COMPUTER SCIENCE
PAPER 1
(THEORY)

Maximum Marks: 70
Time Allotted: Three Hours
Reading Time: Additional Fifteen minutes

Instructions to Candidates

- You are allowed **additional 15 minutes** for **only** reading the question paper.
- You must **NOT** start writing during the reading time.
- This question paper has **12** printed pages.
- It is divided into **two parts** and has **11 questions** in all.
- **Part I** is compulsory and has **two** questions. Answer **all** questions.
- **Part II** is divided into **three sections**: A, B and C.
- While attempting **Multiple Choice Questions** in Part I, you are required to **write only ONE option as the answer**.
- **Each section in Part II** has **three questions. Any two** questions have to be attempted from each section.
- The intended marks for questions are given in brackets [ ].

Instruction to Supervising Examiner

- Kindly read **aloud** the Instructions given above to all the candidates present in the examination hall.

PART I – 20 MARKS

*Answer **all** questions.*

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

Question 1

(i) The compliment of the Boolean expression A' • (B • C' + B' • C) **(Application)** **[1]**

(a) A' • (B+C+B' +C)

(b) A+ (B+C') •(B+C')

(c) A+(B' +C) • (B+C')

(d) A' • (B' +C' +B' •C)

(ii) Given below are two statements marked Assertion and Reason. Read the two statements carefully and choose the correct option. **[1]**

Assertion: Recursion utilises more memory as compared to iteration.

Reason: Time complexity of recursion is higher due to the overhead of maintaining the function call stack. **(Analysis)**

(a) Both Assertion and Reason are true and Reason is the correct explanation for Assertion.

(b) Both Assertion and Reason are true but Reason is not the correct explanation for Assertion.

(c) Assertion is true and Reason is false.

(d) Assertion is false and Reason is true.

(iii) According to the Principle of duality, the Boolean equation (A' + B) • (1 + B) = A' + B will be equivalent to: **(Application)** **[1]**

(a) (A + B') • (0 + B) = A + B'

(b) (A' • B) + (0 • B) = A' • B

(c) (A' • B) + (0 • B) = A' + B

(d) (A' + B) • (0 + B) = A' + B

(iv) Distributive law states that: **(Recall)** **[1]**

(a) A + B • C = (A + B) • (A +C)

(b) A + (A • B) = A

(c) A • (B + C) = (A • B) + (B • C)

(d) A + B • C = A • B + A • C

(v) The complement of the reduced expression of F(A,B) = Σ (0,1,2,3) is: **(Application)** **[1]**

(a) 1

(b) A • B

(c) 0

(d) A' + B'

(vi) Study the given propositions and the statements marked Assertion and Reason that follow it. Choose the correct option on the basis of your analysis. **[1]**

p = I am a triangle
q = I am a three-sided polygon
s1 = p → q
s2 = q → p

Assertion: s2 is converse of s1
Reason: Three-sided polygon must be a triangle. **(Analysis)**

(a) Both Assertion and Reason are true and Reason is the correct explanation for Assertion.

(b) Both Assertion and Reason are true but Reason is not the correct explanation for Assertion.

(c) Assertion is true and Reason is false.

(d) Assertion is false and Reason is true.

(vii) Given below are two statements marked Assertion and Reason. Read the two statements carefully and choose the correct option. **[1]**

Assertion: In Java, the *String* class is used to create and manipulate strings, and it is immutable.

Reason : Immutability ensures that once a *String* object is created, its value cannot be changed. **(Analysis)**

(a) Both Assertion and Reason are true and Reason is the correct explanation for Assertion.

(b) Both Assertion and Reason are true but Reason is not the correct explanation for Assertion.

(c) Assertion is true and Reason is false.

(d) Assertion is false and Reason is true.

(viii) Consider the following statement written in class *Circle* where *pi* is its data member. **[1]**

static final double pi = 3.142;

Which of the following statements are valid for *pi*?

