# ISC Paper 2019
# Computer Science

Maximum Marks: 70
Time allowed: 3 hours

- Candidates are allowed additional 15 minutes for only reading the paper. They must NOT start writing during this time.
- Answer all questions in Part-I (compulsory) and six questions from Part-II, choosing two questions from Section-A, two from Section-B and two from Section-C.
- All working, including rough work, should be done on the same sheet as the rest of the answer.
- The intended marks for questions or parts of questions are given in brackets [ ].

## Part – I (20 Marks)
## Answer all questions.

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

**Question 1.**
(a) Name and draw the logic gate represented by the following truth table, where A and B are inputs and X is the output. [1]

| A | B | X |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

(b) Write the canonical POS expression of: $F(P, Q) = \Pi(0, 2)$ [1]
(c) Find the dual of: X.Y+X.Y' = X + 0 [1]
(d) If F(A, B, C) = A'.B'.C' + A'.B.C' then find F' using De Morgan's Law. [1]
(e) If A = "It is cloudy" and B = "It is raining", then write the proposition for [1]
(i) Contrapositive
(ii) Converse

**Answer:**
(a) XORgate



(b) $(A + B).(A' + B)$
(c) $(X + Y).(X + Y') = X.1$
(d) $(AB'C + ABC')$ $(\because x' + y' = x'.y')$
$= (AB'C')'. (A'.B.C')'$ $(\because (x'.y') = x' + y')$
$= ((A'B')' + (C)'. ((A'B)' + (C')')$
$= ((A')' + (B')') + C).((A')' + (B)' + C)$
$= (A + B + C).(A + B' + C)$
$= AA + AB' + AC + BA + BB' + BC + CA + CB' + CC$ $(\because x.x = x)$
$= A + AB' + C + AC + AB + B'C + BC$
(e) (i) If it is not raining then it is not cloudy.
(ii) If it is raining then it is cloudy.

**Question 2.**
(a) What is an Interface? How is it different from a class? [2]
(b) A matrix ARR[-4 ..... 6, 3 ....... 8] is stored in the memory with each element requiring 4 bytes of storage. If the base address is 1430, find the address of ARR[3] [6] when the matrix is stored in Row Major Wise. [2]
(c) Convert the following infix notation to postfix form: [2]
$(A + B * C) - (E * F / H) + J$
(d) Compare the two complexities O(n2) and O(2n) and state which is better and why. [2]
(e) State the difference between internal nodes and external nodes of a binary tree structure. [2]

**Answer:**
(a) An interface is just like Java Class, but it only has static constants and abstract method. Java uses Interface to implement multiple inheritances.
An interface is syntactically similar to a class, but it lacks in field declaration and the methods inside an interface do not have any implementation.
A class can be instantiated but an interface not.

(b) Given, Address of ARR[3] [6] = 1430
According to row-major,
Address $= B + w(n(p - L_1) + (q - L_2)$
B = 1430, w = 4 bytes
p = 3, q = 6, $U_1$ = 6, $U_2$ = 8, $L_1$ = -4, $L_2$ = 3
Number of columns n = $U_2 - L_2 + 1$
$\Rightarrow n = 8 - 3 + 1 = 6$

Address of ARR[3][6] = 1430 + 4(6 (3 − (-4))+ (6 − 3))
= 1430 + 4(6 × 7 + 3)
= 1430 + 180
= 1610

(c) (A + B * C) − (E * F / H) + J
= (A + (BC*)) − ((EF*) / H) + J
= (ABC *+) − (EF * H/) + J
= (ABC *+ EF * H/-) + J
= ABC*+ EF * H / − J*

(d) $O(n^2)$ is better than $O(2^n)$
Let us see with the help of an example.
Suppose n = 8
$n^2 = 8^2 = 64$
$2^n = 2^8 = 256$
Therefore, the complexity in $2^n$ is higher than $n^2$. If n increases, $2^n$ increases much more than $n^2$. Therefore, the time complexity is $O(n^2)$ is better than $O(2^n)$.

