SAMPLE QUESTION PAPER WITH SOLUTION
SUBJECT: COMPUTER SCIENCE
PAPER 1(THEORY)
PART I (20 Marks)

Question 1

(i) The dual form of the Identity Law: A+1 = 1 is equal to:

(a) A.0 = 0
(b) 0.0 = 0
(c) 1 + 1 = 1
(d) No dual

(ii) **Assertion:** Assume that A=0 and B=1, its Maxterm = (A + B')
Reason: For Maxterms, we consider 0 as the term itself and 1 in its complemented form.
Which one of the following options is correct?

(a) Both Assertion and Reason are true, but Reason is not the correct explanation for Assertion.
(b) Both Assertion and Reason are true, and Reason is the correct explanation for Assertion.
(c) Assertion is true and Reason is false.
(d) Assertion is false and Reason is true

(iii) Find the compliment of:
X.Y'.Z + X.Y +Y.Z'

(a) (X'+Y'+Z') . (X.Y) . (Y' + Z)
(b) **(X'+Y+Z') . (X+Y).(Y'+Z)**
(c) (X.Y.Z) . (X+Y).(Y+Z')
(d) (X+Y+Z) . (X'+Y').(Y+Z)

(iv) Determine the values of A, B, C, and D that make the sum
A' + B + C' + D

(a) A = 1, B = 0, C = 0, D = 0
(b) **A = 1, B = 0, C = 1, D = 0**
(c) A = 0, B = 1, C = 0, D = 0
(d) A = 1, B = 0, C = 1, D = 1

(v) Applying the distributive law to the expression A(B+C'+D) , we get

(a) AB+AC+AD
(b) ABCD
(c) A+B+C+D
(d) **AB+AC'+AD**

(vi) **Assertion:** If p →q AND q →r are two premises, then we can infer that p →r.
Reason: The chain rule of the law of syllogism asserts the given derivation.
Which one of the following options is correct?

(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) State the relevance of the keyword *protected* for a data member of a class

Ans) Protected access specifier is exclusively used to make the member of the class inheritable by any derived class within the same package.

(viii) Write any one purpose of using *inheritance* in the context of Java programming.

Ans) Reusability of code [Consider all valid options].

(ix) If (~p →q) then write its:

(a) Inverse : ~ **(~p →q)**

(b) Converse : **q → ~p**

(x) State any one application of each of the Encoder and Decoder digital circuits.

Ans) Encoders are used to convert digital data into an analogue signal, which can then be transmitted over a communication channel. Decoders are used to convert an analogue signal into digital data, which can then be processed by a computer.

Question 2

(i) Convert the following Infix to Postfix Expression: **A+(B – C *(D/E)*F)**

Ans) ABCDE/*F*-+

Working : A+(B – ((C *(D/E))*F))
A+(B – ((C *(DE/))*F))
A+(B – ((CDE/*))*F))
A+(B – (CDE/*F*))
A+(BCDE/*F*–)
ABCDE/*F*–+

(ii) A matrix A[20][20] is stored in the memory with each element requiring 2 bytes of storage. If the base address is 2000, determine the address at A[10][15] when the matrix is stored in **Column Major Wise.**

Ans) 2620

Working : Rows = 20, BA = 2000, W = 2, A = ?
= 2000 + 2* ((10-0)+(15-0)*20)
= 2000 + 2 * (10+300)
= 2000 + 2*310
= 2620

(iii) The following function **demo( )** is a part of some class:

```
void demo(int num, int q)
{
      if(num>1)
      {
        if(num%q = = 0)
        {
          System.out.print(q+" ");
          demo(num/q,q);
        }
        else
          demo(num, q+1);
      }
}
```

(a) Give the output: (i) When num=24 and q=2
(ii) When num=666 and q=2

Ans) (i) 2 2 2 3

Working :

demo(24, 2)	→	print 2 demo(24/2, 2)
demo(12, 2)	→	print 2 demo(12/2, 2)
demo(6, 2)	→	print 2 demo(6/2, 2)
demo(3, 2)	→	demo(3, 3)
demo(3, 3)	→	print 3 demo(3/3, 3)
demo(1, 3)	→	base case encountered

(ii) 2 3 3 37 (with one space after each factor)

(b) What is being computed in the given method?
Ans) Prime Factorization of any number.

(iv) The following function is a part of some class that sorts an array a[ ] in ascending order using the **Insertion Sort** technique. There are some places in the code marked by **?1?, ?2?, ?3?** which must be replaced by an expression / a statement so that the function works correctly.

```
void insertion(int a[ ])
{
        int b,i,t;
        int m=a.length;
        for(i=1;i<m;i++)
        {
                t=a[i];
                b=i-1;
                while(?1? >=0 && t<=a[b])
                {
                        a[b+1]  =a[b];
                        ?2?;
                }
                ?3? = t;
        }
}
```

(a) What is the expression or statement at **?1?** **Ans) b**
(b) What is the expression or statement at **?2?** **Ans) b=b-1**
(c) What is the expression or statement at **?3?** **Ans) a[b+1]**

Part –II (50 Marks)
SECTION – A

Question 3.

(i) The owner of a company pays the bonus to his salesmen as per the criteria are given below:
If the salesman works overtime for more than 4 hours but does not work on off days/ holidays.
OR
If the salesman works when festival sales are on and updates showroom arrangements.
OR
If the salesman works on an off day/holiday when the festival sales are on.

