GURU TEGH BAHADUR INSTITUTE OF TECHNOLOGY

SEMESTER – IV

JAVA PRACTICAL FILE

SUBJECT CODE: CIC-258

Submitted to,

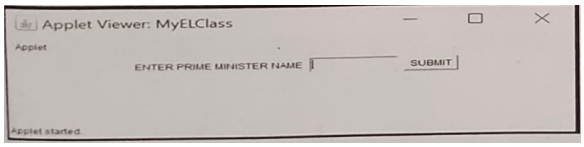
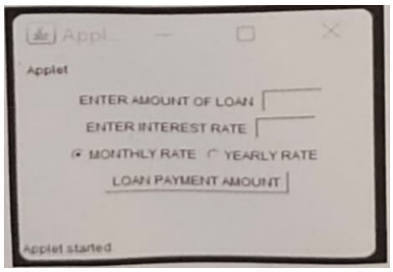
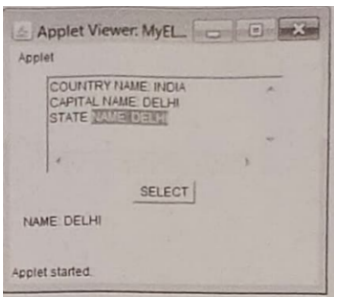
Mr. Gaurav Sandhu

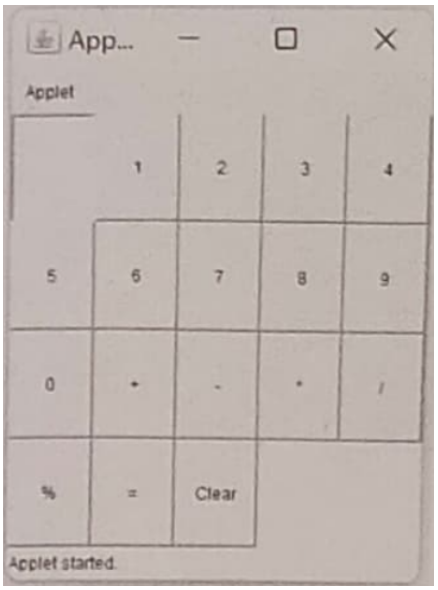
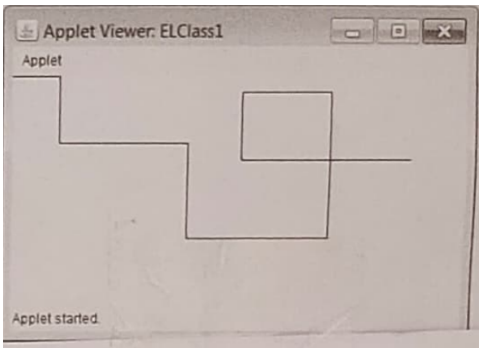
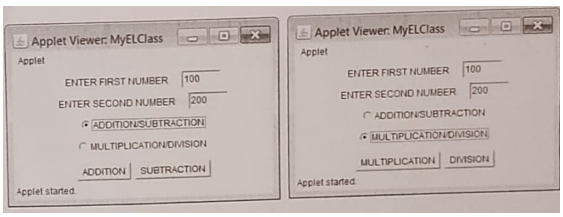
Submitted by,

Abhay Raj (00976803122)

Index

11.	Design a java program to show dynamic polymorphism.			
12.	Design a java program to show interfaces.			
13.	Design a java program to create a customized exception and also make use of all 5 exception keywords.			
14.	Design a java program to sort list of names in ascending order.			
15.	Design a java program to create two threads: one for odd numbers and other for even numbers.			
16.	Design a java program to implement producer consumer problem using concept of inter thread communication.			
17.	Design a java program to create three threads runnable interface. First thread display "GOOD MORNING" at every 1 second. Second thread displays "HELLO" at every 2 seconds and third thread displays "WELCOME" at every 3 seconds.			
18.	Design a java program to create three threads. First thread display "GOOD MORNING" at every 1 second. Second thread displays "HELLO" at every 2 seconds and third thread displays "WELCOME" at every 3 seconds.			
19.	Design an applet to do the following tasks: a) To output the question "WHO IS THE PRIME MINISTER OF INDIA?" b) To accept the answer and print "CORRECT" and then stop if the answer is correct. c) To print "TRY AGAIN" and if the answer is not correct. d) To display the correct answer, if the answer is wrong even after third attempt.			

				
20.	Design a java program to write an applet that computes the payment of loan which is based on the amount of loan, interest rate and number of months. It takes one parameter from browser which is called Monthly Rate. If it is True then interest rate is per month else it is annual. (Use label, textfield, button, check box, checkbox group). 			
21.	Design a java program to write an applet with following AWT components :textarea and button. 			
22.	Design a java program to write a calculator			

	applet. Use grid layout to arrange the buttons for digits and for the +,-,*,/,% operations. Add textfield to display the result. 			
23.	Design a java program to handle keyboard events. 			
24.	Design a program to Implement card layout in applets. 			
25.	Program In Java To Demonstrate FileWriter,			

