import java.util.*;
class FiboString
{   String x,y,z;
    int n;
    FiboString( )
    {
        x="a";
        y="b";
        z="ba";
    }
    void accept ()
    {
        Scanner Sc = new Scanner (System.in);
        System.out.println ("Enter number of terms") ;
        n = Sc.nextInt( );
    }
    void generate()
    {   System.out.print(x+", "+y);
        for(int i=0;i<=n-2;i++)
        {   System.out.print(", "+z);
            x=y;
            y=z;
            z=y+x;   OR  z= y.concat(x);
        }
    }
    static void main( )
    {   FiboString obj=new FiboString( );
        obj.accept( );
        obj.generate();
    }
}
```

## SECTION – C

Answer any *two* questions.

*Each program should be written in such a way that it clearly depicts the logic of the problem stepwise.*

*This can be achieved by using comments in the program and mnemonic names or pseudo codes for algorithms. The programs must be written in Java and the algorithms must be written in general / standard form, wherever required / specified.*

(Flowcharts are **not** required.)

### Question 11

A super class **Stock** has been defined to store the details of the stock of a retail store. Define a subclass **Purchase** to store the details of the items purchased with the new rate and updates the stock. Some of the members of the classes are given below: [10]

**Class name** : **Stock**

#### **Data members/instance variables:**

item	: to store the name of the item
qty	: to store the quantity of an item in stock
rate	: to store the unit price of an item
amt	: to store the net value of the item in stock

#### **Member functions:**

Stock (...)	: parameterized constructor to assign values to the data members
void display( )	: to display the stock details

**Class name** : **Purchase**

#### **Data members/instance variables:**

pqty	: to store the purchased quantity
prate	: to store the unit price of the purchased item

#### **Member functions / methods**

Purchase(...)	: parameterized constructor to assign values to the data members of both classes
void update ( )	: to update stock by adding the previous quantity by the purchased quantity and replace the rate of the item if there is a difference in the purchase rate. Also update the current stock value as: (quantity * unit price)
void display( )	to display the stock details before and after updation

Specify the class **Stock**, giving details of the **constructor()** and **void display()**. Using **concept of inheritance**, specify the class **Purchase**, giving details of the **constructor()**, **void update()** and **void display()**.

**The main function and algorithm need not be written.**

### Comments of Examiners

The concept of Inheritance was not clear to a number of candidates. In several cases, the amount was not calculated, but passed as parameter/argument. Constructor with inheritance was not answered correctly by some candidates. Accessing the members of the super class by the derived class was not clear to many candidates. Keywords 'extends' and 'Super' were found to be missing in some cases. A number of candidates declared the base class data members as private. Double data members were not declared properly by some students. Invoking the update () function in the display() function of the derived class was not answered properly. In a few cases, the algorithm was written instead of a program.

### Suggestions for teachers

- Instruct students to read the question properly and answer accordingly.
- Use of constructor using the base class member should be made clear.
- The different visibility modes and their accessing capability should be explained. Calling the member function from the super class to the derived class must be explained clearly.
- More practicals based on inheritance, involving functions from super class must be done.

### **MARKING SCHEME**

#### **Question 11.**

```
class Stock
{
    String item;
    double qty,rate,amt;
    Stock(String a,double b,double c)
    {
        item=a;
        qty=b;
        rate=c;
        amt=qty * rate;
    }
    void display()
    {
        System.out.println("Name of the item : "+item);
        System.out.println("Quantity: "+qty);
        System.out.println("Rate per unit: "+rate);
        System.out.println("Net value: "+amt);
    }
}
class Purchase extends Stock
{
    int pqty;
    double prate;
    Purchase(String a, double b, double c, int d, double e)
    {
        super(a,b,c);
        pqty=d;
        prate=e;
    }
}
```