The inputs are:

INPUTS	
O	Works overtime for more than 4 hours
F	Festival sales are on.
H	Works on an off day / holiday
U	Updates showroom arrangements

(In all the above cases 1 indicates yes and 0 indicates no.)
Output: X [1 indicates yes, 0 indicates no for all cases]
Draw the truth table for the inputs and outputs given above and write the **POS** expression for **X(O, F, H, U).**

Answer:

(*a*)

O Works Overtime for more than 4 hours	F Festival sales are on	H Working on an off day/holiday	U Updates showroom arrangements	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	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

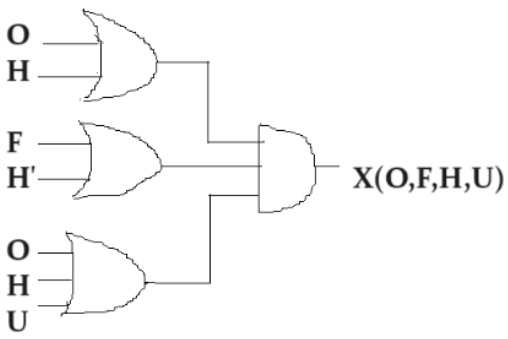
$X(O, F, H, U) = \pi(0, 1, 2, 3, 4, 10, 11)$

(ii) Reduce the above expression **X (O, F, H, U)** by using a 4-variable Karnaugh map, showing the various groups (i.e. octal, quads and pairs). Draw the logic gate diagram for the reduced expression. Assume that the variables and their complements are available as inputs.

OF \ HU	H+U	H+U'	H'+U'	H'+U
O+F	0	0	0	0
O+F'	0			
O'+F'				
O'+F			0	0

Quad1 (0,1,3,2) = O+F
Quad2 (3,2,11,10) = F+H'
Pair(0,4) = O+H+U

X(O,F,H,U) = (O+F) . (F+H') . (O+H+U)



Question 4

(i) (a) Reduce the Boolean function **F(A, B, C, D) = Σ (0, 2, 3, 4, 5, 8, 10, 11, 12, 13)** by using 4-variable Karnaugh map, showing the various groups (i.e. octal, quads and pairs).

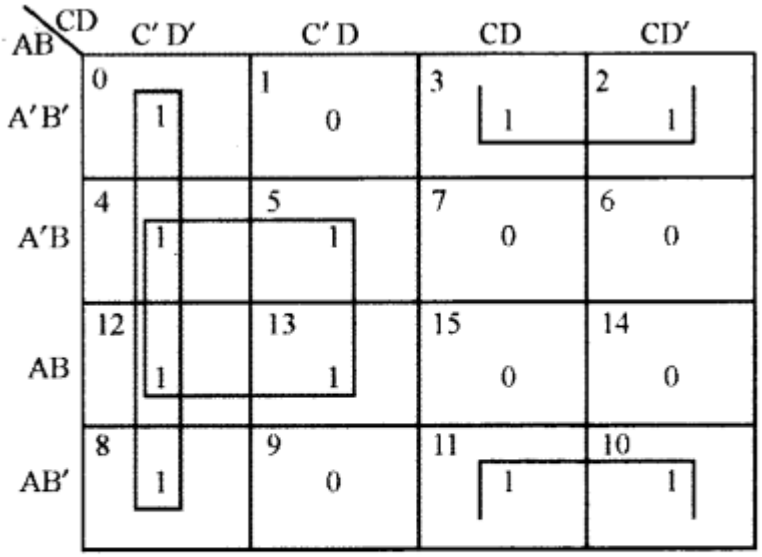
(b) Draw the logic gate diagram for the reduced expression. Assume that the variables and their complements are available as inputs.

Ans)

(i)

(a) Answer:

F(A, B, C, D) = Σ (0, 2, 3, 4, 5, 8, 10, 11, 12, 13)



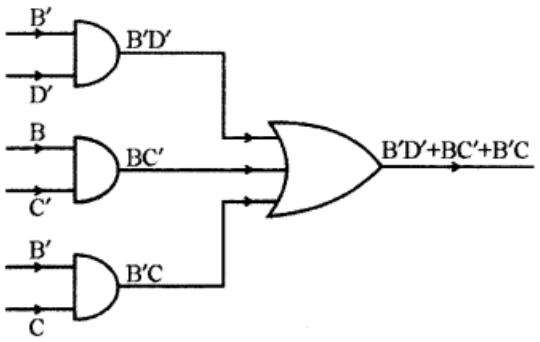
Quad 1 : $(m_0 + m_2 + m_8 + m_{10}) = B'D'$

Quad 2 : $(m_0 + m_4 + m_{12} + m_8) = C'D'$

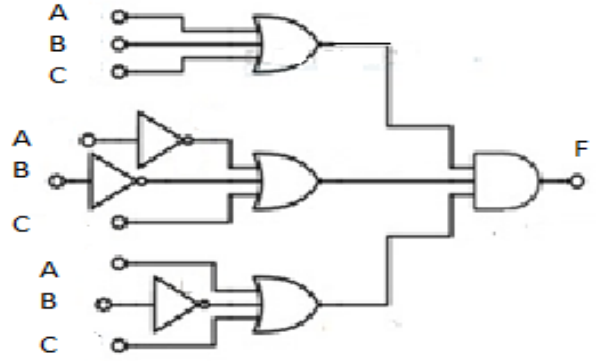
Quad 3 : $(m_3 + m_2 + m_{11} + m_{10}) = B'C$

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

(*ii*)



(ii) (a) From the logic circuit diagram given below, name the 3 output signals and finally derive the Boolean expression (F).

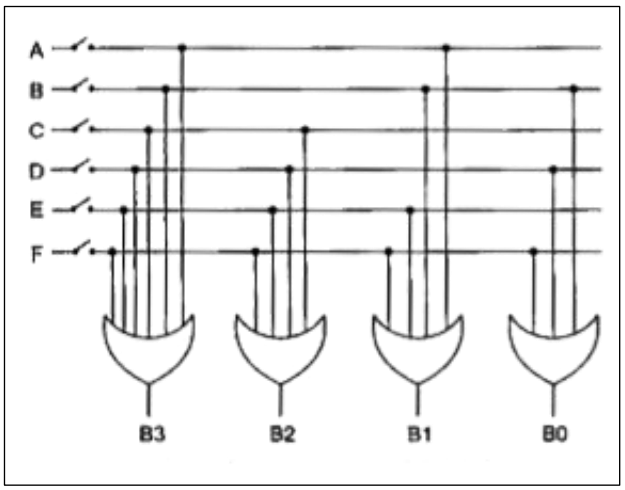


Ans) (1): A+B+C
(2): A'+B'+C
(3): A+B'+C
F = (A+B+C) . (A'+B'+C) . (A+B'+C)

(b) If A=0 and B=1 then find the value of (A.0) + B'
Ans) (0.0) +0 = 0

Question 5
(i) Draw the Encoder circuit to convert A-F hexadecimal numbers to binary.
State an application of Multiplexer.