	FileReader, BufferedReader classes to Enter Multiple Strings And Numbers Into A File, Read File Using Loop And Display.			
26.	Program to implement various functions giving details about a file.			
27.	Design a program in java to implement client-server TCP\IP socket application which exchange strings among them.			
28.	Program In Java To Make Database Connectivity Using JDBC, create statement object, create resultset object And Display (All Or Specific Records using first(), last(), previous(), next() and absolute() functions). (Pre requirements: install MySql software, create database: Shiksha, create table house with attributes as H_No, H_Locality, H_Rooms, H_Owner, insert rows of data into table house)			
29.	Program In Java To Make Database Connectivity Using JDBC, create statement object, create resultset object, insert a record using prepared statement and Display all records. (Pre requirements: install MySql software, create database: Shiksha, create table house with attributes as H_No, H_Locality, H_Rooms, H_Owner, insert rows of data into table house)			
30.	Program In Java To Make Database Connectivity Using JDBC, create statement object, create resultset object, update a record based on H_No and Display all records. (Pre requirements: install MySql software, create database: Shiksha, create table house with attributes as H_No, H_Locality, H_Rooms, H_Owner, insert rows of data into table house)			
31.	Program In Java To Make Database Connectivity Using JDBC, create statement object, create resultset object, delete a record based on H_No and Display all records. (Pre requirements: install MySql software, create database: Shiksha, create table house with attributes as H_No, H_Locality, H_Rooms, H_Owner, insert rows of data into table house)			

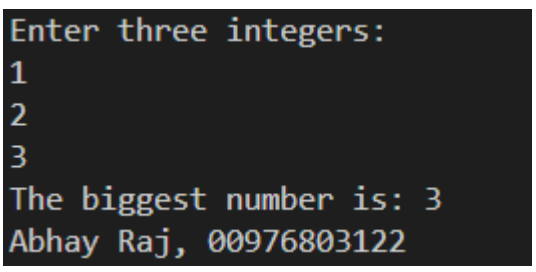
Program – 1

Aim: Design a java program to find the biggest of three given integers.

Code:

```
import java.util.*;

class BiggestNumberFinder {
   public static int findBiggest(int num1, int num2, int num3) {
      int max = num1;

      if (num2 > max) {
         max = num2;
      }

      if (num3 > max) {
         max = num3;
      }

      return max;
   }
}

public class MyELClass {
   public static void main(String[] args) {
      Scanner in = new Scanner(System.in);

      System.out.println("Enter three integers:");
      int num1 = in.nextInt();
      int num2 = in.nextInt();
      int num3 = in.nextInt();

      int max = BiggestNumberFinder.findBiggest(num1, num2, num3);

      System.out.println("The biggest number is: " + max);
      System.out.println("Abhay Raj, 00976803122" );

      in.close();
   }
}
```

Output:



Program – 2

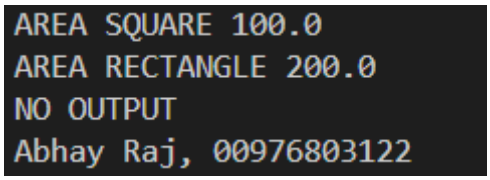
Aim: Design a java program to define a class, describe its constructor, over-load the constructor and instantiate the object.

Code:

```
class Shapes
{
	double len;
	double bre;
	double result;

	Shapes()
	{
		len=0.0;
		bre=0.0;
	}

	Shapes(double l)
	{
		len=l;
		bre=0.0;
	}

	Shapes(double l, double b)
	{
		len=l;
		bre=b;
	}

	void area()
	{
		if(len!=0.0 && bre==0.0)
		{
			result=len*len;
			System.out.println("AREA SQUARE " + result);
		}
		if(len!=0.0 && bre!=0.0)
		{
			result=len*bre;
			System.out.println("AREA RECTANGLE " + result);
		}
		else if(len==0.0 && bre==0.0)
		{
			System.out.println("NO OUTPUT");
		}
	}
}
```

```
class MyELClass
{
	public static void main(String a[])
	{
		double para1, para2;
		para1=10.0;
		para2=20.0;
		Shapes s1=new Shapes();
		Shapes s2=new Shapes(para1);
		Shapes s3=new Shapes(para1,para2);
		s2.area();
		s3.area();
		s1.area();
		System.out.println("Abhay Raj, 00976803122" );

	}
}
```

Output:



Program – 3

Aim: Design a java program to demonstrate the use of nested class.

Code:

```
class Outer
{
	private int outer_x = 100;
	void test()
	{
		Inner inner = new Inner();
		inner.display();
		inner.hello();
	}

	class Inner
	{
		private void hello()
		{
		   System.out.println("Private Method Hello() of Inner Class");
		}

		void display()
		{
		System.out.println("Display: outer_x = " + outer_x);
		Outer o=new Outer();
		System.out.println("Display: outer_x = " + o.outer_x);
		}
	}
}

class MyELClass
{
	public static void main(String args[])
	{
		Outer outer = new Outer();
		outer.test();
		System.out.println("Abhay Raj, 00976803122" );
	}
}
```

Output:

```
Display: outer_x = 100
Display: outer_x = 100
Private Method Hello() of Inner Class
Abhay Raj, 00976803122
```

Program – 4

Aim: Design a java program to print all the real solutions of the quadratic equation ax^2+bx+c=0, read in the values of a, b, c and use the quadratic formula. If the discriminant b^2-4ac is negative, display a message stating that there are no real solutions.