(e) Internal nodes are not leaf nodes whereas external nodes are leaf nodes.

**Question 3.**
The following function Mystery( ) is a part of some class. What will the function Mystery( ) return when the value of num=43629, x=3 and y=4 respectively? Show the dry run/working. [5]

```
int Mystery (int num, int x, int y)
{
if(num<10)
return num;
else
{
int z = num % 10;
if(z%2 == 0)
return z*x + Mystery (num/10, x, y);
else
return z*y + Mystery(num/10, x, y);
}
}
```

**Answer:**
Given n = 43629, x = 3, y = 4

| Step | num | x | y | num < 10 | z | z % 2 | return |
|------|------|---|---|----------|---|-------|--------|
| 1. | 43629 | 3 | 4 | false | 9 | 1 | 36 + Mystery(4362, 5, 4) |
| 2. | 4362 | 3 | 4 | false | 2 | 0 | 6 + Mystery(436 + 3, 4) |
| 3. | 436 | 3 | 4 | false | 6 | 0 | 18 + Mystery(43, 3, 4) |
| 4. | 43 | 3 | 4 | false | 3 | 1 | 12 + Mystery(4, 3, 4) |
| 5. | 4 | 3 | 4 | | | | |

Step 5 returns 4
Step 4 returns 12 + 4 = 16
Step 3 returns 18 + 16 = 34
Step 2 returns 6 + 34 = 40
Step 1 returns 36 + 40 = 76

# 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 4.**
(a) Given the Boolean function F(A, B, C, D) = Σ (0, 2, 3, 4, 5, 8, 10, 11, 12, 13).
(i) Reduce the above expression by using 4-variable Karnaugh map, showing the various groups (i.e. octal, quads and pairs). [4]
(ii) Draw the logic gate diagram for the reduced expression using only NAND gates. Assume that the variables and their complements are available as inputs. [1]
(b) Given the Boolean function : F(P, Q, R, S) = π (0, 1, 2, 8, 9, 11, 13, 15).
(i) Reduce the above expression by using a 4-variable Karnaugh map, showing the various groups (i.e, octal, quads and pairs). [4]
(ii) Draw the logic gate diagram for the reduced expression using only NOR gates. Assume that the variables and their complements are available as inputs. [1]

**Answer:**
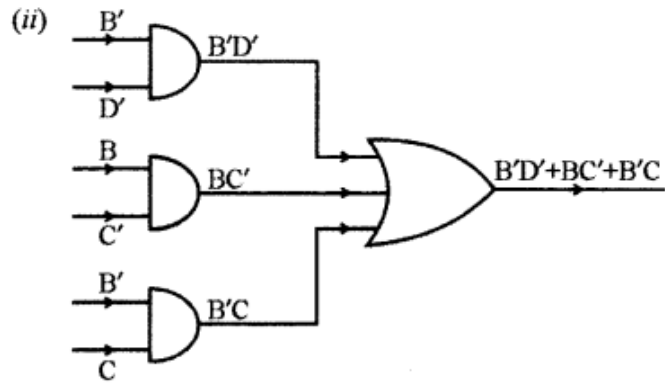(a) (i) $F(A, B, C, D) = \Sigma (0, 2, 3, 4, 5, 8, 10, 11, 12, 13)$

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

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)