Ans) A-F Hexadecimal Encoder circuit:



Truth Table:

HEXADECIMAL (INPUT)	BINARY (OUTPUT)			
	B_3	B_2	B_1	B_0
HxA	1	0	1	0
HxB	1	0	1	1
HxC	1	1	0	0
HxD	1	1	0	1
HxE	1	1	1	0
HxF	1	1	1	1

Application of a multiplexer: The multiplexer is used to perform high-speed switching in networking.

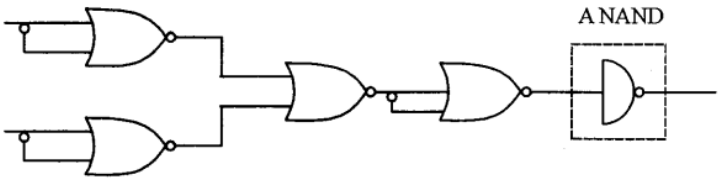
(ii) Using a truth table, state whether the following proposition is a Tautology, Contradiction Or Contingency:
~(P =>Q)<=>(~P ∨ Q)

P	Q	P⇒Q	~(P⇒ Q)	~P	~P∨Q
0	0	1	0	1	1
0	1	1	0	1	1
1	0	0	1	0	1
1	1	1	0	0	1

The last column contains only True values. Therefore, it is Tautology.

(iii) Draw the logic circuit of a NAND gate using NOR gates only.
Ans)

(*e*) A NAND gate can be constructed from the combination of 4 NOR gates.



SECTION – B

Question 6.

A class DeciOct has been defined to convert a decimal number into its equivalent octal number. Some of the members of the class are given below:

Class name : DeciOct

Data members/instance variables:

n	: stores the decimal number
oct	: stores the octal equivalent number

Methods / Member functions:

DeciOct( )	: default constructor to initialize the data members with legal initial values **n = 0, oct = 0**
void getnum(int nn)	: assign **nn** to **n**
int deci_oct(int)	: calculates the octal equivalent of **'n'** and stores it in **oct** and returns it using the **recursive technique**
void show( )	: displays the decimal number **'n'**, calls the function **deci_oct( )** and displays its octal equivalent.

Specify the class **DeciOct,** giving details of the **constructor( ), void getnum(int), int deci_oct(int)** and **void show( )**. Also define a **main( )** function to create an object and call the functions accordingly to enable the task.

Ans)

```
class DeciOct
{
        int n ;
        int oct;
        DeciOct( )
        {
                n = 0;
                oct = 0;
        }
        void getnum.(int nn)
        {
                n = nn;
        }
        int deci_oct( int num)
        {
                int tmp;
                if (num != 0) {
                   oct = oct + (num % 8) * tmp;
                   tmp = tmp * 10;
                   dec_oct(num / 8);
                 }
                 return oct;
        }
        void show( )
        {
                System.out.println (" The decimal number is " + n);
                System.out.println ("The octal of "+ n + " is" + oct);
        }
//main( ) method starts
//input code
//object decl.
```

//func. Call
}

Question 7.
Two matrices are said to be equal if they have the same dimension and their corresponding elements are equal.
For example, the two matrices A and B is given below are equal:

Matrix A

1	2	3
2	4	5
3	5	6

Matrix B

1	2	3
2	4	5
3	5	6

Design a class **EqMat** to check if two matrices are equal or not. Assume that the two matrices have the same dimension.
Some of the members of the class are given below :

Class name	**: EqMat**
Data members/instance variables:	
a[ ][ ]	: to store integer elements
m	: to store the number of rows
n	: to store the number of columns
Member functions/methods:	
EqMat(int, int)	: parameterized constructor to initialize the data members **m = mm and n = nn**
void readArray( )	: to enter elements in the array
boolean check(EqMat P, EqMat Q)	: checks if the parameterized objects P and Q are equal and returns true, otherwise returns false
void print( )	: displays the array elements

Define the class **EqMat** giving details of the **constructor( ), void readarray( ), int check(EqMat, EqMat) and void print( ).** Define the **main( )** function to create objects and call the functions accordingly to enable the task.

Ans: 7)

```
import java.io.*;
import java.util.Scanner;
class EqMat
{
        private int a[ ][ ];
        private static int m;
        private static int n;
        public EqMat(int mm, int nn)
        {
          m = mm;
          n = nn;
          a = newint[m][n];
        }
        public void readArray( )
        {
          Scanner sc = new Scanner(System.in);
          System.out.println("Enter the numbers.");
          for(int i = 0; i < m; i++) {
                for(int j = 0; j < n; j++) {
                a[i][j] = sc.nextInt();
                }
```

```
    }
  }
  public boolean check(EqMat p, EqMat q)
  {
    boolean flag = true;
    for(int i = 0; i < m; i++) {
        for(int j = 0; j < n; j++) {
        if(p.a[i][j] !=q.a[i][j])
        return false;
        }
    }
    return flag;
  }
  public void print( )
  {
    for(int i = 0; i < m; i++) {
        for(int j = 0; j < n; j++) {
        System.out.print(a[i] [j] + "\t");
        }
        System.out.println();
    }
  }
  public static void main(String args[ ])
  {
    Scanner sc = new Scanner(System.in);
    System.out.print("Number of rows: ");
    int rows = sc.nextInt();
    System.out.print("Number of columns: ");
    int columns = sc.nextInt();
    EqMat obj 1 = new EqMat(rows, columns);
    EqMat obj2 = new EqMat(rows, columns);
    System.out.println("Enter elements for first matrix:");
    obj1.readArray();
    System.out.println("Enter elements for second matrix:");
    obj2.readArray();
    System. out.println("First Matrix:");
    obj1.print();
    System.out.println("Second Matrix:");
    obj2.print();
    if(check(obj1, obj2))
    System.out.println("Both Matrices are Equal");
    else
    System.out.println("Matrices are not Equal");
  }
}
```

Question 8.

