EX NO: 1A	IMPLEMENTATION OF SEQUENTIAL SEARCH
DATE:	

AIM:

To develop a java program to input a sequence of integers and check whether a user-given integer is present in the given sequence

ALGORITHM:

Step 1: Start of the program

Step 2: Create an integer array and input the elements

Step 3: Now,input the element to be searched

Step 4: Check through the array using for loop from 0 to size of array -1 , and check if the current element is equal to the searched element

Step 5: If the element is found in the array,then print the position of the element

Step 6: Stop the program

SOURCE CODE:

```
import java.util.*;

public class Exp1A

{

   public int sequentialsearch(int arr[],int s)

  {

      for(int i=0;i<arr.length;i++)

{

         if(arr[i]==s)

           {

            return i;

           }

         }

      return -1;
```

```
    }


    public void printArray(int arr[])
     {
        for(int i=0;i<arr.length;i++)
            {
            System.out.print(arr[i]+" ");
            }
      }
    public static void main(String args[])
     {
            int size;
            Scanner in = new Scanner(System.in);
            Exp1A ob = new Exp1A();
            System.out.println("Enter size: ");
            size = in.nextInt();
            int[] arr = new int[size];
            System.out.println("Enter Data: ");
            for(int i=0;i<size;i++)
 {
               arr[i] = in.nextInt();
              }
  System.out.println("\nEnter element to be searched: ");
            int s = in.nextInt();
            int idx=ob.sequentialsearch(arr,s);
            System.out.println("The element you are looking for "+(idx>0?("is at index "+idx):("is not in
the array")));
            in.close();
        }
  }
```

OUTPUT:

RESULT:

The java program to input a sequence of integers and check whether a user-given integer is present in the given sequence has been verified and executed successfully.

EX NO: 1B	IMPLEMENTATION OF BINARY SEARCH
DATE:	

AIM:

To develop a java program to arrange the sequence of n numbers in an ascending or descending order and find the user defined number in the given sequence of numbers using sequential search.

ALGORITHM:

Step 1: Start of thrthe program

Step 2: Create an integer array arr and input the elements

Step 3: For i=1 to (size of array -1):

Step 4: Set j=i-1 and key=arr[i]

Step 5: While j>=0 and arr[j]>key
Step 6: Move the elements of the array to the right by arr[j+1]=arr[j] and decrement j by 1

Step 7: Endwhile

Step 8: Set arr[j+1]=key

Step 9: Input the element to be searched.

Step 10: Create a function binarysearch(int arr,int first,int last,int search) and call it in the main function

Step 11: Check if the middle element of the array is the element to be found with mid = (first+last)/2

Step 12: If the element is found in the array,then print the position of the element

Step 13 : Else if the element is smaller than arr[mid] then recursively call the function binarysearch(arr,first,mid-1,search)

Step 14: Else if the element is larger,call binarysearch(arr,mid+1,last,search)

Step 15: Else,return -1 indicating the element is not present in the list

Step 16: Display the results

Step 17: Stop the program

SOURCE CODE:

```
import java.util.Scanner;
public class Exp1B
{
	public int binarysearch(int arr[],int first,int last,int s)
	{
		int mid = (first+last)/2;
		if(arr[mid]==s)
		{
			return mid;
		}
		else if(arr[mid]<s)
		{
		  return binarysearch(arr,mid+1,last,s);
		}
		else if(arr[mid]>s)
		{
		   return binarysearch(arr,first,mid-1,s);
		}
		else
		{
		   return -1;
		}
	}

	public void insertionsort(int arr[])
	{
		int temp;
		for(int i=1;i<arr.length;i++)
		{
```

```
                temp=arr[i];
                int j=i-1;
                while(j>=0 && temp<arr[j])
                {
                  arr[j+1]=arr[j];
                  j--;

                }
                arr[j+1]=temp;
        }
}
public void printArray(int arr[])
{
        for(int i=0;i<arr.length;i++)
        {
                System.out.print(arr[i]+" ");
        }

}
public static void main(String args[])
{
        int size;
        Scanner in = new Scanner(System.in);
        Exp1B ob = new Exp1B();
        System.out.println("Enter size: ");
        size = in.nextInt();
        int[] arr = new int[size];
        System.out.println("Enter Data: ");
        for(int i=0;i<size;i++) {
                arr[i] = in.nextInt();
```

```
                }
                System.out.println("The entered array: ");
                ob.printArray(arr);
                System.out.println("\nThe array after sorting: ");
                ob.insertionsort(arr);
                ob.printArray(arr);
                System.out.println("\nEnter element to be searched: ");
                int s = in.nextInt();
                int idx=ob.binarysearch(arr,0,arr.length-1,s);
                System.out.println("The element you are looking for "+(idx>0?("is at index
"+idx):("is not in the array")));
                in.close();
        }
}
```

OUTPUT:

RESULT:

The java program to arrange the sequence of n numbers in an ascending or descending order and find the user defined number in the given sequence of numbers using sequential search.