Code:

```
import java.util.*;

class QuadraticEquationSolver {
   public static void solveQuadraticEquation(double a, double b, double c) {
      double discriminant = b * b - 4 * a * c;

      if (discriminant < 0) {
         System.out.println("No real solutions exist.");
      } else {
         double root1 = (-b + Math.sqrt(discriminant)) / (2 * a);
         double root2 = (-b - Math.sqrt(discriminant)) / (2 * a);

         System.out.println("The real solutions of the quadratic equation are:");
         System.out.println("Root 1: " + root1);
         System.out.println("Root 2: " + root2);
      }
   }
}

public class MyELClass {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);

      System.out.println("Enter the coefficients of the quadratic equation ax^2 + bx + c =
0:");
      System.out.print("a: ");
      double a = scanner.nextDouble();
      System.out.print("b: ");
      double b = scanner.nextDouble();
      System.out.print("c: ");
      double c = scanner.nextDouble();

      QuadraticEquationSolver.solveQuadraticEquation(a, b, c);

      scanner.close();
      System.out.println("Abhay Raj, 00976803122" );
   }
}
```

Output:

```
Enter the coefficients of the quadratic equation ax^2 + bx + c = 0:
a: 1
b: 2
c: 1
The real solutions of the quadratic equation are:
Root 1: -1.0
Root 2: -1.0
Abhay Raj, 00976803122
```

Program – 5

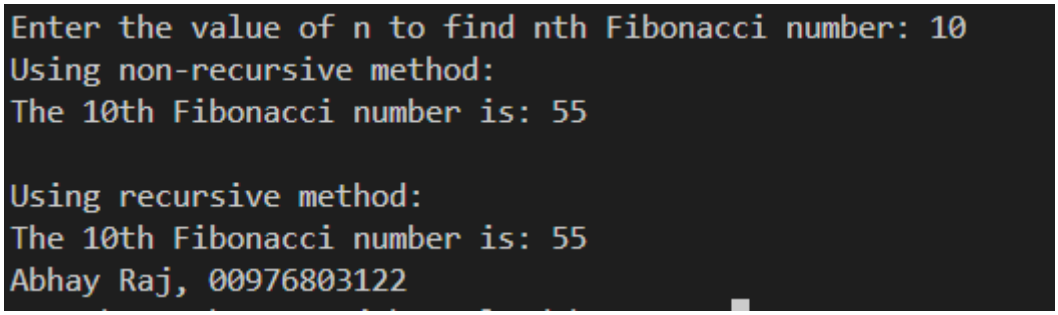
Aim: Design a java program that uses both recursive and non-recursive methods to print the nth value of the Fibonacci series sequence.

Code:

```
import java.util.*;

class FibonacciCalculator {
    // Non-recursive method to find nth Fibonacci number
    public static long fibonacciNonRecursive(int n) {
        long a = 0;
        long b = 1;
        long fib = 0;

        if (n == 0) return a;
        if (n == 1) return b;

        for (int i = 2; i <= n; i++) {
            fib = a + b;
            a = b;
            b = fib;
        }

        return fib;
    }

    // Recursive method to find nth Fibonacci number
    public static long fibonacciRecursive(int n) {
        if (n <= 1) return n;
        return fibonacciRecursive(n - 1) + fibonacciRecursive(n - 2);
    }
}

public class MyELClass {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the value of n to find nth Fibonacci number: ");
        int n = scanner.nextInt();

        // Using non-recursive method
        System.out.println("Using non-recursive method:");
        long resultNonRecursive = FibonacciCalculator.fibonacciNonRecursive(n);
        System.out.println("The " + n + "th Fibonacci number is: " + resultNonRecursive);

        // Using recursive method
        System.out.println("\nUsing recursive method:");
        long resultRecursive = FibonacciCalculator.fibonacciRecursive(n);
        System.out.println("The " + n + "th Fibonacci number is: " + resultRecursive);

        scanner.close();
```

```
    System.out.println("Abhay Raj, 00976803122" );
  }
}
```

Output:



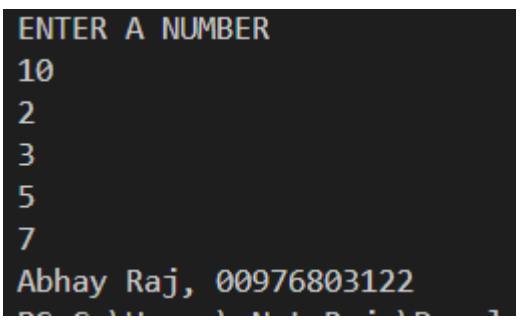
Program – 6

Aim: Design a java program to prompt user for an integer and print all the prime numbers up to that integer.