Design a class Change to perform string-related operations. The details of the class are given below:

Class name	**: Change**
Data Members/instance variables:	
str	: stores the word
newstr	: stores the changed word
len	: store the length of the word

Methods / Member functions:

Change( )	: default constructor to initialize data members with legal initial values
void inputword( )	: to accept a word
char caseconvert (char ch)	: converts the case of the character and returns it
void recchange (int)	: extracts characters using **recursive technique** and changes its case using **caseconvert ( )** and forms a new word
void display ( )	: displays both the words

Specify the class **Change**, giving details of the **constructor ( ), void inputword (), char caseconvert (char ch), void recchange (int) and void display ( ).** Define the **main ( )** function to create an object and call the functions accordingly to enable the above change in the given word.

Ans: 8)

```
import java.io.*;
class change
{
        String str;
        String newstr;
        int len;
        public change()
        {
          str = " " ;
          newstr = " ";
          len = 0;
        }
         public void inputword()
        {
          BufferedReader br = new BufferedReader (new InputStreamReader
          (System.in));
          System.out.println ("Enter the number");
          Str=br.readLine();
        }
        public char caseconvert (char ch)
         {
          if (ch > = 'A' &&. ch<= 'Z')
          {
                ch=(char) ((int) ch + 32);
                return ch;
          }
          if (ch > = 'a' && ch < = 'z')
                return (char) ((int) ch - 32);
          if (ch = = ' ')
                return ch;
          }
        public void recchange (int a)
        {
          if(a<0)
          return;
          else
          newstr+=caseconvert(recchange (a-1));
        }
        void display( )
```

```
        {
          System.out.println (" Old word is " + str);
          System.out.println ("The new word is" + newstr);
        }
//main( ) method starts
//input code
//object decl.
//func. Call
}
```

SECTION – C

Question 9.

Shelf is a kind of data structure which can store elements with the restriction that an element can be added from the rear end and removed from the front end only.

The details of the class **Shelf** are given below:

Class name : Shelf

Data members/instance variables:

ele[ ]	: array to hold decimal numbers
lim	: maximum limit of the shelf
front	: to point the index of the front end
rear	: to point the index of the rear end

Methods / Member functions:

Shelf(int n)	: constructor to initialize lim=n, front= 0 and rear=0
void pushVal(double v)	: to push decimal numbers in the shelf at the rear end if possible, otherwise display the message " SHELF IS FULL "
double popVal( )	: to remove and return the decimal number from the front end of the shelf if any, else returns −999.99

(i) Specify the class **Shelf** giving details of the functions **void pushVal(double)** and **double popVal( )**. Assume that the other functions have been defined.
The main( ) function and algorithm need NOT be written.

(ii) Name the entity described above and state its principle.

Ans:)

```
import java.util.*;
public class Shelf
{
   double ele[ ]=new double[200];
   int lim;
   int front, rear;
   Shelf()
   {
      for(int i=0;i<200;i++)
      {
         ele[i]=0;
      }
   }
```

```
Shelf(int n)
   {
      lim=n;
      front=-0;
      rear=0;
   }

   void pushVal(double v)
   {
      if(rear==(lim-1))
         System.out.println("Shelf is full");

      else
      {
         front=0;
         rear=rear+1;
         ele[rear]=v;
      }
   }

   double popVal( )
   {
      if(front==rear)
      {
         front=-1;
         rear=-1;
         return(ele[front]);
      }
      else if(front==-1 && rear==-1)
      {
         System.out.println("There is no data in the stack");
         return -999.99;
      }
      else
      {
         double a =ele[front];
         front=front+1;
         return a;
      }
   }
```

Question 10.

A superclass **Product** has been defined to store the details of a product sold by a wholesaler to a retailer. Define a subclass **Sales** to compute the total amount paid by the retailer with or without a fine along with service tax. The details of the members of both classes are given below:

Class name : Product

Data members/instance variables:

name : stores the name of the product

code : integer to store the product code

amount : stores the total sale amount of the product (in decimals)

Member functions/methods:

Product (String n, int c, double p): parameterized constructor to assign data members
: name = n, code = c and amount = p

void show( ) : displays the details of the data members

Class name : Sales
Data members/instance variables:
day : stores number of days taken to pay the sale amount
tax : to store the service tax (in decimals)
totamt : to store the total amount (in decimals)

Member functions/methods:
Sales(….) : parameterized constructor to assign values to data members of both the classes
void compute( ) : calculates the service tax @ 12.4% of the actual sales calculates the fine @ 2.5% of the actual sale amount only if the amount paid by the retailer to the wholesaler exceeds 30 days calculates the total amount paid by the retailer as (actual sale amount + service tax + fine)
void show ( ) : displays the data members of the superclass and total

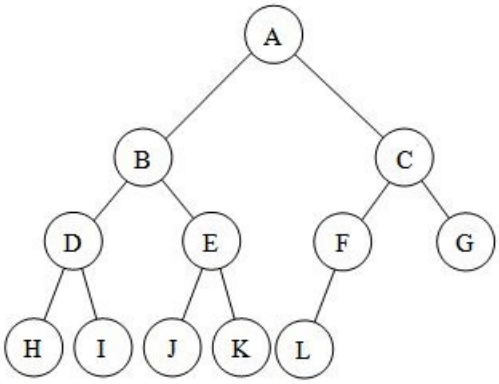
*Assume that the superclass **Product** has been defined.* Using the **concept of inheritance**, specify the class **Sales** giving the details of the **constructor (…), void compute( ) and void show( )**. **The superclass, main function and algorithm need NOT be written.**