```

    }
    void update()
    {   qty += pqty;
        if(prate!=rate)
            rate=prate;
        amt = qty * rate;
    }
    void display()
    {   super.display();
        update();
        super.display();
    }
}

```

### Question 12

A stack is a linear data structure which enables the user to add and remove integers from one end only, using the concept of LIFO(Last In First Out). An array containing the marks of 50 students in ascending order is to be pushed into the stack. [10]

Define a class **Array\_to\_Stack** with the following details:

**Class name** : **Array\_to\_Stack**

**Data members/instance variables:**

m[ ]	:	to store the marks
st[ ]	:	to store the stack elements
cap	:	maximum capacity of the array and stack
top	:	to point the index of the topmost element of the stack

**Methods/Member functions:**

Array_to_Stack (int n)	:	parameterized constructor to initialize cap = n and top = -1
void input_marks( )	:	to input the marks from user and store it in the array m[ ] in ascending order and simultaneously push the marks into the stack st[ ] by invoking the function pushmarks( )
void pushmarks(int v)	:	to push the marks into the stack at top location if possible, otherwise, display “not possible”
int popmarks( )	:	to return marks from the stack if possible, otherwise, return -999
void display( )	:	To display the stack elements

Specify the class **Array\_to\_Stack**, giving the details of the **constructor(int)**, **void input\_marks( )**, **void pushmarks(int)**, **int popmarks( )** and **void display( )**.

**The main function and the algorithm need not be written.**

#### Comments of Examiners

The concept of stack was not clear to most of the candidates. Common errors made by candidates were: (i) incorrect declaration of array; (ii) the condition / logic for underflow and overflow was not answered correctly; (iii) increment / decrement of top index was not done properly; Top was initialized to 0 instead of -1 in some cases; The pushing of marks from array to stack was confusing to many candidates; Some sorted the array and then pushed the values into the stack whereas in other cases it was directly done; The methods pushmarks() and popmarks() were found to be difficult by some candidates. The class declaration and constructors were well answered.

#### Suggestions for teachers

- More practice should be given in data structure programs like the stacks, queues, dequeues, etc. Working as to how the stack or a queue performs must be shown.
- The concept of LIFO and FIFO must be explained to students with live examples. The concept of ‘underflow’ and ‘overflow’ must be taught. The importance of ‘top’ subscript must be explained. Implementation of stacks, queues and dequeue using arrays should be emphasized. Only the concept has to be explained taking the base as an array. It should be made clear to students that it is not an array related program which can be manipulated by shifting / inserting or initializing by any value since these data structures require pointers and pointers are not supported in java. So, the array is used to show the working of a stack, queue or a dequeue.

#### **MARKING SCHEME**

##### **Question 12.**

```
import java.util.*;
class Array_to_Stack
{   int m[], st[];
    int cap, top;
    static Scanner sc=new Scanner(System.in);
    Array_to_Stack(int n)
    {   cap = n;
        top = -1;
        m=new int[cap];
```

```

        st=new int[cap];
    }
    void input_marks( )
    {   System.out.println("Enter "+cap+" elements in ascending order");
        for(int i=0;i<cap;i++)
        {   m[i]=sc.nextInt();
            pushmarks(m[i]);
        }
    }
    void pushmarks(int v)
    {   if (top<cap-1)
        st[++top]=v;
        else
            System.out.println("stack is full");
    }
    int popmarks()
    {   if(top>=0)
        return st[top--];
        else
            return -999;
    }
    void display()
    {   for(int i=top;i>=0;i--)
        System.out.println(st[i]);
    }
}

```

### Question 13

- (a) A linked list is formed from the objects of the class:

[4]