Code:

```
import java.util.*;

class Prime {
   int value, num;
   int i, flag;

   void inputNumber() {
      Scanner sc = new Scanner(System.in);
      System.out.println("ENTER A NUMBER ");
      value = sc.nextInt();
   }
   void displayPrime() {
      for (num = 2; num <= value; num++) {
         flag = 0;
         for (i = 2; i <= num - 1; i++) {
            if (num % i == 0) {
               flag = 1;
               break;
            }
         }
         if (flag == 0) {
            System.out.println(num);
         }
      }
   }
}
class MyELClass {
   public static void main(String a[]) {
      Prime o = new Prime();
      o.inputNumber();
      o.displayPrime();
      System.out.println("Abhay Raj, 00976803122" );
   }
}
```

Output:



Program – 7

Aim: Design a java program to check whether a string is palindrome or not.

Code:

```
import java.util.*;

class P_Check {
   void check(String data) {
      String rdata = "";
      int i, len;
      len = data.length();
      for (i = len - 1; i >= 0; i--)
         rdata = rdata + data.charAt(i);
      if (data.equals(rdata))
         System.out.println("Entered string is a palindrome.");
      else
         System.out.println("Entered string is not a palindrome.");
   }
}

class MyELClass {
   public static void main(String a[]) {
      String name;
      P_Check pc = new P_Check();
      Scanner in = new Scanner(System.in);
      System.out.println("Enter a string");
      name = in.nextLine();
      pc.check(name);
      System.out.println("Abhay Raj, 00976803122" );
   }
}
```

```
Enter a string
AAAbAAA
Entered string is a palindrome.
Abhay Raj, 00976803122
```

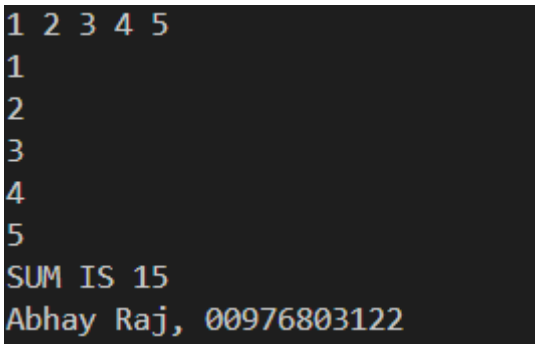
Program – 8

Aim: Design a java program that reads a line of integers, then display each integer and the sum of all integers.

Code:

```
import java.util.*;

class IntData {
   int arr[];
   int i, sum;

   IntData(int size) {
      arr = new int[size];
      sum = 0;
   }

   void input() {
      Scanner sc = new Scanner(System.in);
      for (i = 0; i < arr.length; i++) {
         arr[i] = sc.nextInt();
      }
   }

   void display() {
      for (i = 0; i < arr.length; i++) {
         System.out.println(arr[i]);
      }
   }

   void sumCal() {
      for (i = 0; i < arr.length; i++) {
         sum = sum + arr[i];
      }
      System.out.println("SUM IS " + sum);
   }
}

class MyELClass {
   public static void main(String a[]) {
      IntData obj = new IntData(5);
      obj.input();
      obj.display();
      obj.sumCal();
       System.out.println("Abhay Raj, 00976803122" );
   }
}
```

Output:



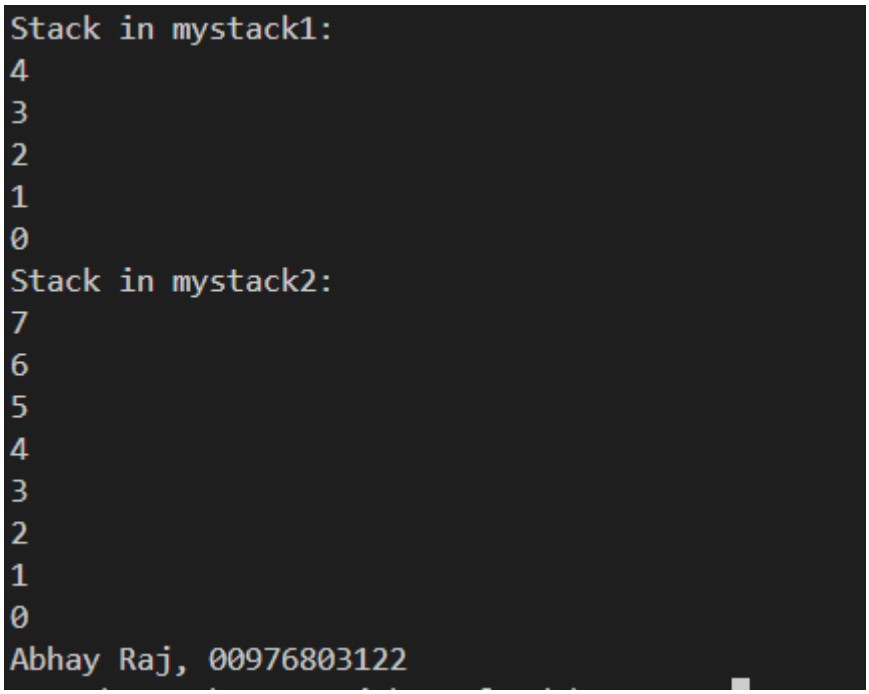
Program – 9

Aim: Design a java program to implement stack and queue concept.