EX NO: 2	IMPLEMENTATION OF STACK AND QUEUE USING CLASSES AND OBJECTS
DATE:	

AIM:

To develop a java program to implement Queue and Stack data structures using classes and objects

ALGORITHM:

Step 1: Start of the program

Step 2: Create a class Queue and declare the variables front,rear and arr of a maximum size

Step 3: Create functions to implement the data structure:
Queue(),isEmpty(),enqueue(),dequeue(),peek()

Step 4: A queue() is used to initialize front and rear to -1

Step 5: A isEmpty() is used to return the result of front==-1 and rear==-1 to check if the queue is empty

Step 6: A enqueue(int data) is used to insert a new element into the queue by incrementing the value of rear and setting arr[rear]= data

Step 7: A dequeue() is used to remove the element at the front of the queue by incrementing the value of front

Step 8: A peek() is used to get the front of the queue without deleting the front element

Step 9: Create a class Stack and declare the variables top and arr of a maximum size

Step 10: Create functions to implement the data structure:

Stack(),isEmpty(),pop(),push(),peek()

Step 11: Stack() is used to initialize the top to -1

Step 12: A isEmpty() is used to return the result of top==-1 to check if the stack is empty

Step 13: A push(int data) is used to insert an element to the stack by incrementing top by 1 and setting arr[top]=data

Step 14: A pop() is used to remove the top element of the stack by decrementing the value of top

Step 15: A peek() is used to get the top of the stack without deleting the top element

Step 17: Create a main class to create instances of the Queue and Stack classes and call the functions with user-defined data

Step 18: Stop of the program.

SOURCE CODE:

```
Import java.util.*;
public class Queue
 {
   int front,rear;
   static final int max= 1000;
   int[] arr = new int[max];

   Queue()
    {
      front=rear=-1;
    }
   boolean isEmpty()
    {
      return front>rear||front<0;
    }
   boolean enqueue(int data)
   {
      rear++;
      if(rear>max)
           {
          System.out.println("Queue is Full");
          return false;
           }
    else if(isEmpty())
          {
          front=0;
          }
   arr[rear]=data;
   return true;
```

```
    }

    int dequeue()
    {
        if(!isEmpty())
         {
            return arr[front++];
         }
        System.out.println("Queue is Empty");
        return -1;
    }

    int peek()
    {
        return arr[front];
    }
}
public class Stack
{
 static final int max = 1000;
 int[] arr = new int[max];
 int top;

 Stack()
  {
     top=-1;
   }

 boolean isEmpty() {
    return(top<0);
 }
```

```
boolean push(int data)
 {
    top++;
    if(top<max)
    {
        arr[top] = data;
        return true;
    }
   else
   {
        System.out.println("Stack Overflow");
        return false;
    }
}

int pop()
{
    int res;
    if(top>=0)
      {
        res=arr[top];
        top--;
      }
    else
     {
        System.out.println("Stack Underflow");
        res=-1;
      }
    return res;
}
```

```
 int peek()
  {
     return arr[top];
  }

 boolean search(int s)
 {
     int temp = top;
     while(temp>=0)
       {
        if(arr[temp]== s)
            {
            return true;
            }
        temp--;
       }
     return false;
  }
}
public class Exp2
  {
    public static void main(String args[])
     {
        Scanner in = new Scanner(System.in);
        Stack st = new Stack();
        Queue q = new Queue();
        System.out.println("Enter the size of stack: ");
        int stsize= in.nextInt();
        System.out.println("Enter the data of stack: ");
        for(int i=0;i<stsize;i++)
         {
            st.push(in.nextInt());
```

```
        }
      System.out.println("stack: ");
      while(!st.isEmpty())
       {
         System.out.println(st.pop());
        }
      System.out.println("Enter the size of queue: ");
      int qsize= in.nextInt();
      System.out.println("Enter the data of queue: ");
      for(int i=0;i<qsize;i++)
       {
         q.enqueue(in.nextInt());
       }
      System.out.println("stack: ");
      while(!q.isEmpty())
      {
         System.out.println(q.dequeue());
      }
   }
}
```

OUTPUT:

RESULT:

The java program to implement Queue and Stack data structures using classes and objects has been verified and executed successfully.