Output: $B'D' + BC' + B'C$

(b) (i) $F(P, Q, R, S) = \pi(0, 1, 2, 8, 9, 11, 13, 15)$

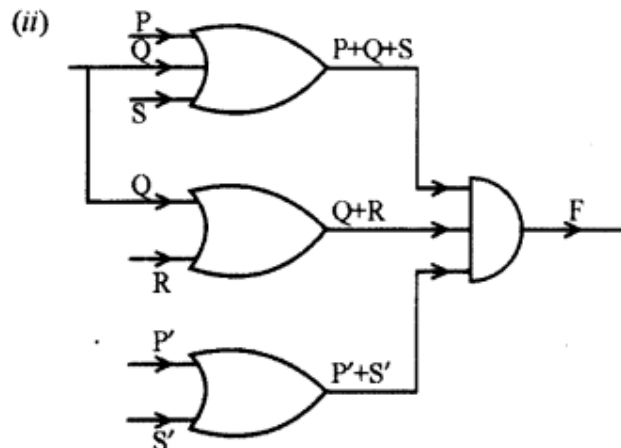| | R+S | R+S' | R'+S' | R'+S |
|---|---|---|---|---|
| P+Q | 0 — **0** | 1 — **0** | 3 | 2 — **0** |
| P+Q' | 4 **1** | 5 **1** | 7 **1** | 6 **1** |
| P'+Q' | 12 **1** | 13 **0** | 15 **0** | 14 **1** |
| P'+Q | 8 **0** | 9 **0** | 11 **0** | 10 **1** |

Quad 1 : $(m_0\, m_1\, m_8\, m_9) = Q + R$

Quad 2 : $(m_{13}\, m_{15}\, m_9\, m_{11}) = P' + S'$
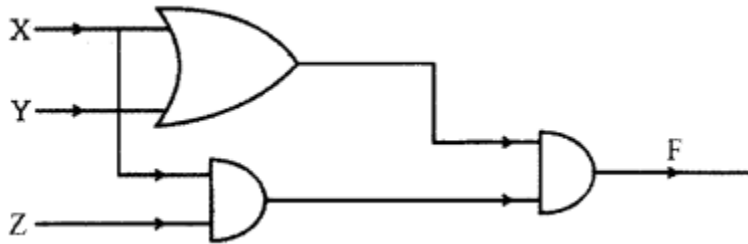
Pair : $(m_0\, m_2) = P + Q + S$

Hence,

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

(ii)

**Question 5.**
(a) How is a decoder different from a multiplexer? Write the truth table and draw the logic circuit diagram for a 3 to 8 decoder and explain its working. [5]
(b) From the logic circuit diagram given below, derive the Boolean expression and simplify it to show that it represents a logic gate. Name and draw the logic gate. [3]



(c) Using a truth table, state whether the following proposition is a Tautology, Contradiction or Contingency: [2]
~(P =>Q)<=>(~PvQ)

**Answer:**
(a) A multiplexer is a device which consists of multiple input channels through a single line while decoder consists of multiple inputs passing through multiple outputs. Multiplexer converts inputs from unary codes (initial) to binary codes while decoder converts binary codes to inputs.