I. It contains a common value for all objects class *Circle*.
II. Its value is non-changeable.
III. At a time two access modifiers, static and final, cannot be applied to a single data member *pi*. **(Application)**

(a) I and II

(b) II and III

(c) I and III

(d) Only III

(ix) For Big O notation, state the difference between O(n) and O(n^2). **(Analysis)** **[1]**

(x) A full adder needs five gates and those are 3 AND gates, 1 OR gate and 1 XOR gate. When a full adder is constructed using 2 half adders, it also requires 5 gates. State the names along with the quantity those gates. **(Analysis)** **[1]**

Question 2

(i) Convert the following *infix notation* to *prefix* form. **(Create)** **[2]**

(A – B) / C * (D + E)

(ii) A matrix M[-6….10, 4…15] is stored in the memory with each element requiring 4 bytes of storage. If the base address is 1025, find the address of M[4][8] when the matrix is stored in **column major wise**. **(Application)** **[2]**

(iii) The following function *getIt*() is a part of some class. Assume ***x*** is a positive integer, *f* is the lower bound of *arr[ ]* and *l* is the upper bound of the *arr[ ]*.

Answer the questions given below along with dry run/working.

```
public int getIt(int x,intarr[],int f,int l)
  {
    if(f>l)
       return -1;
    int m=(f+l)/2;
    if(arr[m]<x)
       return getIt(x,m+1,l);
    else if(arr[m]>x)
       return getIt(x,f,m-1);
    else
       return m;
  }
```

(a) What will the function **getIt( )** return if **arr[ ] = {10,20,30,40,50}** and **x=40**? **(Analysis)** **[2]**

(b) What is function **getIt( )** performing apart from recursion? **(Analysis)** **[1]**

(iv) The following is a function of class *Armstrong*. This *recursive function* calculates and returns the sum of the cubes of all the digits of *num*, where *num* is an integer data member of the class *Armstrong*.
[A number is said to be Armstrong if the sum of the cubes of all its digits is equal to the original number].
There are some places in the code marked by **?1?, ?2?**,**?3?** which may be replaced by a statement/expression so, that the function works properly.

```
public int sumOfPowers(int num)
  {
      if (num == 0)
          return ?1?;
      int digit = ?2?;
          return (int) Math.pow(digit, 3) + ?3?;
  }
```

(a) What is the expression or statement at **?1?** **(Analysis)** **[1]**

(b) What is the expression or statement at **?2?** **(Analysis)** **[1]**

(c) What is the expression or statement at **?3?** **(Analysis)** **[1]**

PART II– 50 MARKS

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

SECTION - A

*Answer **any two** questions.*

Question 3

(i) A shopping mall announces a special discount on all its products as a festival offer only to those who satisfy any one of the following conditions. **[5]**

- If he/she is an employee of the mall and has a service of more than 10 years.

OR

- A regular customer of the mall whose age is less than 65 years and should not be an employee of the mall.

OR

- If he/she is a senior citizen but not a regular customer of the mall.

The inputs are :

INPUTS	
E	Employee of the mall
R	Regular customer of the mall
S	Service of the employee is more than 10 years
C	Senior citizen of 65 years or above

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

Output: **X** - Denotes eligible for discount [1 indicates YES and 0 indicates NO in all cases]

Draw the truth table for the inputs and outputs given above and write the **SOP** expression for **X (E, R, S, C)**. **(Application)**

(ii) Reduce the above expression **X (E, R, S, C)** by using 4-variable Karnaugh map, showing the various groups (i.e. octal, quads and pairs). **[5]**

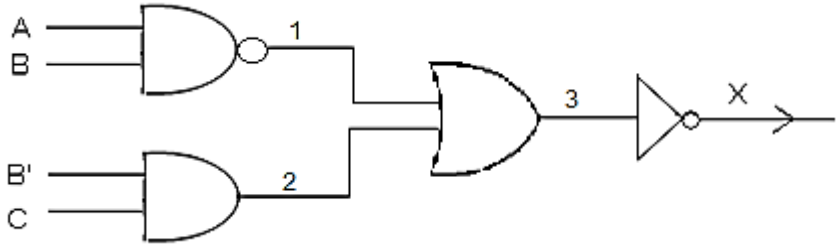
Draw the logic gate diagram for the reduced expression. Assume that the variables and their complements are available as inputs. **(Create)**