EX NO: 3	EMPLOYEE CLASS
DATE:	

AIM:

To develop a java program to define an abstract class “Employee” with Emp_name, Emp_id, Address with sub-classes “Professor”, ”AssistantProfessor”,”AssociateProfessor” and ”Programmer” and add BasicPay(bp) as the member of all the inherited classes with 97% of bp as da,10% of bp as HRA,12% of bp as PF,0.1% of bp for staff club funds and Generate pay slips for the employees with their gross and net salaries.

ALGORITHM:

Step 1: Start of the program

Step 2: Create a class employee with the instance variables Emp_name,Emp_id,Address,Mail_id and Mobile_no

Step 3: Create a constructor to initialize the instance variables

Step 4: Create classes Programmer,AssociateProfessor,AssistantProfessor,Professor and add new instance bp,hra,da,pf,scf,gross,net for Basic Pay and the other allowances.

Step 5: Create a function input() to input the details from the user ,printslip() in all the inherited classes to display the payslip of the employee,calculate() to calculate net and gross salary

Step 6: Create a main class and the main function and create objects for the inherited classes and call the functions of the inherited classes

Step 7: Display the results

Step 8: Stop the program

SOURCE CODE:

```
import java.util.Scanner;

class Employee
 {
   String Emp_name,Emp_id,Address,Mail_id,Mobile_no;
   Employee()
    {
      Emp_name ="";
```

```
        Emp_id ="";
        Address ="";
        Mail_id ="";
        Mobile_no="";
    }
}


class Programmer extends Employee
{
    double bp,hra,da,pf,scf,gross,net;
    Programmer()
     {
        super();
     }
    void input()
    {

        Scanner in = new Scanner(System.in);
        System.out.println("Enter Name:");
        super.Emp_name = in.nextLine();
        System.out.println("Enter ID:");
        super.Emp_id = in.nextLine();
        System.out.println("Enter address:");
        super.Address = in.nextLine();
        System.out.println("Enter mail:");
        super.Mail_id = in.nextLine();
        System.out.println("Enter Mobile No.:");
        super.Mobile_no = in.nextLine();
        System.out.println("Enter Base Salary:");
        bp=in.nextDouble();
        da=(0.97)*bp;
```

```
        hra=0.1*bp;
        pf=0.12*bp;
        scf=0.001*bp;
        in.close();
     }

    void calculate()
     {
        gross=bp+hra+da+scf;
        net=gross-pf;
     }

    void printslip()
    {
        System.out.println("Name:\t"+this.Emp_name);
        System.out.println("ID:\t"+this.Emp_id);
        System.out.println("Address:\t"+this.Address);
        System.out.println("Mail ID:\t"+this.Mail_id);
        System.out.println("Mobile No.:\t"+this.Mobile_no);
        System.out.println("-----------------------------------");
        System.out.println("Base Salary:\t"+this.bp);
        System.out.println("HRA:\t"+this.hra);
        System.out.println("DA:\t"+this.da);
        System.out.println("PF:\t"+this.pf);
        System.out.println("GROSS SALARY:\t"+this.gross);
        System.out.println("NET SALARY:\t"+this.net);
    }
 }

class Professor extends Employee
{
    double bp,hra,da,pf,scf,gross,net;
```

```
 Professor()
{
    super();
 }
 void input()
{
    Scanner in = new Scanner(System.in);
    System.out.println("Enter Name:");
    super.Emp_name = in.nextLine();
    System.out.println("Enter ID:");
    super.Emp_id  = in.nextLine();
    System.out.println("Enter address:");
    super.Address = in.nextLine();
    System.out.println("Enter mail:");
    super.Mail_id = in.nextLine();
    System.out.println("Enter Mobile No.:");
    super.Mobile_no = in.nextLine();
    System.out.println("Enter Base Salary:");
    bp=in.nextDouble();
    da=(0.97)*bp;
    hra=0.1*bp;
    pf=0.12*bp;
    scf=0.001*bp;
    in.close();
 }

 void calculate()
 {
    gross=bp+hra+da+scf;
    net=gross-pf;
 }
```

```
    void printslip()
    {
        System.out.println("Name:\t"+this.Emp_name);
        System.out.println("ID:\t"+this.Emp_id);
        System.out.println("Address:\t"+this.Address);
        System.out.println("Mail ID:\t"+this.Mail_id);
        System.out.println("Mobile No.:\t"+this.Mobile_no);
        System.out.println("-----------------------------------");
        System.out.println("Base Salary:\t"+this.bp);