| Inputs | | | Outputs | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| X | Y | Z | $D_0$ | $D_1$ | $D_2$ | $D_3$ | $D_4$ | $D_5$ | $D_6$ | $D_7$ |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
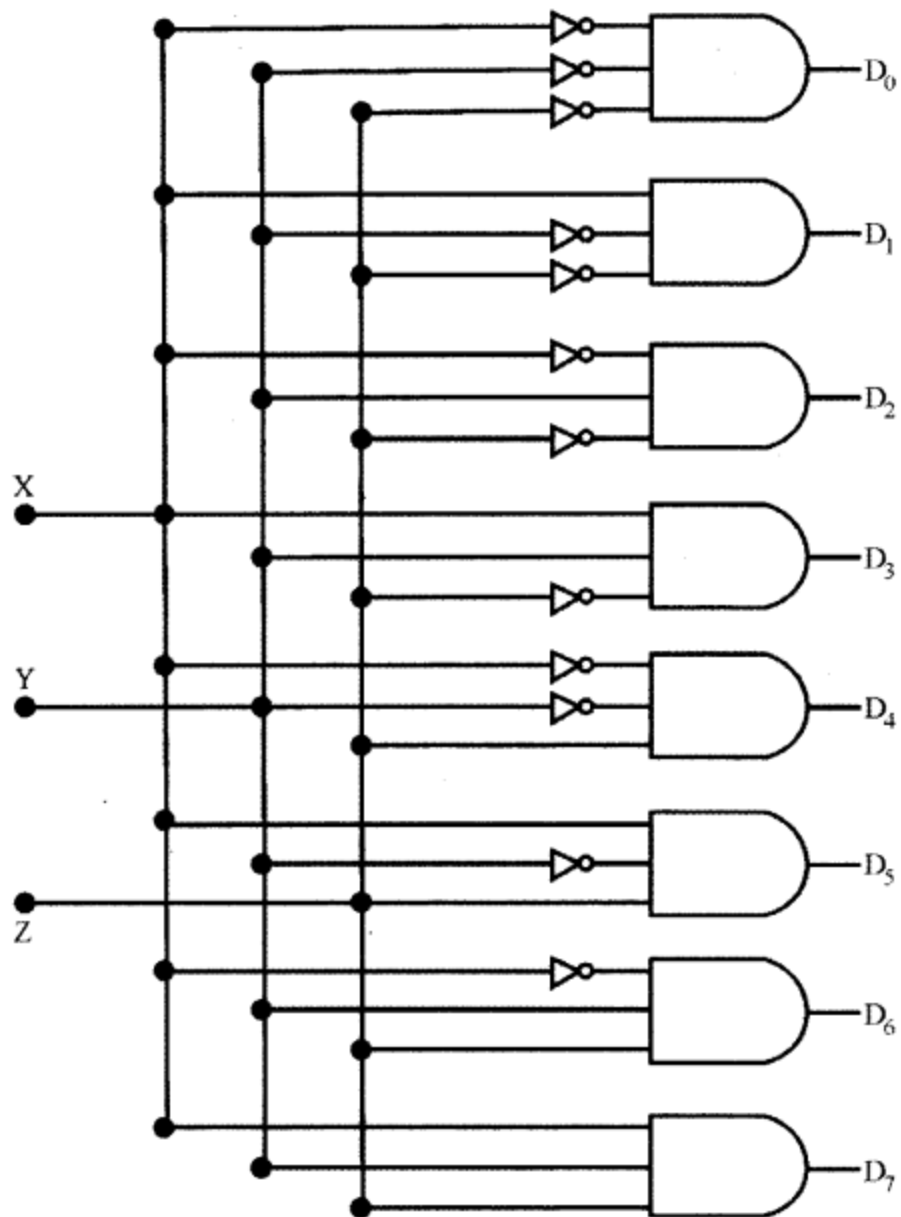
The circuit is designed with AND and NAND logic gates. It takes 3 binary inputs and activates one of the eight outputs. 3 to 8 line decoder circuit is also called as binary to an octal decoder.

(b)



(c)
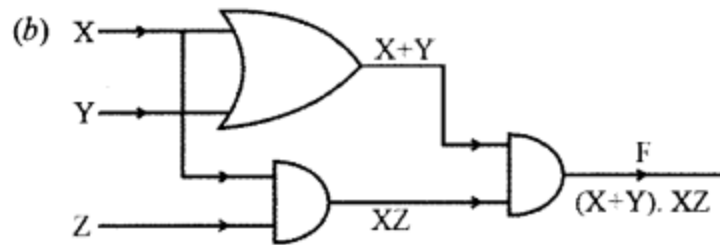
| P | Q | P⇒Q | ~(P⇒Q) | ~P | ~PvQ |
|---|---|-----|--------|----|------|
| 0 | 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 1 |
| 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 | 1 |

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

**Question 6.**
(a) The owner of a company pays the bonus to his salesmen as per the criteria are given below: [5]
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 | Working 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).
(b) What is a half adder? Write the truth table and derive an SOP expression for sum and carry for a half adder. [3]
(c) Simplify the following expression, using Boolean laws: [2]
(X + Z).(X.Y + Y.Z') + X.Z + Y

**Answer:**

(a)

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

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

(b) The half adder is an example of a simple, functional digital circuit built from two logic gates. The half adder adds two one-bit binary numbers (AB). The output is the sum of the two bits (S) and the carry (C).