Question 4

(i) (a) Reduce the Boolean function **F(P,Q,R,S) = (P+Q+R+S) •(P+Q+R+S') • (P+Q+R'+S)•(P+Q'+R+S) • (P+Q'+R+S') • (P+Q'+R'+S) • (P+Q'+R'+S') •(P'+Q+R+S)• (P'+Q+R+S')** by using 4-variable Karnaugh map, showing the various groups (i.e. octal, quads and pairs). **[4]** **(Application)**

(b) Draw the logic gate diagram for the reduced expression. Assume that the variables and their complements are available as inputs. **[1]** **(Create)**

(ii) From the given logic diagram :



(a) Derive Boolean expression and draw the truth table for the derived expression. **(Application)** **[4]**

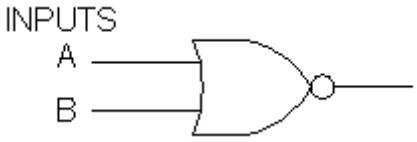
(b) If A=1, B=0 and C=1 then find the value of X. **(Application)** **[1]**

Question 5

(i) Draw the logic circuit to decode the following binary number **(0001, 0101, 0111, 1000, 1010, 1100, 1110,1111)** to its hexadecimal equivalents. Also state the Hexadecimal equivalents of the given binary numbers. **(Application & Analyse)** **[5]**

(ii) Verify if the following proposition is valid using the truth table: **[3]**
$(X \wedge Y) \Rightarrow Z = (Y \Rightarrow Z) \wedge (X \Rightarrow Y)$ **(Application)**

(iii) Answer the following questions related to the below image:



(a) What is the output of the above gate if input A=0, B=1? **(Analysis)** **[1]**

(b) What are the values of the inputs if output =1? **(Application)** **[1]**

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 6

Given are two strings, input string and a mask string that remove all the characters of the mask string from the original string. **[10]**

Example: INPUT: ORIGINALSTRING: **communication**
MASK STRING: **mont**
OUTPUT: **cuicai**

A class **StringOp** is defined as follows to perform above operation.

Some of the members of the class are given below:

Class name	:	**StringOp**
Data members/instance variables:		
str	:	to store the original string
msk	:	to store the mask string
nstr	:	to store the resultant string
Methods / Member functions:		
StringOp()	:	default constructor to initialize the data member with legal initial value
void accept( )	:	to accept the original string str and the mask string msk in lower case
void form()	:	to form the new string nstr after removal of characters present in mask from the original string
void display( )	:	to display the original string and the newly formed string nstr

Specify the class **StringOp** giving details of the **constructor( )**, **void accept( )**, **void form()** and **void display( )**. Define a **main( )** function to create an object and call all the functions accordingly to enable the task**.** **(Create)**

Question 7 **[10]**

A class **Mixarray** contains an array of integer elements along with its capacity (More than or equal to 3). Using the following description, form a new array of integers which will contain only the first 3 elements of the two different arrays one after another.

Example: Array1: { 78, 90, 100, 45, 67 }
Array2: {10, 67, 200, 90 }
Resultant Array: { 78, 90, 100, 10, 67, 200}

The details of the members of the class are given below:

Class name	:	**Mixarray**
Data members/instance variables:		
arr[]	:	integer array
cap	:	integer to store the capacity of the array
Member functions/methods:		
Mixarray (int mm)	:	to initialize the capacity of the array cap=mm
void input( )	:	to accept the elements of the array
Mixarray mix(Mixarray P, Mixarray Q)	:	returns the resultant array having the first 3 elements of the array of objects P and Q
void display( )	:	to display the array with an appropriate message.