Ans)

```
class Sales extends Product
{
        int day;
        double tax;
        double totamt;
        double fine = 0.0;
        Sales(String n, int c, double p, int d) {
          super(n, c, p);
          day = d;
        }
        void compute( ) {
          if(day <= 30){
                tax = 12.4 * amount /100;
                totamt = amount + tax;
                }
          else {
                tax= 12.4 * amount /100;
                fine = 2.5 * amount /100;
                totamt = amount + tax + fine;
          }
        }
        void show ( ) {
          super.show( );
          System.out.println("Total amount to be paid::"+ totamt);
        }
}
```

Question 11.
(i) Mention the two major factors that determine the complexity of an algorithm.
Ans) 1. Space Complexity
2. Time Complexity

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



(a) Write the In-order traversal of the above tree structure.
(b) Write all the existing levels for the entire tree.
(c) Name the siblings of the nodes H and J.

Ans: a) H,.D,.I, J, E, K, B, A, L, F, G, C
Ans: b) Level 0, Level 1, Level 2, Level 3
Ans: c) I and K, respectively.

****

SAMPLE PAPER (UNSOLVED)
For practice

Question 1.

(i) Complementarity law states that –
(a) (A . B)' = A'+ B'
(b) A + A' = 1
(c) A . A' = 0
(d) Both (b) and (c)

(ii) The dual of X + X'.Y is :
a) X.(X'+Y)
b) X.Y
c) X'.(X+Y')
d) X+X'+Y

(iii) The complement of the Boolean expression – AB' + A' + BC is
a) (A + B'). A . (B + C)
b) (A' + B). A . (B' + C')
c) (A' . B') + A + (B' . C')
d) None of the above

(iv) **Assertion**: An operator (or connective) joins simple propositions into compounds, and joins compounds into larger compounds.
Reason : We use symbols +, ., V, ^, →,↔ to designate the sentential connectives only.
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

(v) If X => X.Y then its inverse will be :
   (a) ~X => X.Y
   (b) X => ~X . ~Y
   (c) ~X => ~X + ~Y
   (d) None of these

(vi) Assertion : Two statements are consistent if their conjunction is not a contradiction.
Reason: The proposition having all 0's except one value is a contradiction.
   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) State two advantages of using the concept of inheritance in Java.
(viii) Draw the logic gate of NOT gate using NAND gate only.
(ix) Define *Canonical form* of an expression with respect to its *Cardinal form.*
(x) What is meant by the terms 'Overflow' and 'Underflow' with reference to data structure.

Question 2.

(i) Convert the following infix notation to postfix form.
   A / (B + C) * D + (E – F)

(ii) A 2–D array defined as A[-2...10, 3...8] contains double type elements. If the array is stored in row–major form, calculate the address of A[4][5] given the base address as 4110.

(iii) The following function smile( ) is a part of some class. :

```
String smile(int n)
{  if(n>2)
    {  if(n%2!=0)
       return smile(n/10)+ "A";
      else
       return smile(n/10+2)+ "B";
    }
   else
    return  "";
 }
```

What will the function smile( ) return when the value of
   a) n = 9465
   b) n = 3245
   Show dry run and working also.

(iv) The following function is a part of some class which computes and returns the value of a number 'p' raised to the power 'q' (p^q). There are some places in the code marked by **?1? , ?2? , ?3?** which must be replaced by an expression / a statement so that the function works correctly.

```
double power ( double p , int q )
{
      double r = ?1? ;
      int c = ( q<0 ) ? -q : q ;
      if ( q == 0)
         return  1 ;
      else
```

```
{
        for (int i = 1; i <= c ;?2?, i++);
            return (q>0)? r : ?3?;
    }
}
```

a) What is the expression or statement at **?1?**
b) What is the expression or statement at **?2?**
c) What is the expression or statement at **?3?**

Part II
SECTION - A

Question 3.

i. A company intends to develop a device to show high-status power load for a household inverter depending on the criteria given below:
Condition 1. If Air conditioner and Geyser are on. **OR**
Condition 2. If Air conditioner is off, but Geyser and Refrigerator are on. **OR**
Condition 3. If Geyser is off, but Air conditioner and Water purifier are on. **OR**
Condition 4. When all are on.

INPUTS are :

A : Air Conditioner is on.
G : Geyser is on.
R : Refrigerator is on.
W : Water purifier is on.
(In all the above cases 1 indicates yes and 0 indicates no.)

Output :

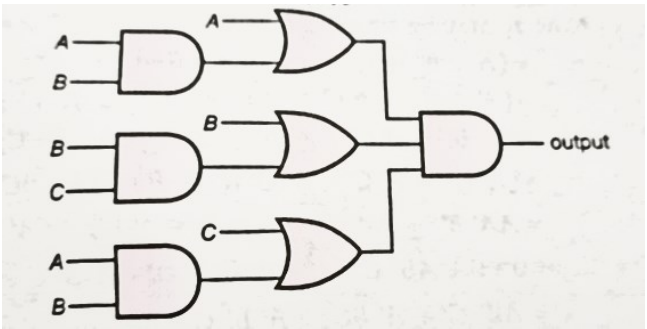
X : (1 indicates high power, 0 indicates no for all cases)
Draw the truth table for the inputs and outputs given above. Write the SOP expression for X(A, G, R, W).

ii. Reduce the above expression **X (A, G, R, W)** by using 4-variable Karnaugh map, showing the various groups (i.e. octal, quads and pairs). Draw the logic gate diagram for the reduced expression. Assume that the variables and their complements are available as inputs.