Code:

```
class Stack {
   int stck[];
   int tos;

   Stack(int size) {
      stck = new int[size];
      tos = -1;
   }

   void push(int item) {
      if (tos == stck.length - 1)
         System.out.println("Stack is full.");
      else
         stck[++tos] = item;
   }

   int pop() {
      if (tos < 0) {
         System.out.println("Stack underflow.");
         return 0;
      } else
         return stck[tos--];
   }
}
class MyELClass {
   public static void main(String args[]) {
      Stack mystack1 = new Stack(5);
      Stack mystack2 = new Stack(8);

      for (int i = 0; i < 5; i++)
         mystack1.push(i);

      for (int i = 0; i < 8; i++)
         mystack2.push(i);

      System.out.println("Stack in mystack1:");
      for (int i = 0; i < 5; i++)
         System.out.println(mystack1.pop());

      System.out.println("Stack in mystack2:");
      for (int i = 0; i < 8; i++)
         System.out.println(mystack2.pop());
       System.out.println("Abhay Raj, 00976803122" );
   }
}
```

Output:


Stack in mystack1:
4
3
2
1
0
Stack in mystack2:
7
6
5
4
3
2
1
0
Abhay Raj, 00976803122

Program – 10

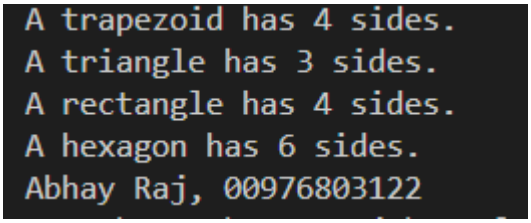
Aim: Design a java program to create an abstract class named Shape, that contains an empty method called numberOfSides(). Provide four classes named Trapezoid, Triangle, Rectangle and Hexagon such that each one of the classes contains only the method numberOfSides(), that contains number of sides in each geometrical figure.

Code:

```
abstract class Shape {
   abstract void numberOfSides();
}

class Trapezoid extends Shape {
   void numberOfSides() {
      System.out.println("A trapezoid has 4 sides.");
   }
}

class Triangle extends Shape {
   void numberOfSides() {
      System.out.println("A triangle has 3 sides.");
   }
}

class Rectangle extends Shape {
   void numberOfSides() {
      System.out.println("A rectangle has 4 sides.");
   }
}

class Hexagon extends Shape {
   void numberOfSides() {
      System.out.println("A hexagon has 6 sides.");
   }
}

public class MyELClass {
   public static void main(String a[]) {
      Shape trapezoid = new Trapezoid();
      trapezoid.numberOfSides();

      Shape triangle = new Triangle();
      triangle.numberOfSides();

      Shape rectangle = new Rectangle();
      rectangle.numberOfSides();

      Shape hexagon = new Hexagon();
      hexagon.numberOfSides();
      System.out.println("Abhay Raj, 00976803122" );
   }
```

}

Output:


A trapezoid has 4 sides.
A triangle has 3 sides.
A rectangle has 4 sides.
A hexagon has 6 sides.
Abhay Raj, 00976803122

Program – 11

Aim: Design a java program to show dynamic polymorphism.

Code:

```
class MyArea {
   void area() {
      System.out.println("A concrete function called Area()");
   }
}

class SQArea extends MyArea {
   double s;

   void area() { // Same name function in base class
      double areasq;
      s = 10.0;
      areasq = s * s;
      System.out.println("Area of Square " + areasq);
   }
}

class RECTArea extends MyArea {
   double l, b;

   void area() { // Same name function in base class
      double arearect;
      l = 10.0;
      b = 2.0;
      arearect = l * b;
      System.out.println("Area of Rectangle " + arearect);
   }
}

class MyELClass {
   public static void main(String a[]) {
      MyArea ma; // Base class object
      ma = new SQArea(); // Referring to SQArea
      ma.area(); // Calling SQArea method (Dynamic Polymorphism)
      ma = new RECTArea(); // Referring to RECTArea
      ma.area(); // Calling RECTArea method (Dynamic Polymorphism)
      System.out.println("Abhay Raj, 00976803122" );
   }
}
```

Output:

```
Area of Square 100.0
Area of Rectangle 20.0
Abhay Raj, 00976803122
```