Specify the class **Mixarray**giving details of the **constructor(int)**, **void input( )**, **Mixarray mix(Mixarray,Mixarray)** and **void display( ).** Define a **main( )** function to create objects and call all the functions accordingly to enable the task. **(Create)**

Question 8 **[10]**

A class **LCM** has been defined to find the Lowest Common Multiple of two integers.
Some of the data members and member functions are given below:

Class name	:	**LCM**

Data members/instance variables:

n1	:	to store an integer number
n2	:	to store an integer number
large	:	integer to store the largest from n1,n2
sm	:	integer to store the smallest from n1,n2
l	:	to store lcm of two numbers

Methods / Member functions:

LCM( )	:	default constructor to initialize data members with legal initial values
void accept( )	:	to accept n1 and n2
int getLCM( )	:	returns the lcm of n1 and n2 using the **recursive technique**
void display( )	:	to print the numbers n1, n2 and lcm

Specify the class **LCM** giving details of the **constructor( )**, **void accept( )**, **int getLCM() and void display( )**. Define a **main ( )** function to create an object and call the member functions accordingly to enable the task. **(Create)**

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 9

Recycle is an entity which can hold at the most 100 integers. The chain enables the user to add and remove integers from both the ends i.e. front and rear.

Define a class **ReCycle** with the following details:

Class name : **ReCycle**

Data members/instance variables:

ele[ ]	:	the array to hold the integer elements
cap	:	stores the maximum capacity of the array
front	:	to point the index of the front
rear	:	to point the index of the rear

Methods / Member functions:

ReCycle (int max)	:	constructor to initialize the data cap = max, front = rear = 0 and to create the integer array.
void pushfront(int v)	:	to add integers from the front index if possible else display the message("full from front").
int popfront( )	:	to remove the return elements from front. If array is empty then return-999.
void pushrear(int v)	:	to add integers from the front index if possible else display the message("full from rear").
int poprear( )	:	to remove and return elements from rear. If the array is empty then return-999.

(i) Specify the class **ReCycle** giving details of the functions **void pushfront(int)** and **int poprear( )**. Assume that the other functions have been defined. **[4]**

The main( ) function and algorithm need NOT be written. **(Create)**

(ii) Name the entity described above and state its principle. **(Understanding)** **[1]**

Question 10 **[5]**

A library issues books on rental basis at a 2% charge on the cost price of the book per day. As per the rules of the library, a book can be retained for 7 days without any fine. If the book is returned after 7 days, a fine will also be charged for the excess days as per the chart given below:

Number of excess days	Fine per day (Rs.)
1 to 5	2.00
6 to 10	3.00
Above 10	5.00

A super class **Library** has been defined. Define a sub class **Compute** to calculate the fine and the total amount. The details of the members of both the classes are given below:

Class name : **Library**

Data members/instance variables:

name	:	to store the name of the book
author	:	to store the author of the book
p	:	to store the price of the book (in decimals)

Methods / Member functions:

Library(...)	:	parameterized constructor to assign values to the data members
void show( )	:	displays the book details

Class name **Compute**

Data members/instance variables:

d	:	number of days taken in returning the book
f	:	to store the fine (in decimals)

Methods / Member functions:

Compute(...)	:	parameterized constructor to assign values to the data members of both the classes
void fine( )	:	calculates the fine for the excess days as given in the table above
void show( )	:	displays the book details along with the number of days, fine and the total amount to be paid. Total amount is (2% of price of book * total no of days) + fine

*Assume that the super class **Library** has been defined*. Using the **concepts of Inheritance**, specify the class **Compute** giving the details of **constructor**, **void fine ( )** and **void show ( )** functions.

The super class, main function and algorithm need NOT be written. **(Create)**

Question 11

(i) A linked list is formed from the objects of the class **Node**. The class structure of the Node is given below: **[2]**

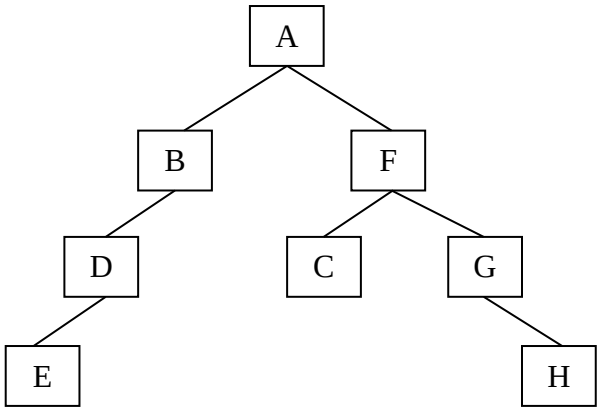
```
class   Node
{
        int n;
        Node link;
}
```

Write an *Algorithm* **OR** a *Method* to search for a number from an existing linked list.