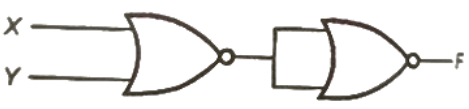
Question 4.

i. Given the Boolean function: F(A, B, C, D) = π (3, 4, 6, 9, 11, 12, 13, 14, 15)
   a. Reduce the above expression by using a 4-variable K-Map, showing the various groups (i.e. octal, quads and pairs).
   b. Draw the logic gate diagram of the reduced expression. Assume that the variables and their complements are available as inputs.

ii. From the logic circuit diagram given below, derive the Boolean expression and simplify it.



iii. Answer the question related to the circuit given below :



   a. Give the output if, X =1 and Y = 0.
   b. Name the basic gate represented by the diagram.

Question 5.

i. What is a multiplexer? Draw the logic diagram for a 8:1 Multiplexer.

ii. Simplify the following expression, using Boolean laws –

$F = [(X' + Y) . (Y' + Z)]'. (X' + Z)$

iii. Define encoder. Draw the truth table and logic gate for an Octal to Binary encoder.

SECTION - B

Question 6.

Design a class Dudeney to check if a given number is a Dudeney number or not.
A Dudeney number is a positive integer that is a perfect cube, such that the sum of its digits is equal to the cube root of the number.

Example: $5832 = (5 + 8 + 3 + 2)^3 = (18)^3 = 5832$

Some of the members of the class are given below.

Class name	:	**Dudeney**
Data members		
Num	:	To store a positive integer number
Member Methods		
Dudeney( )	:	Default constructor to initialize the data members with legal initial value
void input( )	:	To enter a positive integer
int sumDigit(int x)	:	Returns the sum of the digits of number 'x' using a recursive technique
void isDude( )	:	**Checks whether the given number is a Dudeney number or not by invoking the function sumDigits( ) and displays the result with an appropriate message.**

Specify the class **Dudeney** giving details of the constructor and other functions. Define the **main( )** function to create an object and call the functions accordingly to enable the task.

Question 7.

A class Array2D has been defined which can store a maximum of 100 integers.
Some of the members of the class are given below:

Class name	:	**Array2D**
Data member/instance variable:		
int Arr[ ][ ]	:	to store integers in the array
int M	:	To store the actual size of the array.
int num	:	Integer to store a number to be checked
Member methods:		
Array2D(int mm)	:	Parameterized constructor to initialize the data member M = mm and to allocate memory to the member array.
void fnInput( )	:	To enter elements of the array
int count(Array2D X, int K)	:	Returns the frequency of K from the array contained in the object X
void display( )	:	To invoke the function int count(….) and print the frequency of num in the array

Specify the class **Array2D**, giving details of the **Constructor**, and other functions. Define the **main()** function to create an object and call the functions accordingly to enable the task.

Question 8.

A class **Process** has been defined to count vowels and consonants. Some of the members of the class are given below:

Class name	:	**Process**
Data members/instance variables:		
str	:	String to store a sentence
nStr	:	String to store changed sentence
cv	:	Integer to store the number of vowels
cc	:	Integer to store the number of consonants
Methods / Member functions:		
Process( )	:	default constructor to initialize data members with legal initial values
void accSent( )	:	to accept a sentence
void changeCase( )	:	To convert the sentence to capitalize each word and store it in nStr
void countLetters( )	:	To count and print the number of vowels and the number of consonants in each word and display it in the following format:

Example – **Input:** the sky is BLUE

Output:

Word	**Vowels**	**Consonants**
The	1	2
Sky	0	3
Is	1	1
Blue	2	2

Specify the class **Process** giving details of the **constructor( )**, and other functions,
Define a **main ( )** function to create an object and call the functions accordingly to enable the task.

SECTION - C

Question 9.

Shelf is a kind of data structure which can store elements with the restriction that an element can be added from the rear end and removed from the front end only.

The details of the class **Shelf** are given below:

Class name	:	**Shelf**
Data member/instance variable:		
ele[]	:	array to hold decimal numbers
lim	:	maximum limit of the shelf
front	:	to point the index of the front end
rear	:	to point the index of the rear end
Member functions/methods:		
Shelf(int n)	:	constructor to initialize lim=n, front= 0 and rear=0
void pushVal(double v)	:	to push decimal numbers in the shelf at the rear end if possible, otherwise display the message “SHELF IS FULL”
double popVal( )	:	to remove and return the decimal number from the front end of the shelf if any, else returns −999.99
void display( )	:	To display the elements of the Shelf

Specify the class **Shelf**, giving details of the **Constructor**, and the other functions.
The **main()** function need not to be written.

Question 10

A super class **Demand** has been defined to store the details of the demands for a product.
Define a subclass **Supply** which contains the production and supply details of the products.
The details of the members of both the classes are given below:

Class name	:	**Demand**
Data members/instance variables:		
pid	:	string to store the product ID

pname	:	string to store the product name
pdemand	:	integer to store the quantity demanded for the product
Member functions:		
Demand(...)	:	parameterized constructor to assign values to the data members
void display( )	:	to display the details of the product
Class name		**Supply**
Data members:		
pproduced	:	integer to store the quantity of the product produced
prate	:	to store the cost per unit of the product in decimal
Member functions:		
Supply(...)	:	parameterized constructor to assign values to the data members of both the classes
double calculation( )	:	returns the difference between the amount of demand (rate × demand) and the amount produced (rate × produced)
void display( )	:	to display the details of the product and the difference in the amount of demand and amount of supply by invoking the method *calculation()*

*Assume that the super class **Demand** has been defined*. Using the **concept of inheritance**, *specify* the class **Supply** giving the details of the **constructor(...)**, **double calculation( )** and **void display( )**. **The super class, main function and algorithm need NOT be written.**

Question 11

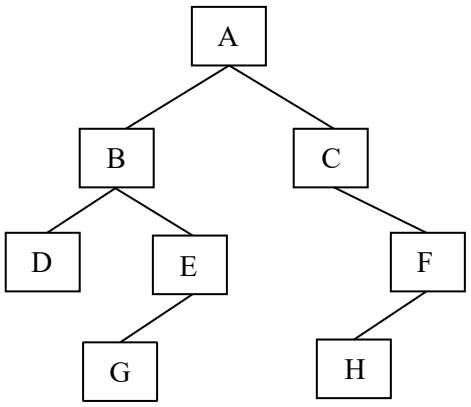
a) Define the Dominant term. Write any two external factors that affect the algorithm's performance.

b) Answer the following questions from the diagram of a binary tree given below :

i. Write the preorder traversal of the tree structure.

ii. Name the leaf nodes of the tree.

iii. State the height of the tree, if the root is at level 0 (zero).