Program – 12

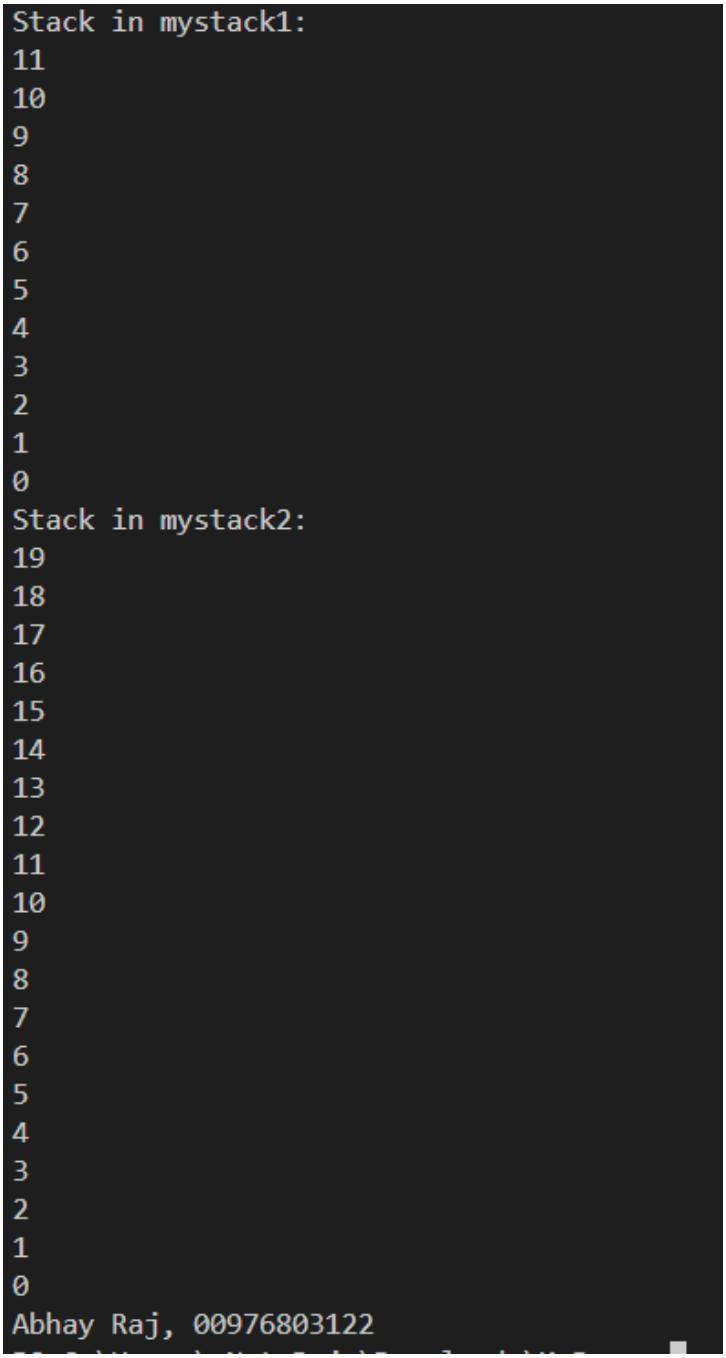
Aim: Design a java program to show interfaces.

Code:

```
interface IntStack {
   void push(int item); // Store an item.
   int pop(); // Retrieve an item.
}

class DynStack implements IntStack {
   private int stck[];
   private int tos;

   // Allocate and initialize stack.
   DynStack(int size) {
      stck = new int[size];
      tos = -1;
   }

   // Push an item onto the stack.
   public void push(int item) {
      // If stack is full, allocate a larger stack.
      if (tos == stck.length - 1) {
         int temp[] = new int[stck.length * 2]; // Making a double size array.
         for (int i = 0; i < stck.length; i++)
            temp[i] = stck[i];
         stck = temp;
         stck[++tos] = item;
      } else
         stck[++tos] = item;
   }

   // Pop an item from the stack.
   public int pop() {
      if (tos < 0) {
         System.out.println("Stack underflow.");
         return 0;
      } else
         return stck[tos--];
   }
}

class ELClass1 {
   public static void main(String args[]) {
      DynStack mystack1 = new DynStack(5);
      DynStack mystack2 = new DynStack(8);

      // These loops cause each stack to grow.
      for (int i = 0; i < 12; i++)
         mystack1.push(i);
```

```
        for (int i = 0; i < 20; i++)
            mystack2.push(i);

        System.out.println("Stack in mystack1:");
        for (int i = 0; i < 12; i++)
            System.out.println(mystack1.pop());

        System.out.println("Stack in mystack2:");
        for (int i = 0; i < 20; i++)
            System.out.println(mystack2.pop());
         System.out.println("Abhay Raj, 00976803122" );
    }
}
```

Output:



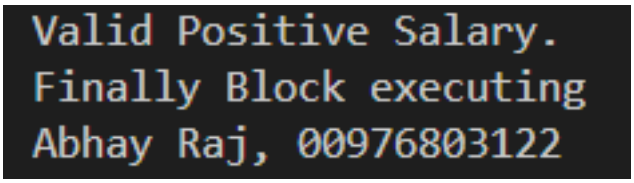
Program – 13

Aim: Design a java program to create a customized exception and also make use of all 5 exception keywords.

Code:

```
class NSalException extends Exception {
   public NSalException(String s) {
      super(s); // CALLING CONSTRUCTOR Exception()
   }
}

class PSalException extends RuntimeException {
   public PSalException(String s) {
      super(s); // CALLING CONSTRUCTOR RuntimeException()
   }
}

class Employee {
   public void decideSal(String s1) throws NSalException, PSalException,
NumberFormatException {
      int sal = Integer.parseInt(s1);
      if (sal <= 0) {
         NSalException no = new NSalException("Invalid Salary");
         throw(no);
      } else {
         PSalException po = new PSalException("Valid Salary");
         throw(po);
      }
   }
}

class MyELClass {
   public static void main(String args[]) {
      try {
         Employee e = new Employee();
         e.decideSal(args[0]);
      } catch (NumberFormatException nfe) {
         System.err.println("Please enter integer values.");
      } catch (NSalException no) {
         System.err.println("Negative Salary.");
      } catch (PSalException po) {
         System.err.println("Valid Positive Salary.");
      } finally {
         System.out.println("Finally Block executing");
      }
     System.out.println("Abhay Raj, 00976803122" );
   }
}
```