The truth table for a half-adder is:

| x | y | C | S |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 |

'x' and 'y' are the two inputs, and S (Sum) and C (Carry) are the two outputs.
The Carry output is 'O' unless both the inputs are 1.
'S' represents the least significant bit of the sum.
The simplified sum of products (SOP) expressions is:
$S = x'y + xy'$, $C = xy$
(c) $(X + Z).(X.Y + Y.Z') + X.Z + Y$
$= (X + Z).(Y.Z') + X.Z + Y$ (a.b = 0)
$= X.Y.Z' + Z.Z' + X.Z + Y$ (a.a' = 0) and (a.c = 0)

= 0.Z' + 0 + 0 + Y
= Y

# 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 7.**
Design a class ArmNum to check if a given number is an Armstrong number or not. [A number is said to be Armstrong if sum of its digits raised to the power of length of the number is equal to the number] [10]
Example:
$371 = 3^3 + 7^3 + 1^3$
$1634 = 1^4 + 6^4 + 3^4 + 4^4$
$54748 = 5^5 + 4^5 + 7^5 + 4^5 + 8^5$
Thus, 371, 1634 and 54748 are all examples of Armstrong numbers.
Some of the members of the class are given below:
Class name: ArmNum
Data members/instance variables:
n: to store the number
1: to store the length of the number
Methods/Member functions:
ArmNum (int nn): parameterized constructor to initialize the data member n = nn
int sum_pow(int i): returns the sum of each digit raised to the power of the length of the number using recursive technique eg., 34 will return 32 + 42 (as the length of the number is 2)
void isArmstrong(): checks whether the given number is an Armstrong number by invoking the function sum_pow () and displays the result with an appropriate message.
Specify the class ArmNum giving details of the constructor( ), int sum_pow(int) and void isArmstrong( ). Define a main() function to create an object and call the functions accordingly to enable the task.

**Answer:**

import java.io.*;
import javautil.*;
class ArmNum
{

```
private int n;
private int 1;
public ArmNum(int num)
{
n = num;
i = 0;
for(int i = n; i! = 0; i/= 10)
i++;
}
public int sumPow(int i)
{ if(i < 10)
return (int)Math.pow(i, 1);
return (int)Math.pow(i % 10, 1) + sumPow(i/10);
}
public void isArmstrong()
{
if(n == sumPow(n))
System.out.println(n+" is an Armstrong number.");
else
System.out.println(n+ " is not an Armstrong number.");
}
public static void main(String args[])throws IOException{
Scanner sc = new Scanner(System.in);
System.out.print("N =");
int num = sc.nextInt();
ArmNum obj = new ArmNum(num);
obj.isArmstrong();
}
}
```

**Question 8.**
Design a class MatRev to reverse each element of a matrix. [10]
Example:

| 72 | 371 | 5 |
|----|-----|---|
| 12 | 6 | 426 |
| 5 | 123 | 94 |

becomes

| 27 | 173 | 5 |
|----|-----|---|
| 21 | 6 | 624 |
| 5 | 321 | 49 |

Some of the members of the class are given below:
Class name: MatRev
Data members/instance variables:
arr[][] : to store integer elements
m: to store the number of rows
n: to store the number of columns
Member functions/methods:

MatRev(int mm, int nn): parameterized constructor to initialise the data members m = mm and n = nn
void fillarray(): to enter elements in the array
int reverse(int x): returns the reverse of the number x
void revMat(MatRev P): reverses each element of the array of the parameterized object and stores it in the array of the current object
void show(): displays the array elements in matrix form
Define the class MatRev giving details of the constructor ( ), void fillarray (), int reverse(int), void revMat(MatRev) and void show(). Define the main () function to create objects and call the functions accordingly to enable the task.

