ISC SEMESTER 2 EXAMINATION
SAMPLE PAPER - 2
COMPUTER SCIENCE PAPER 1 (THEORY)

Maximum Marks: 35

Time allowed: One and a half hour

*Candidates are allowed an additional **10 minutes** for **only** reading the paper.*

*They must **NOT** start writing during this time.*

---

*Answer **all** questions in **Section A, Section B** and **Section C**.*

While answering questions in Sections A and B, working and reasoning may be indicated briefly.

All working, including rough work, should be done on the same sheet as the rest of the answer.

Section-A

Question 1.

(i) The keyword used by a member function to define it as a class member:

(a) import (b) new (c) static (d) void

(ii) One of the forms in Java to exhibit polymorphism is:

(a) inheritance (b) abstraction (c) overriding (d) overloading

(iii)
```
void repeat(int n)
{
    repeat(n/2);
    System.out.print(n%2);
    if(n==0)
        return;
}
```
With reference to the program code given above, what will the function **repeat( )** returns when the value of n=56 ?

(a) 100100 (b) 11 (c) 100 (d) 1100

(iv) Write the statement in Java to check the word "HOLIDAY" is ending with "DAY" or not.

(v) What is the logic of Pre-order traversal in a Binary Tree?

(vi) What is the output of the statement given below?

System.out.print("EXCELLENT".charAt("AUTOMATIC".lastIndexOf('A'));

(vii) Give one point of difference between stack and queue.

Section-B

Question 2.

Define:

(i) Binary Tree

(ii) Recursion

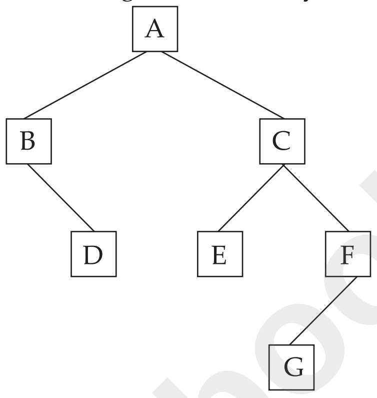
Question 3.

Convert the following infix notation to postfix notation:

A + B * C / D – E / F

Question 4.

Answer the following questions on the diagram of a Binary Tree given below:



(i) State the height and depth of the node F, if the root is at level 0.

(ii) Write the In order and pre order traversal of the above tree structure.

Section-C

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 5.

(i) Design a class **Check** which checks whether a word is both starting and ending with same vowel or not.

Example: AFRICA, AMERICA etc.

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

Class name	**:**	**Check**
Data members/instance variables:		
Wrd	:	stores a word
Len	:	to store the length of the word
Methods/Member functions:		
Check( )	:	default constructor
void acceptword( )	:	to accept the word in uppercase
boolean checkVowel ( )	:	checks and returns 'true' if the word is starting as well as ending with the same vowel otherwise returns 'false'.
void display( )	:	displays the word along with an appropriate message

Specify the class **Check** giving details of the **constructor, void acceptword( ), boolean checkVowel( )** and **void display( )**. Define the **main( )** function to create an object and call the functions accordingly to enable the task.

OR

(ii) Design a class **Change** which toggles a word by converting all vowels to next letter and all the consonants to previous letter present in the word.

Example: The word "Motivate" becomes "Lpsjubsf"

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

Class name	**:**	**Change**
Data members/instance variables:		
Str	:	stores a word
Newstr	:	stores the toggled word
Len	:	to store the length of the word

Methods/Member functions:

Change( )	:	default constructor
void readword( )	:	to accept the word
void swap( )	:	converts all the vowels to next alphabet and all the consonant to previous alphabet and store it in newstr.
void display( )	:	displays the original word along with the new word

Specify the class **Change** giving details of the **constructor, void readword( ), void swap( )** and **void display( )**. Define the **main( )** function to create an object and call the functions accordingly to enable the task.

Question 6.

(i) A class **Padovan** has been defined to generate the Padovan series 0, 1, 1, 1, 2, 2, 3, 4, 5, 7,……. (Padovan series is a set of numbers that starts with three fixed nos. 0,1,1 and then next no. will be the sum of first two, next the sum of next two and so on).

Some of the members of the class are given below:

Class name	:	**Padovan**
Data member/instance variable:		
start	:	integer to store the starting value
end	:	integer to store the ending value
Member functions/methods:		
Padovan( )	:	default constructor
void accept( )	:	to accept the starting and ending value
int padovan(int x)	:	return the nth term padovan no. using recursion
void display( )	:	displays the result with an appropriate Message

Specify the class **Padovan**, giving details of the **Constructor, void read( ), int padovan(int)**, and **void display( )**. Define the **main( )** function to create an object and call the functions accordingly to enable the task.

OR

(ii) A class **Prime** has been defined to check and print the number is Prime number or composite number. Some of the members of the class are given below:

Class name	:	**Prime**
Data member/instance variable:		
num	:	integer to store an integer
Member functions/methods:		
Prime( )	:	default constructor
void accept( )	:	to accept the number
int prime(int x)	:	check num as prime or not using **recursive technique** and return 1 or 0
void display( )	:	displays the result with an appropriate Message

Specify the class **Prime**, giving details of the **Constructor, void accept( ), int prime(int)**, and **void display( )**. Define the **main( )** function to create an object and call the functions accordingly to enable the task.

Question 7.

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 are 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 8.

A Queue is a linear data structure in which the operations are performed based on FIFO (First In First Out).