Some more practice problems:

Question 1.

(i) The dual form of the Identity Law: A+0 = A is equal to:

(e) A.0 = 0
(f) A+0 = A'
(g) A + 1 = 1
(h) A .1 = A

(ii) **Assertion:** Given the Boolean statement for F(A,B,C) as A+B.C = (A+B).(A+C)
Reason: The above expression is a statement from Distributive law.
Which one of the following options is correct?

(a) Both Assertion and Reason are true, but Reason is the correct explanation for Assertion.
(b) Both Assertion and Reason are true, and 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) Find the compliment of : F(A,B,C) = (A+B)' . (A'+B')

(a) (A+B) . (A+B)
(b) (A+B) + (A.B)
(c) A'.B' + A.B
(d) (A'+B').(A.B)

(iv) Determine the values of A, B, C, and D that will give the value of the term as 1:
A'. B . C . D'

(a) A = 1, B = 0, C = 0, D = 0
(b) A = 1, B = 0, C = 1, D = 0
(c) A = 0, B = 1, C = 1, D = 0
(d) A = 1, B = 0, C = 1, D = 1

(v) Applying the distributive law to the expression A.(B+C') + A.(B'+C), we get

(a) (A+B).(A+C') + (A+B').(A+C)
(b) A+B.C' + A+B'.C
(c) A.B + A.C' + A.B' + A.C
(d) A.(B'+C) . A(B+C')

(vi) **Assertion:** The complements law states that A+A' = 1 and A.A'=0.
Reason: The involution rule states that A'' = A.
Which one of the following options is correct?

(a) Both Assertion and Reason are true, but Reason is the correct explanation for Assertion.
(b) Both Assertion and Reason are true, and 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) State the relevance of the keyword ***static*** for a data member of a class.

(viii) Write any one purpose of using ***void*** in the context of Java programming.

(ix) If (~p → ~q) then write its:

(c) Inverse
(d) Converse

(x) Why NAND and NOR gates are called Universal gates?

Question 2.

(i) Let P= "I will be an Engineer", Q= "My stream is CS" and R= "My grades are excellent". Express the given propositions in Boolean Expression:

(a) If my stream is CS and my grades are excellent, only then I will be an Engineer.

(b) I will not be an Engineer but my stream is CS as my grades are not excellent.

(ii) State the Associative law of Boolean Algebra and prove that using a truth table.

(iii) The following function **demo( )** is a part of some class:

```
void demo(int num)
{
        int s=0;
        while(num>0)
        {
                int p=1;
                for(int i=1; i<=num%2; i++)
                        p*=i;
                s+=p;
                num/=2;
        }
        System.out.println(s);
}
```

(a) Give the output when num=125

(b) Show the working.

(iv) The following function is a part of some class that sorts an array a[ ] in descending order using the **Selection Sort** technique. There are some places in the code marked by **?1? , ?2? , ?3?** which must be replaced by an expression / a statement so that the function works correctly.

```
void sort(int a[ ])
{
        int len=a.length;
        for(int i=0; ?1? ; i++)
        {
          int p=i;
          for(int j=i+1; j<len; i++)
                if( ?2? )
                  ?3?
          int t=a[i];
          a[i]=a[p];
          a[p]=t;
        }
}
```

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

(b) What is the expression or statement at **?2?**

(c) What is the expression or statement at **?3?**

Question 3.

(i) The tourist company is giving a gift tour to its customer as per the criteria given below:
If the person is a regular customer of the company and taking the service for a continuous 5 years.

OR

If the person is a regular customer and taking a foreign tour for a consecutive 3 times in a year.

OR

If the person is not a regular customer but brings 10 new customers to the company.

The inputs are:

INPUTS	
A	The person is a regular customer of the company.
B	The person is taking the service for a continuous 5 years from the company.
C	The person is taking a foreign tour for a consecutive 3 times in a year.
D	The person brings 10 new customers to the company.

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

Output: F [1 indicates yes, 0 indicates no for all cases]

Draw the truth table for the inputs and outputs given above and write the **SOP** expression for **F(A,B,C,D).**

(ii) Reduce the given Boolean expression using Boolean Laws:

F(A,B,C) = A . (A'+B') . C + A.B.C

Draw the logic gate diagram using the universal NAND gate only for the reduced expression. Also, state the name of the reduced gate.

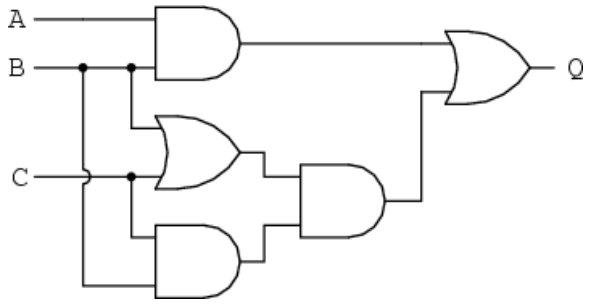
Question 4

(j) (a) Draw the truth table for the following Boolean expression and state it is a tautology or contradiction or contingency:

F = (X+Y+Z) . (X'+Y') . Z'

(b) Define well-formed formulae (wff).

(ii) (a) Write down the Boolean expression for Q (A, B, C) from the given logic diagram.



What will be the reduced form of the above circuit?

(b) Differentiate between disjunction and conjunction.

Question 5.

(i) Draw the truth table and logic diagram of a full adder circuit. State one application of Full Adder.