The method declaration is as follows:

void FindNode(Node str, int b) **(Analysis)**

(ii) Answer the following questions from the diagram of a Binary Tree given below:



(a) Name the root of the left sub tree and its siblings. **(Understanding)** **[1]**

(b) State the size and depth of the right sub tree. **(Understanding)** **[1]**

(c) Write the in-order traversal of the above tree structure. **(Understanding)** **[1]**

COMPUTER SCIENCE
PAPER 1
(THEORY)
ANSWER KEY
PART I– 20 MARKS

Question 1

(i) (c) A + (B' + C) • (B + C') **[1]**

(ii) (b) Both Assertion and Reason are true but Reason is not the correct explanation for Assertion. **[1]**

(iii) (b) (A' • B) + (0 • B) = A' • B **[1]**

(iv) (a) A + B • C = (A + B) • (A +C) **[1]**

(v) (c) 0 **[1]**

(vi) (b) Both Assertion and Reason are true but Reason is not the correct explanation for Assertion. **[1]**

(vii) (a) Both Assertion and Reason are true and Reason is the correct explanation for Assertion. **[1]**

(viii) (a) I and II **[1]**

(ix) O(n) – Single loop O(n^2) – Nested loop **[1]**

(x) 2 AND gates, 1 OR gate and 2 XOR gates **[1]**

Question 2

(i) */-ABC+DE **[2]**

(ii) Address of M[4][8] = B + W * ((J – LC) * M + (I – LR)) **[2]**
= 1025 + 4 ((8-4) 17+4 - - 6)
= 1025 + 4(68 +10) = 1025+4 *78
= 1025 + 312 = 1337

(iii) (a) 3 **[2]**

(b) Function *getIt( )* is performing binary search on *arr [ ]* **[1]**

(iv) (a) 0 **[1]**

(b) num%10 **[1]**

(c) sumOfPowers(num/10) **[1]**

PART II– 50 MARKS

SECTION – A

Question 3

(i) **[5]**

E	R	S	C	X
Employee of the mall	Regular customer of the mall	Service of the employee is more than 10 years	Senior citizen of 65 years or above	OUTPUT
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
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	0
1	1	1	0	1
1	1	1	1	1

$X(E,R,S,C) = \sum (1,3,4,6,9,10,11,14,15)$

$= E'R'S'C + E'R'SC + E'RS'C' + E'RSC' + ER'S'C + ER'SC' + ER'SC + ERSC' + ERSC$

(ii) **[5]**

	S' C'	S' C	S C	S C'
E' R'	0 0	1 **1**	3 **1**	2 0
E' R	4 **1**	5 0	7 0	6 **1**
E R	12 0	13 0	15 **1**	14 **1**
E R'	8 0	9 **1**	11 **1**	10 **1**

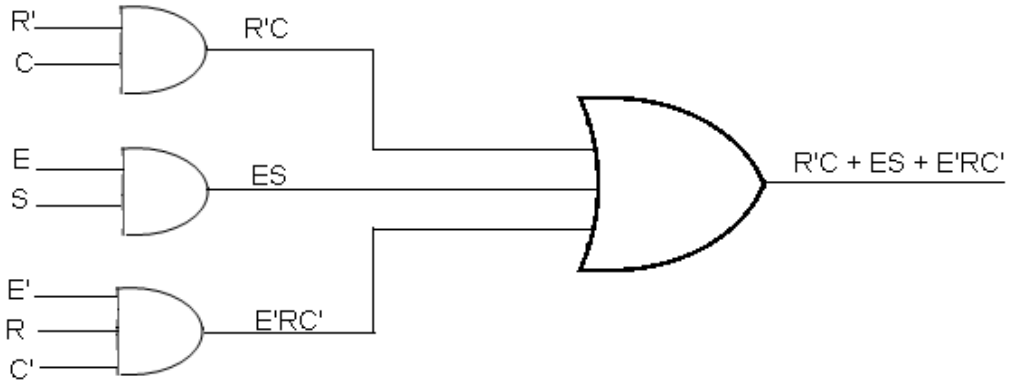
There are two quads and a pair:

Quad 1 ($m_1 + m_3 + m_9 + m_{11}$) = R' C

Quad 2 ($m_{10} + m_{11} + m_{14} + m_{15}$) = E S

Pair ($m_4 + m_6$) = E' R C'

Hence F(E,R,S,C) = R' C + E S + E' R C'



Question 4

(i) (a) **[4]**

	R+S	R+S'	R'+S'	R'+S
P+Q	0 **0**	1 **0**	3 1	2 **0**
P+Q'	4 **0**	5 **0**	7 **0**	6 **0**
P'+Q'	12 1	13 1	15 1	14 1
P'+Q	8 **0**	9 **0**	11 1	10 1

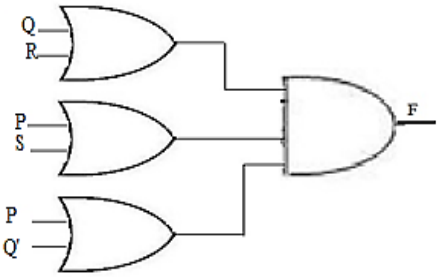
There are three quads :

Quad 1: ($M_0 M_1 M_8 M_9$) = **Q + R** Quad 2: ($M_4 M_5 M_6 M_7$) = **P + Q'**

Quad 3: ($M_0 M_2 M_4 M_6$) = **P + S**

Hence F(P,Q,R,S) = (Q + R) .(P + Q') . (P + S)