**Answer:**

```
import java.io.*;
import java.util.*;
class MatRey{
private int arr[][];
private int m;
private int n;
public MatRev(int mm, int nn) {
m=mm;
n = nn;
arr=newint[m][n];
}
public void fillArray( )throws IOException{
Scanner sc = new Scanner(System.in);
System.out.println("Enter matrix elements::");
for(int i = 0; i < m; i++) {
for(int j = 0; j < n; j++) {
arr[i][j] = sc.nextInt();
}
}
}
public int reverse(int x) {
int rev = 0;
for(int i = x; i != 0; i /= 10)
rev = rev * 10 + i % 10;
return rev;
}
public void revMat(MatRev p) {
for(int i = 0; i < m; i++) {
for(int j = 0; j < n; j++) {
this.arr[i] [j] = reverse(p.arr[i] [j]);
}
```

```
}
}
public void show() {
for(int i = 0; i < m; i++) {
for(int j = 0; j < n; j++) {
System, out. print(arr[i][j] + "\t");
}
System.out.println();
}
}
public static void main(String args[])throws IOException{
Scanner sc = new Scanner(System.in);
System.out.print("Enter number of rows::");
int x = sc.nextInt();
System.out.print("Enter number of columns::");
int y = sc.nextInt();
MatRev obj 1 = new MatRev(x, y);
MatRev obj2 = new MatRev(x, y);
obj1.fillArray();
obj2.revMat(obj 1);
System.out.println("Original Matrix is::");
obj1.show();
System.out.println("Matrix with reversed elements");
obj2.show();
}
}
```

**Question 9.**
A class Rearrange has been defined to modify a word by bringing all the vowels in the word at the beginning followed by the consonants. [10]
Example:
ORIGINAL becomes OIIARGNL
Some of the members of the class are given below:
Class name: Rearrange
Data Member/instance variable:
wrd: to store a word
newwrd: to store the rearranged word
Member functions/methods:
Rearrange(): default constructor
void readword(): to accept the word in UPPER case
vow freq_vow_con(): finds the frequency of vowels and consonants in the word and displays them with an appropriate message
void arrange(): rearranges the word by bringing the vowels at the beginning followed by consonants

void display(): displays the original word along with the rearranged word
Specify the class Rearrange, giving the details of the constructor(), void readword(), void freq _vow_con(), void arrange() and void display(). Define the main() function to create an object and call the functions accordingly to enable the task.

**Answer:**

```
import java.io.*;
import java.util.*;
class Rearrange {
private String wrd;
private String newwrd;
public Rearrange() {
wrd = new String();
newwrd = new String();
}
public void readword() throws IOException {
Scanner sc = new Scanner(System.in);
System.out.print("Enter the word:");
wxd = sc.next();
}
public void freq_vow_con() {
wrd = wrd.toUpperCase();
int v = 0;
int c = 0;
forint i = 0; i < wrd.length(); i++) {
char ch = wrd. char At(i);
if(Character.isLetter(ch)) {
switch(ch) {
case 'A':
case'E':
case T:
case 'O':
case 'U':
v++;
break;
default:
C++;
}
}
}
System.out.println("Frequency of vowels: "+v);
System.out.println("Frequency of consonants: "+c);
}
```

```java
public void arrange() {
String v = new String();
String c = new String();
wrd = wrd.toUpperCase();
for(int i = 0; i < wrd. length(); i++) {
char ch = wrd.charAt(i);
if(Character.isLetter!ch)) {
switch(ch) {
case 'A':
case'E':
case T:
case 'O':
case 'U':
v += ch;
break;
default:
c + = ch;
}
}
}
newwrd = v + c;
}
public void display() {
System.out.println("Original word:" + wrd);
System.out.prfntln("Rearranged word:" + newwrd);
}
public static void main(String args[])throws IOException {
Rearrange obj = new Rearrange();
obj.readword();
obj.freq_vow_con();
obj.arrange();
obj.display();
}
}
```

## 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 10.**
A superclass Record contains names and marks of the students in two different single dimensional arrays. Define a subclass Highest to display the names of the students obtaining the highest mark [5]
The details of the members of both classes are given below:
Class name: Record
Data member/instance variable:
n[] : array to store names
m[]: array to store marks
size: to store the number of students
Member functions/methods:
Record(int cap): parameterized constructor to initialize the data member
size = cap
void readarray() : to enter elements in both the arrays
void display() : displays the array elements
Class name: Highest
Data member/instance variable:
ind: to store the index
Member functions/methods:
Highest(…): parameterized constructor to initialize the data members of both the classes
void find(): finds the index of the student obtaining the highest mark and assign it to 'ind'
void display(): displays the array elements along with the names and marks of the students who have obtained the highest mark

Assume that the superclass Record has been defined. Using the concept of inheritance, specify the class Highest giving the details of the constructor(…), void find() and void display().
The superclass, main function and algorithm need NOT be written.