        System.out.println("HRA:\t"+this.hra);
        System.out.println("DA:\t"+this.da);
        System.out.println("PF:\t"+this.pf);
        System.out.println("GROSS SALARY:\t"+this.gross);
        System.out.println("NET SALARY:\t"+this.net);
    }
}

class AssistantProfessor extends Employee
{
    double bp,hra,da,pf,scf,gross,net;
    AssistantProfessor()
     {
        super();
     }
    void input()
    {
        Scanner in = new Scanner(System.in);
        System.out.println("Enter Name:");
        super.Emp_name = in.nextLine();
        System.out.println("Enter ID:");
        super.Emp_id = in.nextLine();
        System.out.println("Enter address:");
```

```
    super.Address = in.nextLine();
    System.out.println("Enter mail:");
    super.Mail_id = in.nextLine();
    System.out.println("Enter Mobile No.:");
    super.Mobile_no = in.nextLine();
    System.out.println("Enter Base Salary:");
    bp=in.nextDouble();
    da=(0.97)*bp;
    hra=0.1*bp;
    pf=0.12*bp;
    scf=0.001*bp;
    in.close();
}

void calculate()
{
    gross=bp+hra+da+scf;
    net=gross-pf;
 }

void printslip()
 {
    System.out.println("Name:\t"+this.Emp_name);
    System.out.println("ID:\t"+this.Emp_id);
    System.out.println("Address:\t"+this.Address);
    System.out.println("Mail ID:\t"+this.Mail_id);
    System.out.println("Mobile No.:\t"+this.Mobile_no);
    System.out.println("-------------------------------------");
    System.out.println("Base Salary:\t"+this.bp);
    System.out.println("HRA:\t"+this.hra);
    System.out.println("DA:\t"+this.da);
    System.out.println("PF:\t"+this.pf);
```

```
        System.out.println("GROSS SALARY:\t"+this.gross);
        System.out.println("NET SALARY:\t"+this.net);
    }
}
class AssociateProfessor extends Employee
  {
    double bp,hra,da,pf,scf,gross,net;
    AssociateProfessor()
     {
        super();
     }
    void input()
    {
        Scanner in = new Scanner(System.in);
        System.out.println("Enter Name:");
        super.Emp_name = in.nextLine();
        System.out.println("Enter ID:");
        super.Emp_id = in.nextLine();
        System.out.println("Enter address:");
        super.Address = in.nextLine();
        System.out.println("Enter mail:");
        super.Mail_id = in.nextLine();
        System.out.println("Enter Mobile No.:");
        super.Mobile_no = in.nextLine();
        System.out.println("Enter Base Salary:");
        bp=in.nextDouble();
        da=(0.97)*bp;
        hra=0.1*bp;
        pf=0.12*bp;
        scf=0.001*bp;
        in.close();
    }
```

```
    void calculate()
   {
        gross=bp+hra+da+scf;
        net=gross-pf;
    }
    void printslip()
    {
        System.out.println("Name:\t"+this.Emp_name);
        System.out.println("ID:\t"+this.Emp_id);
        System.out.println("Address:\t"+this.Address);
        System.out.println("Mail ID:\t"+this.Mail_id);
        System.out.println("Mobile No.:\t"+this.Mobile_no);
        System.out.println("-------------------------------------");
        System.out.println("Base Salary:\t"+this.bp);
        System.out.println("HRA:\t"+this.hra);
        System.out.println("DA:\t"+this.da);
        System.out.println("PF:\t"+this.pf);
        System.out.println("GROSS SALARY:\t"+this.gross);
        System.out.println("NET SALARY:\t"+this.net);
    }
}
class main
{
public static void main(String args[])
    {
        Programmer pro = new Programmer();
        System.out.println("Programmer:");
        pro.input();
        pro.calculate();
        pro.printslip();
        Professor pro1 = new Professor();
        System.out.println("Professor:");
```

```
        pro1.input();
        pro1.calculate();
        pro1.printslip();
        System.out.println("Assistant Professor:");
        AssistantProfessor pro2 = new AssistantProfessor();
        pro2.input();
        pro2.calculate();
        pro2.printslip();
        AssociateProfessor pro3 = new AssociateProfessor();
       System.out.println("Associate Professor:");
        pro3.input();
        pro3.calculate();
        pro3.printslip();
    }
}
```

OUTPUT:

RESULT:

Thus, the given program for the payslip of an employee has been verified and executed successfully using the concept of abstract class.