(b) **[1]**



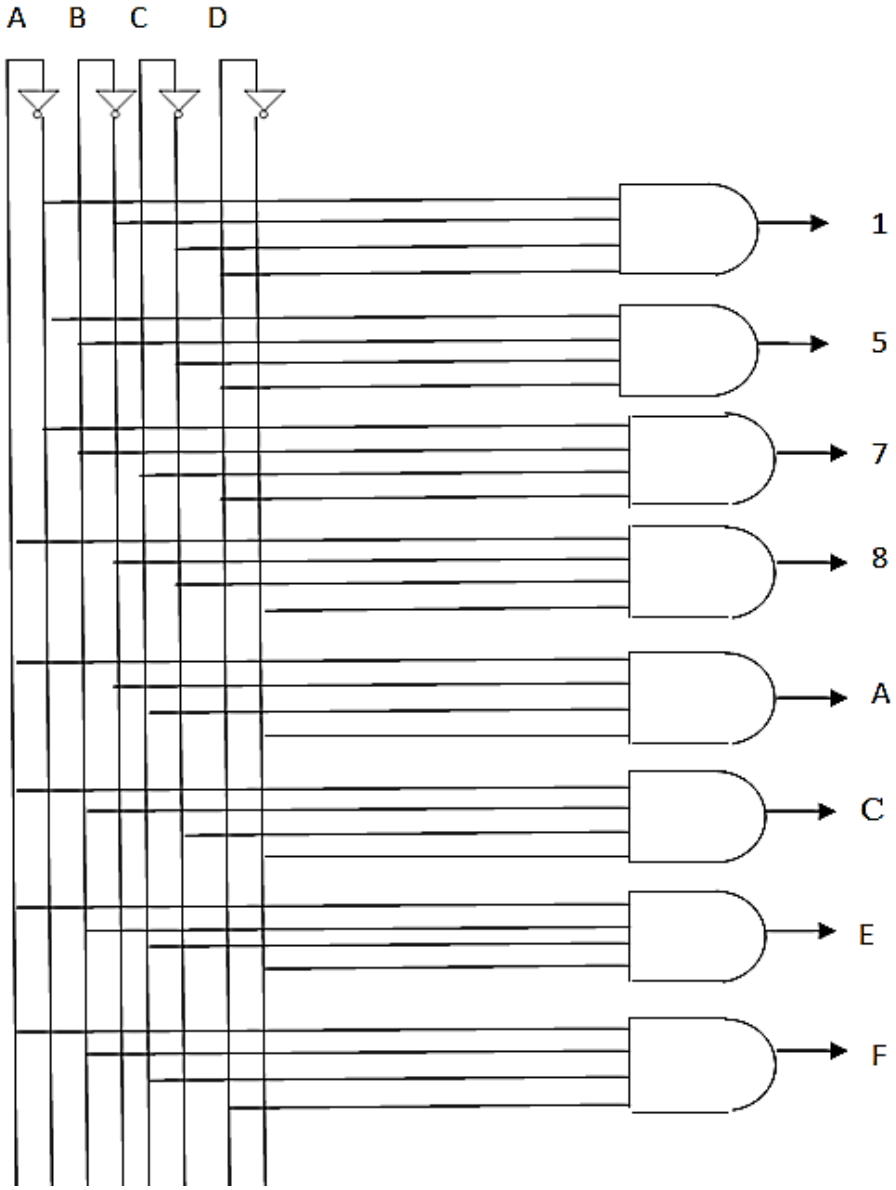
(ii) (a) [(A.B)' + B'.C]' **[4]**

A	B	C	B'	B'.C	(A.B)'	(A.B)' + B'.C	[(A.B)' + B'.C]'
0	0	0	1	0	1	1	0
0	0	1	1	1	1	1	0
0	1	0	0	0	1	1	0
0	1	1	0	0	1	1	0
1	0	0	1	0	1	1	0
1	0	1	1	1	1	1	0
1	1	0	0	0	0	0	1
1	1	1	0	0	0	0	1

(b) 0 **[1]**

Question 5

(i) **[5]**



(0001) = 1
(0101) = 5
(0111) = 7
(1000) = 8
(1010) = A
(1100) = C
(1110) = E
(1111) = F

(ii) **[3]**

X	Y	Z	X ∧ Y	(X ∧ Y)=>Z	Y=>Z	X=>Y	(Y=>Z)∧(X=>Y)
0	0	0	0	1	1	1	1
0	0	1	0	1	1	1	1
0	1	0	0	1	0	1	0
0	1	1	0	1	1	1	1
1	0	0	0	1	1	0	0
1	0	1	0	1	1	0	0
1	1	0	1	0	0	1	0
1	1	1	1	1	1	1	1

INVALID

(iii) (a) 0 **[1]**

(b) A=0, B=0 **[1]**

SECTION – B

Question 6 **[10]**

```
import java.util.*;
class StringOp
{
   String str;
   String nstr;
   String msk;
   Scanner sc=new Scanner(System.in);
StringOp()
   {
      str="";
      nstr="";
      msk="";
   }
   void accept()//to accept the original and mask string
   {
      System.out.println("Enter the original word");
      str=sc.next()+sc.nextLine();
      System.out.println("enter the mask string");
      msk=sc.next();
   }
   void form() //formation of a new string according to the requirement
   {
      int l1=str.length();int l2=msk.length();
      for (int i=0;i<l1;i++)
      {
         int fr=0;char c1=str.charAt(i);
         if (msk.indexOf(c1)==-1)
             nstr=nstr+c1;
         }
      }
   void display()  //display original and newly formed string
   {
       System.out.println("original string: "+str);
       System.out.println("changed string: "+nstr);
    }
   public static void main()
   {
         StringOp ob=new StringOp();
         ob.accept();
         ob.form();
         ob.display();
   }
}
```