(ii) Using a truth table, state whether the following proposition is a Tautology, Contradiction or Contingency: $\boldsymbol{F(P,Q) = P.Q \leftrightarrow (\overline{P} + \overline{Q})}$

(iii) Draw the logic circuit of the above Boolean expression.

SECTION – B

Question 6.

Rare numbers are the non-palindromic numbers when added and subtracted to its reverse giving a perfect square. For example, 65 is a rare number because $65 + 56 = 121 = 11^2$ and $65 - 56 = 9 = 3^2$.

A class RareNum has been defined to check whether a positive number is a Real Number or not. Some of the members of the class are given below:

Class name	**: RareNum**
Data members/instance variables:	
n	: stores the positive number
Methods / Member functions:	
RareNum( )	: default constructor to initialize the data member with the legal initial value.
void getnum( )	: accept a positive number from the user and store it to n.
int reverse(int)	: return the reverse of the number n.
void show( )	: check the number is a Rare number or not.

Specify the class **RareNum,** giving details of the **constructor(), void getnum(), int reverse()** and **void show( )**. Also, define a **main( )** function to create an object and call the functions accordingly to enable the task.

Question 7.

An identity matrix is a matrix whose all the elements are the same. The Identity matrix can be a square or non-square matrix. For example, the matrices A is an identity matrix.

1	1	1	1
1	1	1	1
1	1	1	1
1	1	1	1

Design a class **IdnMat** to check if the matrix is an Identity matrix or not an Identity matrix. Some of the members of the class are given below :

Class name	**: IdnMat**
Data members/instance variables:	
a[ ][ ]	: to store integer elements
m	: to store the number of rows
n	: to store the number of columns
Member functions/methods:	
IdnMat(int mm, int nn)	: parameterized constructor to initialize the data members **m = mm and n = nn**
void readArray( )	: to enter elements in the array from the user.
boolean check( )	: checks if the matrix is an Identity matrix or not and accordingly returns true or false
void print( )	: displays the array elements along with the message - an Identity Matrix or not an Identity Matrix.

Define the class **IdnMat** giving details of the **constructor( ), void readarray( ), boolean check( ) and void print( ).** Define the **main( )** function to create an object and call the functions accordingly
to enable the task.

Question 8.
Design a class Change to perform string-related operations. The first letter of each word will be changed
to uppercase and the rest will be in lowercase. Print the new string along with the length of each word separately. The details of the class are given below:

Class name : Change
Data Members/instance variables:
str : stores the string

Methods / Member functions:
Change ( ) : default constructor to initialize data members with legal initial values.
void readstr( ) : to store the string in str from the user.
String caseconvert (String W): converts the case of string W as mentioned above and returns the same.
void recchange ( ) : extracts each word from the string and makes the new string and displays the new string in the given format:

India	**Is**	**Great**
5	**2**	**5**

Specify the class **Change**, giving details of the **constructor ( ), caseconvert ( ), void recchange (int) and void display ( ).** Define the **main ( )** function to create an object and call the functions accordingly to enable the above change in the given word.

SECTION – C

Question 9.
Create two classes named Library and Issue. The class library will have the following structure to store detail of the books. Another class Issue will inherit class Library purchase that will store the number of days late in returning the book and fine calculated.

Class name : Library
Member data:
bookno as Integer to store book number.
Bookname as String to store the name of the book.
Studentname as String to store the name of the student.
Member function:
Library(….) parameterized constructor to give initial values to member data
void display( ) to display the member data

Class name : Issue
Member data:
daysLate : as Integer to store the number of days late to return the book.
fine : as Double to store the fine calculated.

Member function

Issue(….)	parameterized constructor to give initial values to the Super class data members
void return_book( )	to receive date of return of the book from student and compute the fine if no. of days is late according to the given rule otherwise no fine. Number of days late * 0.5
void display( )	to display the detail of the student issued the book along with total amount as fine (if applicable)

Assume that the super class ***Library*** *has been defined*. Using the **concept of inheritance**, specify the class **Issue** giving details of the **constructor**, **void return_book( )** and **void display( )**. **The super class, main function and algorithm need NOT be written**.

Question 10.

A Stack is a linear data structure in which the operations are performed based on LIFO (Last In First Out). Define a class **Stack** with the following details:

Class name : **Stack**

Data member/instance variable:

dat[ ]	:	array to hold the integer elements
cap	:	stores the maximum capacity of the queue
top	:	to point the index of the top

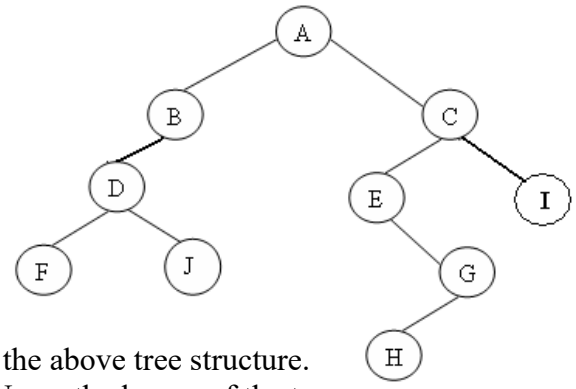
Member functions/methods:

Stack(int max)	:	constructor to initialize the data member cap = max, top=-1 and create the integer array
void push(int v)	:	to add integers from the rear index if possible else display the message("Stack full")
int pop( )	:	to remove and return elements from top, if any, else returns -999
void display()	:	to display elements of the stack

Specify the class **Stack** giving the details of **void push(int)** and **int pop( )**. Assume that the other functions have been defined. **The main( ) function and algorithm need NOT be written.**

Question 11.

Given a following binary tree, answer the following:



(i) Write down the preorder and post order traversal of the above tree structure.

(ii) What is the condition for a node to be a leaf node? Name the leaves of the tree.

(iii) When a tree is said to be a binary tree? Identify the above tree as complete binary tree or not.

(iv) What is height of a tree? Write down the height of the above given tree?

(v) How many nodes required to be added to the tree to make it a strictly binary tree? Show the tree.