EX NO: 4	SHAPE CLASS
DATE:	

AIM:

To develop a java program to create an abstract class “Shape” that contains two integers and an abstract method named printArea().Provide three classes named Rectangle, Triangle, Circle such that each one of the classes inherits the class “Shape”. Each one of the classes contains only the method printArea() that prints the area of the given shape

ALGORITHM:

Step 1: Start of the program

Step 2: Create an abstract class Shape and declare instance variables x and y and create a constructor to initialize the variables and create an abstract function printArea()

Step 3: Create classes Rectangle,Triangle,Circle and inherit the Shape class and create constructors to set the values of the instances in the super class

Step 4: Define printArea() functions in the classes to calculate the areas:

Rectangle: x*y

Triangle: 0.5*x*y

Circle: pi*x*y (x=y=radius)

Step 5: Create main class and define main function ,create objects for Rectangle,Triangle and Circle ,pass the values of the shapes during the object creation and call the functions

Step 6: Stop the program

SOURCE CODE:

```
import java.util.*;
import java.math.*;
abstract class Shape
{
   int x,y;
   Shape(int a,int b)
  {
      x=a;
```

```
        y=b;
    }
    abstract void printArea();
}
 class Rectangle extends Shape
{
    Rectangle(int a,int b)
    {
        super(a,b);
     }
    public void printArea()
    {
         int area =this.x*this.y;
         System.out.println("The Area of Rectangle = "+area);
     }
}
 class Triangle extends Shape
{
    Triangle(int base, int height)
     {
        super(base, height);
     }
    void printArea()
     {
         double area = 0.5*super.x*super.y;
         System.out.println("The Area of Triangle = "+area);
      }
}
 class Circle extends Shape
{
    Circle(int radius)
     {
```

```
        super(radius,radius);
    }

    void printArea()
     {
         double area = Math.PI*super.x*super.y;
         System.out.println("The Area of Circle = "+area);
     }
}
 class Main
{
   public static void main(String args[])
    {
        int r;
        Scanner in = new Scanner(System.in);
        System.out.println("Enter Radius: ");
        r = in.nextInt();
        Circle c = new Circle(r);
        c.printArea();

        int a,b;
        System.out.println("Enter Base: ");
        a = in.nextInt();
        System.out.println("Enter Height: ");
        b = in.nextInt();
        Triangle t = new Triangle(a,b);
        t.printArea();

       int ra,rb;
        System.out.println("Enter length: ");
        ra = in.nextInt();
        System.out.println("Enter width: ");
```

```
        rb = in.nextInt();
        Rectangle rect = new Rectangle(ra,rb);
        rect.printArea();


    }
}
```

OUTPUT:

RESULT:

Thus, the given program for printing the area of the given shape has been verified and executed successfully using the concept of abstract class.

EX NO: 5	SHAPE INTERFACE
DATE:	

AIM:

To develop a java program to create an interface “Shape” that contains an empty method named printArea().Provide three classes named Rectangle, Triangle, Circle such that each one of the classes implements the interface “Shape”. Each one of the classes contains only the method printArea() that prints the area of the given shape.

ALGORITHM:

Step 1: Start of the program

Step 2: Create an interface Shape with printArea() method

Step 3: Create classes rectangle,triangle and circle with its dimensions as instance variables and implement Shape and define printArea() in which calculate Area according to the shape :

Rectangle : x*y

Triangle: 0.5*x*y

Circle: pi*radius*radius

Step 4: Create main class and define main function and instantiate the inherited classes rectangle,triangle,circle ,pass the values for the dimensions of the shapes during object instantiation and call the functions