Output:


Valid Positive Salary.
Finally Block executing
Abhay Raj, 00976803122

Program – 14

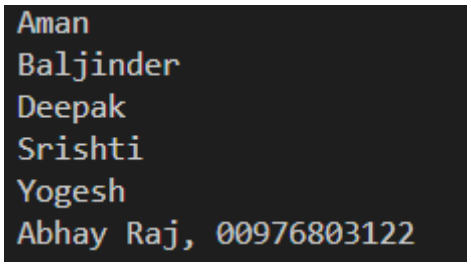
Aim: Design a java program to sort list of names in ascending order.

Code:

```
import java.util.*;

class SortNames {
   String name[] = new String[5];
   int i, n = 5;

   SortNames() {
      for (i = 0; i < n; i++) {
         name[i] = null;
      }
   }

   void inputNames() {
      name[0] = "Yogesh";
      name[1] = "Deepak";
      name[2] = "Aman";
      name[3] = "Srishti";
      name[4] = "Baljinder";
   }

   void bubbleSort() {
      int j;
      String temp;
      for (i = 0; i < n; i++) {
         for (j = 0; j < n - 1 - i; j++) {
            if (name[j].compareTo(name[j + 1]) > 0) {
               temp = name[j];
               name[j] = name[j + 1];
               name[j + 1] = temp;
            }
         }
      }
   }

   void displayNames() {
      for (i = 0; i < n; i++) {
         System.out.println(name[i]);
      }
   }
}

class MyELClass {
   public static void main(String a[]) {
      SortNames sn = new SortNames();
      sn.inputNames();
```

```
        sn.bubbleSort();
        sn.displayNames();
        System.out.println("Abhay Raj, 00976803122" );
    }
}
```

Output:



Program – 15

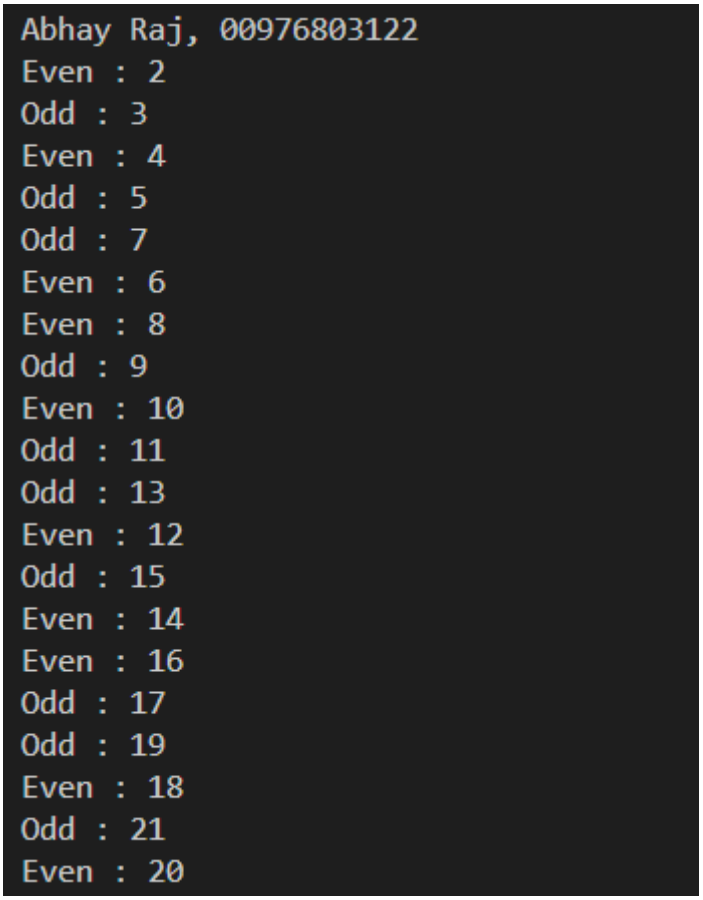
Aim: Design a java program to create two threads: one for odd numbers and other for even numbers.

Code:

```
class EvenThread extends Thread {
   int num;

   void set() {
      this.num = 0;
   }

   public void run() {
      for (int i = 1; i <= 10; i++) {
         try {
            even();
            Thread.sleep(500);
         } catch (InterruptedException e) {
            System.out.println("INTERRUPTED EXCEPTION ");
         }
      }
   }

   void even() {
      num = num + 2;
      System.out.println("Even : " + num);
   }
}

class OddThread implements Runnable {
   int num;

   void set() {
      this.num = 1;
   }

   public void run() {
      for (int i = 1; i <= 10; i++) {
         try {
            odd();
            Thread.sleep(500);
         } catch (InterruptedException e) {
            System.out.println("INTERRUPTED EXCEPTION ");
         }
      }
   }

   void odd() {
      num = num + 2;
      System.out.println("Odd : " + num);
```

```
    }
}

class JavaLab {
    public static void main(String a[]) {
        int i;
        EvenThread et = new EvenThread();
        OddThread ot = new OddThread();
        Thread t = new Thread(ot);
        et.set();
        ot.set();
        et.start();
        t.start();
        System.out.println("Abhay Raj, 00976803122" );
    }
}
```

Output:



Program – 16

Aim: Design a java program to implement producer consumer problem using concept of inter thread communication.