**Answer:**

```
class Record
{
protected String n[];
protected int m[];
protected int size;
public Recordfint cap) {
}
public void readarray() {
}
public void display() {
}
```

```
}
class Highest extends Record {
private int ind;
public Highest(int cap) {
super(cap)
}
public void find() {
ind = 0;
for (int i = 0; i < size; i++) { if(m[i]>m[ind]){
ind = i;
}
}
}
public void display() {
super.display();
System.out.println("Highest marks are::" +m[ind]);
System.out.println("Students who score the highest marks are::");
for (int i = 0; i < size; i++) }
if(m[i] == m[ind]) {
System.out.println(n[i]);
}
}
}
}
```

## Question 11.

A linear data structure enables the user to add an address from rear end and remove address from front. Define a class Diary with the following details :

Class name: Diary

Data members/instance variables:

Q[]: array to store the addresses

size: stores the maximum capacity of the array

start: to point the index of the front end

end: to point the index of the rear end

Member functions:

Diary(int max): constructor to initialize the data member size = max, start=0 and end=0

void pushadd(String n): to add the address in the diary from the rear end if possible, otherwise display the message "NO SPACE"

String popadd(): removes and returns the address from the front end of the diary if any, else returns "?????"

void show (): displays all the addresses in the diary

(a) Specify the class Diary giving details of the functions void pushadd(String) and String popadd(). Assume that the other functions have been defined. [4]

The main function and algorithm need NOT be written.

(b) Name the entity used in the above data structure arrangement. [1]

**Answer:**

(a) class Diary {
public void pushAdd(String n) {
if(Q[0].equals(""))
Q[0] = n;
else if(end + 1 < size) Q[end + 1] = n;
else
System.out.println("NO SPACE");
}
public String popadd() {
if(start -1 >= 0)
return Q[start--];
else
return "?????";
}
}
(b) Queue

**Question 12.**
(a) 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 num;
Node next;
}
Write an Algorithm OR a Method to find and display the sum of even integers from an existing linked list.
The method declaration is as follows:
void SumEvenNode(Node str)
(b) Answer the following questions from the diagram of a Binary Tree given below:
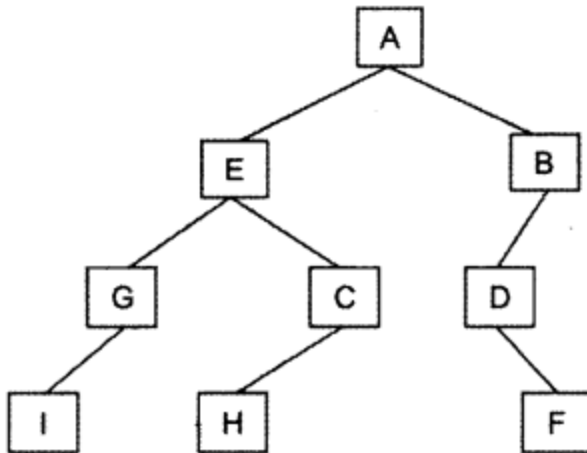
(i) Write the pre-order traversal of the above tree structure. [1]
(ii) State the size of the tree. [1]
(iii) Name the siblings of the nodes E and G [1]

**Answer:**

(a) void sumEvenNode(Node str) {
if(str = null)
return 0;
else if(str.num % 2 == 0)
return str.num + sumEvenNode(str.next);
else
return 0 + sumEvenNode(str.next);
(b) (i) A → E → G → I → C → H → B → D → F
(ii) The size of the tree is 4.
(iii) Sibling of E is B.
Sibling of G is C.