Step 5: Stop the program

SOURCE CODE:

```
import java.math.*;
import java.util.*;
public interface Shape1
{
   void printArea();
}
public class Rectangle implements Shape1
{
   int x,y;
```

```
   Rectangle(int a,int b)
   {
      this.x=a;
      this.y=b;
    }
   public void printArea()
    {
      int area =this.x*this.y;
      System.out.println("The Area of Rectangle = "+area);
    }
}
public class Triangle extends Shape1
{
int x,y;
 Triangle(int base, int height)
    {
      this.x=base;
     this.y=height;
    }
   void printArea()
    {
      double area = 0.5*this.x*this.y;
      System.out.println(“The Area of Triangle = “+area);
    }
}
public class Circle extends Shape1
{
int x,y;
Circle(int radius)
  {
    this.x=radius;
    this.y=radius;
```

```
    }
   void printArea()
    {
        double area = Math.PI*this.x*this.y;
        System.out.println("The Area of Circle = "+area);
    }
}

public class Main
{
public static void main(String args[])
   {
       int r;
       Scanner in = new Scanner(System.in);
       System.out.println("Enter Radius: ");
       r = in.nextInt();
       Circle c = new Circle(r);
       c.printArea();
       int a,b;
       System.out.println("Enter Base: ");
       a = in.nextInt();
       System.out.println("Enter Height: ");
       b = in.nextInt();
       Triangle t = new Triangle(a,b);
       t.printArea();
      int ra,rb;
       System.out.println("Enter length: ");
       ra = in.nextInt();
       System.out.println("Enter width: ");
       rb = in.nextInt();
       Rectangle rect = new Rectangle(ra,rb);
       rect.printArea();
```

```
  }
}
```

OUTPUT:

RESULT:

Thus, the given program for printing the area of the given shape has been verified and executed successfully using the concept of interface.

EX NO: 6	EXCEPTION HANDLING
DATE:	

AIM:

To develop a java program that validates the age of the user whether they are valid for voting and takes the vote either of the candidates a, b, c or d and prints the vote count of each candidate

ALGORITHM:

Step 1: Start of the program

Step 2: Initialize variables a,b,c,d to 0

Step 3: Within the try block, input the age of the user

Step 4: If the age is less than 18,throw an exception with an appropriate message

Step 5: If an exception is thrown, print the message of the exception

Step 6: Else, input the vote,input being either a ,b, c or d and increment the count of the variables a,b,c,d accordingly to the input

Step 7: If the choice does not match with the given set mentioned above,then throw an exception

Step 8: If an exception is thrown, print the message and call the main function again

Step 9: Stop the program

SOURCE CODE:

```
import java.io.IOException;
import java.util.*;
public class vote
{
   public static void main(String args[])
    {
      int a=0,b=0,c=0,d=0;
      try
        {
          Scanner in = new Scanner(System.in);
          System.out.println("Enter Age:");
          int age = in.nextInt();
          if(age<18)
```

```
    {
   throw new ArithmeticException("Age is Under 18: Not Valid for Voting");
    }
   else {
      System.out.println("Valid for Voting...");
      System.out.println("Enter your vote:\na\nb\nc\nd");
      String choice = in.next();
      switch(choice.toLowerCase())
       {
      case "a":
         a++;
         break;
      case "b":
         b++;
         break;
      case "c":
         c++;
         break;
      case "d":
         d++;
         break;
      default:
         throw new IOException("Invalid Choice");
        }
      System.out.println("The vote count:");
      System.out.println("a: "+a);
      System.out.println("b: "+b);
      System.out.println("c: "+c);
      System.out.println("d: "+d);
    }
}
catch(ArithmeticException e)
```

```
        {
          System.out.println(e.getMessage());
        }
       catch(IOException e)
        {
          System.out.println(e.getMessage());
          main(null);
        }
    }
}
```

OUTPUT:

RESULT:

The java program that validates the age of the user whether they are valid for voting and takes the vote either of the candidates a, b, c or d and prints the vote count of each candidate has been verified and executed successfully.

EX NO: 7	MULTITHREADING
DATE:	

AIM:

To develop a java program that implements a multithreaded application that has three threads .First thread generates a random integer every 1 second and if the value is even ,the second thread computes the square of the number and prints. If the value is odd, then the third thread computes the cube of the number and prints it.

ALGORITHM:

Step 1: Start of the program

Step 2: Create three class for each thread tasks. i.e., thread1 , thread2 and thread3 and 1 main class

Step 3: In thread1 generate a random integer, rand every 1 sec

Step 4: If rand is even. Set the data and start thread 2 to square and print the number.

Step 5: If rand is odd. Set the data and start thread 3 to cube and print the number.