```

class Node
{
    int    number;
    Node  nextNode;
}

```

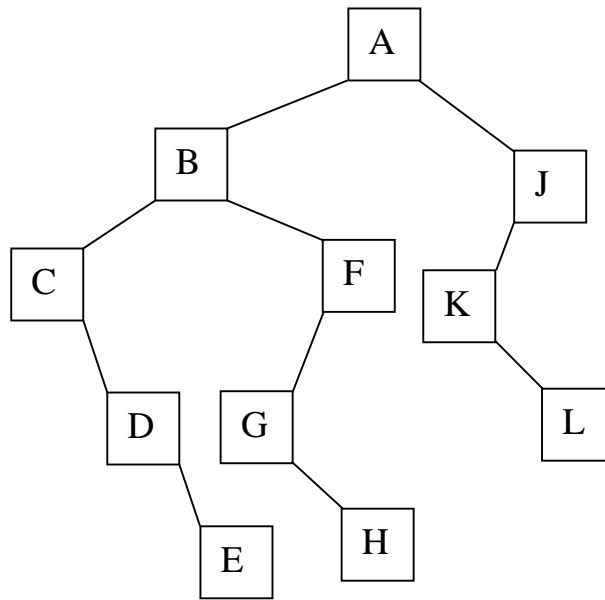
Write an *Algorithm* **OR** a *Method* to add a node at the end of an existing linked list.  
The method declaration is as follows:

**void addnode ( Node start, int num )**

- (b) Define the terms *complexity* and *big 'O' notation*.

[2]

(c) Answer the following from the diagram of the Binary Tree given below:



- |                                     |     |
|-------------------------------------|-----|
| (i) Root of the tree.               | [1] |
| (ii) Left subtree                   | [1] |
| (iii) Inorder traversal of the tree | [1] |
| (iv) Size of the tree.              | [1] |

#### Comments of Examiners

- (a) This part was well answered by many candidates. Some candidates had problems in moving the pointer to the next node and checking for null. Some wrote the algorithm in simple English language covering all the main steps. A few candidates were not able to link the new code to the end. Incomplete steps were found in some cases.
- (b) Some candidates wrote the same definition for both the terms.
- (c) (i) This part was answered correctly by almost all the candidates.
- (ii) A few candidates gave partial / incomplete answers.
- (iii) The answers given by some candidates were not complete.
- (iv) Most of the candidates were confused with the size of the tree and mentioned the height or level of the tree instead. Only a few candidates could answer this part correctly.

#### Suggestions for teachers

- More programs / algorithms should be practiced with link list and binary tree data structure. Diagrams to illustrate the link list and the Binary Trees must be practiced.
- Definition of complexities / big 'O' and the three cases of complexities must be explained in details along with examples. Important keywords 'growth rate', 'efficiency' etc. must be used / emphasized.
- Explain binary tree with the different parts like root, nodes(internal and external), height, depth, level, size, tree traversal (preorder, inorder and postorder), etc.

## MARKING SCHEME

### Question 13.

(a) **Algorithm to add node at the end of an existing linked list.**

**Steps :**

- 1 - Start
- 2 - Set temporary pointer to start node
- 3 - Repeat steps 4 until it reaches null
- 4 - move pointer to the next node
- 5 - create new node, assign num to number and link to temporary node
- 6 - End

**Method to add node at the end of an existing linked list.**

```
void addnode ( node start, int num)
{
    Node A = new Node(start)
    while (A != null)
        A = A.nextNode;
    Node C = new node ( );
    C.number = num;
    C.nextNode = null;
    A.next = C; ;
}
```

(b) **Complexity:** It is the measurement or growth rate of an algorithm with respect to time and space.

Big 'O' notation: it is unit of measurement of an algorithm or represents complexity.

- (c)
- i. A
  - ii. B , C , F , D , G , E , H
  - iii. C , D , E , B , G , H , F , A , K , L , J
  - iv. 11

## GENERAL COMMENTS:

(a) **Topics found difficult by candidates in the Question Paper:**

- Truth table of conjunction and disjunction in a proposition.
- Advantages of using the concept of inheritance.
- Difference between `nextInt()` and `hasNextInt()` functions
- Output of a character array without mentioning subscript.
- K-MAPS (Labeling , Grouping , map-rolling , place value)
- Passing objects to functions.

- Recursive technique and invoking function within another function (Nesting of functions).
- Pushing marks from array to stack.
- Size of a binary tree

**(b) Concepts in which candidates got confused:**

- Conjunction and disjunction in a proposition.
- Difference between 'nextInt()' and 'hasNextInt()' functions.
- Representing a single gate after reducing a logic circuit diagram.
- Truth table of a 3 input XNOR gate.
- Passing of objects in the mix() function.
- Invoking function within function.
- Concatenation of strings.

**(c) Suggestions for students:**

Apart from Text book learning, look up other reliable resources to gather comprehensive knowledge of the subject.

Prepare summary for each chapter or use highlighters to recognize important terms and definitions. Knowledge of the prescribed syllabus and its scope is very important.

Answers and definitions should be short and precise and according to marks allotted. Important words and terms should be underlined or highlighted.

Working should be shown at the side of each question wherever required.

Laws must be mentioned while reducing a Boolean Expression.

Practice one form of K-Map with proper place value for both SOP and POS.

In programming, documentation is compulsory and should be mentioned with each program.

Declare the class with data members and member functions. Expand or define each function according to the instructions given by the side of each function.

Do not memorize the program, try to understand the logic.

Practice constructors with every program.

Treat each function of a class as separate program.