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

Class name	:	**Queue**
Data member/instance variable:		
dat[ ]	:	array to hold Alphabets
cap	:	stores the maximum capacity of the queue
front	:	to point the index of the front
rear	:	to point the index of the rear.
Member functions/methods:		
Queue(int max)	:	constructor to initialize the data member cap = max, front = rear = 0 and create the integer array
void enque(char v)	:	to add integers from the rear index if possible else display the message(“Queue full”)
char delque( )	:	to remove and return elements from front, if any, else returns ‘\u0000’
void display()	:	to display elements of the queue

Specify the class **Queue** giving the details of **void enque(char)** and **char delque( )**.

Assume that the other functions have been defined.

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

Answers

Section-A

Answer 1.

(i) (c) static

(ii) (d) overloading

(iii) (a) 100100

(iv) "HOLIDAY".endsWith("DAY");

(v) Pre-order traversal follows Root->Left->Right traversal in the tree.

(vi) L

(vii) Stack follows LIFO technique whereas queue follows FIFO technique.

Answer 2.

(i) Binary Tree is a data structure where each node can have maximum two sub nodes.

(ii) Recursion is a process in which a function calls itself within its body.

Answer 3.

ABC*D/+EF/–

Answer 4.

(i) Height of F = 1

Depth of F = 2

(ii) In-order: DBAECGF

Pre-order: ABDCEFG

Answer 5.

(i)
```
import java.util.Scanner;
class Check2
{
    String wrd;
    int len;

    public Check2( )
    {
        wrd="";
        len=0;
    }

    public void acceptword( )
    {
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter a word");
        wrd=sc.next( );
        len=wrd.length();
    }

    public boolean checkVowel( )
```

```
        {
            String vowel[]={“A”,”E”,”I”,”O”,”U”};
            for(int i=0;i<5;i++)
            {
                if(wrd.startsWith(vowel[i])&&wrd.endsWith(vowel[i]))
                    return true;
            }
            return false;
        }

        public void display( )
        {
            System.out.print(wrd);
            if(checkVowel( ))
                System.out.println(“ is starting and ending with same vowel”);
            else
                System.out.println(“ is not starting and ending with same vowel”);
        }

        public static void main(String ar[ ])
        {
            Check2 ob=new Check2( );
            ob.acceptword( );
            ob.display( );
        }
    }
```

(ii)
```
    import java.util.Scanner;
    class Toggle2
    {
        String str, newstr;
        int len;

        public Toggle2( )
        {
            str=newstr=””;
            len=0;
        }

        public void readword( )
        {
            Scanner sc=new Scanner(System.in);
            System.out.println(“Enter a word”);
            str=sc.next( );
            len=str.length( );
        }
        public void swap( )
        {
            for(int i=0; i<len;i++)
            {
                char c=str.charAt(i);
                if(c=='a'||c=='e'||c=='i'||c=='o'||c=='u'||c=='A'||c=='E'||c=='I'||c=='O'||c=='U')
                newstr+=(char)(c+1);
            else
                newstr+=(char)(c–1);
            }
        }

        public void display( )
        {
            System.out.println("Original string:"+str);
            System.out.println("New string:"+newstr);
        }

        public static void main(String ar[ ])
        {
            Toggle2 Ob=new Toggle2( );
             Ob.readword( );
            Ob.swap( );
            Ob.display( );
        }
    }
```

Answer 6.

(i)
```
import java.util.*;
class Padovan
{
    int start, end;

    public Padovan ( )
    {
        start=end=0;
    }

    public void read( )
    {
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter start and end value");
        start=sc.nextInt( );
        end=sc.nextInt( );
    }
```

```
        public int padovan(int a)
        {
            if(a==0)
                return 0;
            else if(a==1 || a==2)
                return 1;
            else
                return padovan(a-2)+padovan(a-3);
        }

        public void display ()
        {
            for(int i=start; i<end; i++)
            {
                int p=padovan(i);
                if(p>=start && p<=end)
                System.out.print(padovan(i)+" ");
            }
        }

        public static void main(String ar[ ])
        {
            Padovan Ob=new Padovan( );
            Ob.read( );
            Ob.display( );
        }
    }
```

(ii)
```
    import java.util.*;
    class Prime
    {
        int num;
        void accept( )
        {
            Scanner sc= new Scanner(System.in);
            System.out.println("Enter a number");
            num=sc.nextInt( );
        }

            int prime(int x)
            {
                if(num==1)
                    return 0;
                else if(num==2)
                    return 1;
                else if(num%x==0)
                    return 0;
                else if(x>num/2)
                    return 1;
                else
                    return prime(x+1);
            }

            void display( )
            {
                if(prime(2)==1)
                    System.out.println("Prime no");
                else
                    System.out.println("Non-prime no");
            }

            static void main( )
            {
                Prime ob=new Prime( );
                ob.accept( );
                ob.display( );
            }
        }
```

Answer 7.

```
        class Issue extends Library
        {
            int daysLate;
            double fine;

            public Issue(int bno, String bname, String Sname, int dl)
            {
                super(bno, bname, Sname);
                daysLate=dl;
                fine=0;
            }

            public void return_book( )
            {
                fine=daysLate*0.5;
            }

            public void display( )
            {
                super.display( );
                System.out.println("Days late:"+daysLate);
                System.out.println("Fine:"+fine);
            }
        }
```

Answer 8.

```
import java.util.*;
class Queue
{
    int dat[ ];
    int cap, front, rear;

    public Queue( int max)
    {
        cap=max;
        dat = new int[cap];
        front=rear=0;
    }

    public void enque(int v)
    {
        if(rear==cap–1)
        {
            System.out.println("Queue full");
            return;
        }
        dat[rear++]=v;
    }

    public int delque( )
    {
        if(front==rear)
        {
            System.out.println("Queue empty");
            return -999;
        }
        else
            return dat[front++];
    }

    public void display( )
    {
        for(int i=front; i<rear; i++)
        {
            System.out.print(dat[i]+ " ");
        }
    }
}
```

❑❑