Step 6: In thread 2, using a setter get the data and square and print it

Step 7: In thread 3, using a setter get the data and cube and print it

Step 8: If an exception is thrown, print the message and call the main function again

Step 9: Stop the program

SOURCE CODE:

```
public class rand
{
   public static void main(String args[]) throws InterruptedException
     {
       thread1 ob = new thread1();
       Thread t1 = new Thread(ob);


       t1.start();
     }
}
class thread1 extends Thread
{
   int rand;
```

```
thread2 o;
thread3 b;

thread1()
  {
    this.rand = (int) (Math.random()*100);
    o = new thread2();
    o.setData(rand);
    b = new thread3();
    b.setData(rand);
  }
public void task()
 {
    Thread t2 = new Thread(o);
    Thread t3 = new Thread(b);

    try
     {
        this.rand = (int) (Math.random()*100);
        System.out.println("from 1: "+rand);
        if(rand%2==0)
           {
            o.setData(rand);
            t2.start();
            }
        else
          {
            b.setData(rand);
            t3.start();
          }
              Thread.sleep(1000);
      }
```

```
        catch (InterruptedException e)
        {
            e.printStackTrace();
         }
    }
@Override
public void run()
    {
       for(int i=0;i<5;i++)
            {
            this.task();
            System.out.println("");
             }
    }
}
class thread2 extends Thread
{
   int a;
   void setData(int a)
    {
      this.a =a;
    }
   @Override
   public void run()
   {
     System.out.println("from 2: "+Math.pow(a , 2));
    }
}
class thread3 extends Thread
{
   int a;
   void setData(int a)
```

```
    {
      this.a=a;
    }
   @Override
   public void run()
    {
      System.out.println("from 3: "+Math.pow(a, 3));

    }
}
```

OUTPUT:

RESULT:

Thus, the given program for the printing the squares and cubes of even and odd randomly generated integers respectively has been verified and executed successfully using the concept of multithreading

EX NO: 8	FILE OPERATIONS
DATE:	

AIM:

To develop a java program to create a file with a user-defined name , write some data in the file and read the data and print it on the console.

ALGORITHM:

Step 1: Start of the program

Step 2: Within the try block, input the name of the file

Step 3: Create the file using file class. And print the appropriate message.

Step 4: Input the contents of the files which you want to write it into the file.

Step 5: For read operation we read the same file with scanner and print the contents.

Step 6: Delete the file using file inbuilt method

Step 7: If an exception is thrown, print the message and call the main function again

Step 8: Stop the program

SOURCE CODE:

```
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileWriter;
import java.io.IOException;
import java.util.*;

public class Fops
 {
   public static void main(String atgs[]) throws FileNotFoundException
      {
      try
         {
            Scanner in = new Scanner(System.in);
         System.out.println("Enter file name:");
         String fname = in.nextLine();
         File file = new File(fname);
         System.out.println("File Created!");
         System.out.println("\nWriting data:\nEnter data:");
         FileWriter fw;
```

```
            fw = new FileWriter(file);
            String con;
            do
             {
                  con= in.nextLine();
                  fw.write(con+"\n");;
               }while(con!="");
            fw.close();
            System.out.println("\nReading data:...");
            Scanner f = new Scanner(file);
            while(f.hasNextLine())
              {
               System.out.println(f.nextLine());
              }
            f.close();
            System.out.println("\nDeleting the file...");
            file.delete();
            System.out.println("File Deleted!");
        }
        catch (IOException e)
         {
           e.printStackTrace();
         }
    }
}
```

OUTPUT:

RESULT:

Thus ,the java program to implement file operations has been verified and executed successfully.

EX NO: 9	JAVA GENERICS
DATE:	

AIM:

To develop a java application that implements the Stack data structure using java generics classes

ALGORITHM:

Step 1: Start of the program

Step 2: Inside the main class create a nested class to store the nodes.

Step 3: Initialize top object with null

Step 4: Create a method to push the data to the linked list and point it towards new top.

Step 5: Create a method to pop the data from the top and point to the previous data.

Step 6: Create a test class to check the operations and implementation.

Step 7: Print the sum of all elements after creating the stack using the Generics.

Step 9: Stop the program

SOURCE CODE:

```
import java.util.Scanner;
 class StackGen<T> {

    class Node<T>{
        T data;
        Node<T> next;

        Node(T data){
            this.data= data;
            this.next = null;
        }

    }

    Node<T> top;
    StackGen(){
        this.top =null;
    }
```

```
    boolean isEmpty() {
        return this.top==null;
    }

    boolean push(T data) {
        Node<T> newnode = new Node<T>(data);
        newnode.next = this.top;
        this.top = newnode;
        return this.top.next!=null;
    }

    T pop() {
        T data = this.top.data;
        this.top=this.top.next;
        return data;
    }

    T peek() {
        return this.top.data;
    }
}
 public class StackGenTest {
    public static void main(String args[]) {
        Scanner in = new Scanner(System.in);

        System.out.println("Enter the size of integer stack:");

        int isize = in.nextInt();

        StackGen<Integer> stk = new StackGen<Integer>() ;

        System.out.println("\nEnter data:");
```

```
        for(int i=0;i<isize;i++) {
            stk.push(in.nextInt());
        }

        System.out.println("\n");
        int sum=0;

        while(!stk.isEmpty()) {
            System.out.println(stk.peek());
            sum+=stk.pop();
            System.out.println("--");

        }

        System.out.println("The sum of the elements: "+sum);
    }
}
```

OUTPUT:

RESULT:

The java application that implements the Stack data structure has been verified and executed successfully using the concept of java generics classes.