Question 7 **[10]**

```
import java.util.*;
class Mixarray
{  int arr[];
   int cap;
   static Scanner sc=new Scanner(System.in);
Mixarray(int mm)
   {  cap=mm;
      arr=new int[cap];
   }
void input()
{  System.out.println("Enter the content of the array");
   for (int i=0;i<cap;i++)
        arr[i]=sc.nextInt();
}
void display()
{    for (int i=0;i<cap;i++)
           System.out.print(arr[i]+" ");
     System.out.println();
}
Mixarray  mix(Mixarray P,Mixarray Q)
{  Mixarray res=new Mixarray(6);
   int k=0;
   for (int i=0;i<3;i++)
      res.arr[k++]=P.arr[i];
   for (int i=0;i<3;i++)
      res.arr[k++]=Q.arr[i];
   return res;
}
public static void main()
{  System.out.println("enter the capacity of both the array");
   int c1=sc.nextInt();    int c2=sc.nextInt();
   Mixarray ob1=new Mixarray(c1);
   Mixarray ob2=new Mixarray(c2);
   System.out.println("Enter the content of Ist array");
   ob1.input();
   System.out.println("Enter the content of 2nd array");
   ob2.input();
   Mixarray r=new Mixarray(c1+c2);
   Mixarray res=r.mix(ob1,ob2);
   System.out.println("content of the combined array");
   res.display();
  }
}
```

Question 8 **[10]**

```
import java.util.*;
class LCM
{
   int n1,n2;
   int large,sm;
   int l;
   static Scanner sc=new Scanner(System.in);
   void accept()
   {
      System.out.println("enter 2 different integers:");
      n1=sc.nextInt();
      n2=sc.nextInt();
      if (n1>n2)
      {
         large=n1;
         sm=n2;
      }
      elseif (n2>n1)
          { large=n2;
            sm=n1;
          }
      }
      int getLCM()
      { if(large!=sm)
         {
             if (large>sm)
                   large=large-sm;
             else if (large<sm)
                   sm=sm-large;
             return getLCM();
            }
            else
               return (n1*n2)/large;
         }
         void display()
         {
            l=getLCM();
            System.out.println("LCM of "+n1 +"and "+n2+"="+l);
         }
         public static void main()
         {
            LCM ob=new LCM();
            ob.accept();
            ob.display();
         }
       }
```

SECTION – C

Question 9

(i)

```
class ReCycle
{
  void pushfront(int v)
  {     if(front !=0)
            q[front--]=v;
        else
             System.out.println("FULL FROM FRONT");
  }
  int poprear()
  {     if(front !=rear)
            return(q[rear--]);
        else
            return -999;
  }
}
```

[4]

(ii) Entity is dequeue and works on the principle of FIFO **[1]**

Question 10 **[5]**

```
class Compute extends Library
{
      private int d;
      private double f;
      public Compute(String name, String author, double p, int d)
      {
         super(name, author, p);
         this.d=d;
          f=0.0;
       }
       public void fine( )
      {   int d1=d-7;
         if(d1>=1 && d1<=5)
               f = d1*2;
         else if(d1>=6 && d1<=10)
                f = d1*3;
         else
               f = d1*5;
        }
       public void show( )
        {
            super.show( );
            System.out.println("Fine =" +f);
            System.out.println("Total amount ="+ ((0.02*p*d)+f));
         }
      }
```

Question 11

(i) ALGORITHM: **[2]**

Step 1. Start
Step 2. Set temporary pointer to the first node
Step 3. Repeat steps 4 and 5 until the pointer reaches null. Display number not found
Step 4. check for number, if found display, exit
Step 5. Move pointer to the next node
Step 6. End algorithm

METHOD:

```
void FindNode(Node str, int b)
  {
     Node temp=str;
     while(temp.link!=null)
       { if (temp.n == b)
            {System.out.prinln(b+“ is found ”);
             break;
       }
          temp=temp.link;
       }
      if (temp.link= =null)
           System.out.prinln(b+“ is not found ”);
   }
```

(ii) (a) Root: B Sibling: F **[1]**

(b) Size: 4 Depth: 2 **[1]**

(c) E D B A C F G H **[1]**