Code:

```
class Q {
   int n;
   boolean valueSet = false;

   synchronized int get() {
      while (valueSet == false)
         try {
            wait();
         } catch (InterruptedException e) {
            System.out.println("InterruptedException caught");
         }
      System.out.println("Got by Consumer: " + n);
      valueSet = false;
      notify();
      return n;
   }

   synchronized void put(int n) {
      while (valueSet == true)
         try {
            wait();
         } catch (InterruptedException e) {
            System.out.println("InterruptedException caught");
         }
      this.n = n;
      valueSet = true;
      System.out.println("Put by Producer: " + n);
      notify();
   }
}

class Producer implements Runnable {
   Q q;

   Producer(Q q) {
      this.q = q;
      new Thread(this, "Producer").start();
   }

   public void run() {
      int i = 0;
      while (true) {
         q.put(i++);
      }
   }
```

```
}

class Consumer implements Runnable {
   Q q;

   Consumer(Q q) {
      this.q = q;
      new Thread(this, "Consumer").start();
   }

   public void run() {
      while (true) {
         q.get();
      }
   }
}

class MyELClass {
   public static void main(String args[]) {
      Q q = new Q();
      new Producer(q);
      new Consumer(q);
      System.out.println("Press Control-C to stop.");
      System.out.println("Abhay Raj, 00976803122" );
   }
}
```

Output:

```
Press Control-C to stop.
Abhay Raj, 00976803122
Put by Producer: 0
Got by Consumer: 0
Put by Producer: 1
Got by Consumer: 1
Put by Producer: 2
Got by Consumer: 2
Put by Producer: 3
Got by Consumer: 3
Put by Producer: 4
Got by Consumer: 4
Put by Producer: 5
Got by Consumer: 5
Put by Producer: 6
Got by Consumer: 6
Put by Producer: 7
Got by Consumer: 7
Put by Producer: 8
Got by Consumer: 8
Put by Producer: 9
Got by Consumer: 9
Put by Producer: 10
Got by Consumer: 10
Put by Producer: 11
```

Program – 17

Aim: Design a java program to create three threads runnable interface. First thread display "GOOD MORNING" at every 1 second. Second thread displays "HELLO" at every 2 seconds and third thread displays "WELCOME" at every 3 seconds.

Code:

```
class Callme {
   void call(String msg, int t) {
      System.out.println(msg);
      try {
         Thread.sleep(t * 1000);
      } catch (InterruptedException e) {
         System.out.println("Interrupted");
      }
   }
}

class Caller implements Runnable {
   String msg;
   int time;
   Callme target;
   Thread t;

   public Caller(Callme targ, String s, int et) {
      target = targ;
      msg = s;
      time = et;
      t = new Thread(this);
      t.start();
   }

   public void run() {
      int i;
      for (i = 1; i <= 5; i++) {
         synchronized (target) {
            target.call(msg, time);
         }
      }
   }
}

class MyELClass {
   public static void main(String args[]) {
      Callme target = new Callme();
      Caller ob1 = new Caller(target, "GOOD MORNING", 1);
      Caller ob2 = new Caller(target, "HELLO", 2);
      Caller ob3 = new Caller(target, "WELCOME", 3);
      try {
         ob1.t.join();
         ob2.t.join();
```

```
            ob3.t.join();
        } catch (InterruptedException e) {
            System.out.println("Interrupted");
        }
      System.out.println("Abhay Raj, 00976803122" );
    }
}
```

Output:

Program – 18

Aim: Design a java program to create three threads. First thread display "GOOD MORNING" at every 1 second. Second thread displays "HELLO" at every 2 seconds and third thread displays "WELCOME" at every 3 seconds.

Code:

```
class DisplayMessage extends Thread {
    private String message;
    private int interval;
    private int count;
    public DisplayMessage(String message, int interval) {
        this.message = message;
        this.interval = interval;
        this.count = 0;
    }
    public void run() {
        while (count < 5) { // Run for five times
            System.out.println(message);
            try {
                Thread.sleep(interval * 1000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
            count++;
        }
    }
}
public class MyELClass {
    public static void main(String[] args) {
        DisplayMessage thread1 = new DisplayMessage("GOOD MORNING", 1);
        DisplayMessage thread2 = new DisplayMessage("HELLO", 2);
        DisplayMessage thread3 = new DisplayMessage("WELCOME", 3);
        thread1.start();
        thread2.start();
        thread3.start();
        System.out.println("Abhay Raj, 00976803122" );
    }
}
```

Output:

```
Abhay Raj, 00976803122
GOOD MORNING
HELLO
WELCOME
GOOD MORNING
HELLO
GOOD MORNING
WELCOME
GOOD MORNING
HELLO
GOOD MORNING
WELCOME
HELLO
HELLO
WELCOME
WELCOME
```