EX NO: 10	JAVAFX
DATE:	

AIM:

To develop a mini-game using javafx in which a button when clicked moves to a random position within the application window and displays how long the player took to click the button 20 times in a row.

SOURCE CODE:

```
package com.example.demo;

import javafx.application.Application;

import javafx.collections.FXCollections;

import javafx.event.ActionEvent;

import javafx.event.Event;

import javafx.event.EventHandler;

import javafx.scene.Scene;

import javafx.scene.control.Button;

import javafx.scene.control.ChoiceBox;

import javafx.scene.image.Image;

import javafx.scene.input.MouseEvent;

import javafx.scene.layout.AnchorPane;

import javafx.scene.text.Font;

import javafx.stage.Stage;

import javafx.scene.control.Label;

import java.io.IOException;

import java.net.URL;

import java.util.ResourceBundle;



public class Clicker extends Application

 {

    int count=0;

    double t1,t2;


    @Override

    public void start(Stage stage) throws IOException
```

```
{
    Label text = new Label("0");
    AnchorPane bg = new AnchorPane();
    String[] colors = {"RED","BLUE","BLACK","GREEN","WHITE","CYAN","PINK"};
    ChoiceBox colorchanger = new ChoiceBox(FXCollections.observableArrayList(colors));
    Button button = new Button();

    bg.setStyle("-fx-background-color :LIGHTSKYBLUE;");



    colorchanger.setLayoutX(14);
    colorchanger.setLayoutY(14);
    colorchanger.setPrefWidth(150);
    colorchanger.setOnAction(event ->
       {
        String bgcolor = (String) colorchanger.getValue();
        bg.setStyle("-fx-background-color: "+bgcolor);
       }
     );

 button.setLayoutX(119);
 button.setLayoutY(353);
 button.setText("START");
 button.setFont(Font.font("Bauhaus 93",15));

 EventHandler<MouseEvent> eventHandler = new EventHandler<MouseEvent>()
     {
      @Override
       public void handle(MouseEvent mouseEvent)
        {
          setCount(button,text);
     }
```

```
    };
    button.addEventFilter(MouseEvent.MOUSE_CLICKED,eventHandler);

    text.setFont(Font.font("Bauhaus 93",50));
    AnchorPane.setTopAnchor(text,192.0);
    AnchorPane.setLeftAnchor(text,136.0);
    bg.getChildren().addAll(text,button,colorchanger);
    Scene scene = new Scene(bg, 300, 500);

    stage.setTitle("Clicker!!");
    Image icon = new Image("C:\\Users\\gamar\\IdeaProjects\\demo1\\src\\cursor-vector-icon.jpg");
    stage.getIcons().add(icon);
    stage.setScene(scene);
    stage.setResizable(false);
    stage.show();
}


public void setCount(Button button , Label text)
 {
    count++;
    button.setText("CLICK");
    text.setText(Integer.toString(count));
    button.setLayoutX(Math.random()*250);
    button.setLayoutY((Math.random()*375)+100);
    AnchorPane.setLeftAnchor(text,136.0);
    AnchorPane.setTopAnchor(text,192.0);
    text.setFont(Font.font("Bauhaus 93",50));
    if(count==1)
     {
       t1=System.currentTimeMillis();
     }
```

```
        if(count==20)
         {
            t2=System.currentTimeMillis();
            text.setText("YOU'VE CLICKED\n 20 TIMES IN:\n"+((t2-t1)/1000)+" SECS");
            text.setFont(Font.font("Bauhaus 93",20));
            AnchorPane.setLeftAnchor(text,86.0);
            AnchorPane.setTopAnchor(text,192.0);
            button.setLayoutX(119);
            button.setLayoutY(353);
            button.setText("START");
            count=0;
         }
      }


      public static void main(String[] args)
     {
         launch();
      }
}
```

OUTPUT:

RESULT:

The mini-game using javafx in which a button when clicked moves to a random position within the application window and displays how long the player took to click the button 20 times in a row has been successfully executed.