Laboratory Work

Subject: Java Technologies
Branch: B.Tech. (CE)
Semester: IV
Batch: A1

Student Roll No: CE 005

Student Name: Chaudhari krish L.

Department of Computer Engineering,
Faculty of Technology,
Dharmsinh Desai University, Nadiad – 387001.
Gujarat, INDIA.

LAB 1

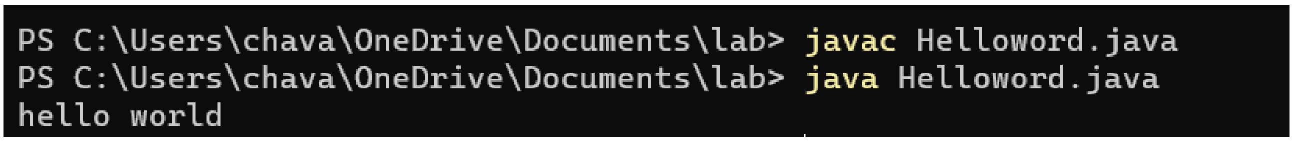
Topics: print(), println(), Scanner class, 1-D, 2-D array, jagged array

1. Write a Java program to display “Hello World”.

Code:

```
class Helloword{
        public static void main(String [] args)
        {
         System.out.println("hello world");
        }

}
```

Output:



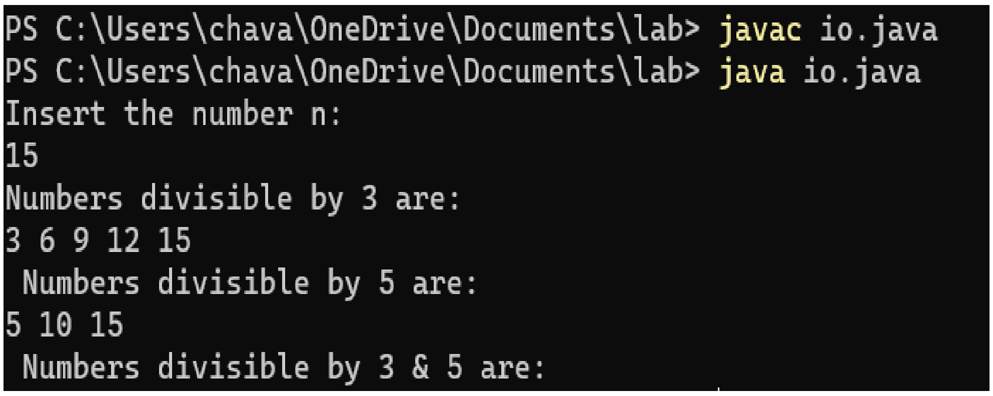
```
PS C:\Users\chava\OneDrive\Documents\lab> javac Helloword.java
PS C:\Users\chava\OneDrive\Documents\lab> java Helloword.java
hello world
```

2. Write a Java program to print numbers between 1 to n which are divisible by 3, 5 and
   By both (3 and 5) by taking n as an input from the user.

Code:

```
import java.util.Scanner;

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

        System.out.println("Insert the number n:");
        int n = Scanner.nextInt();
        Scanner.close();

        System.out.println("Numbers divisible by 3 are:");
        for(int i=1;i<=n;i++){
            if(i % 3 == 0){
                System.out.print(i + " ");
            }
        }

        System.out.println("\n Numbers divisible by 5 are:");
        for(int i=1;i<=n;i++){
            if(i % 5 == 0){
                System.out.print(i + " ");
            }
        }

        System.out.println("\n Numbers divisible by 3 & 5 are:");
        for(int i=1;i<n;i++){
            if(i % 3 == 0 && i % 5 == 0){
                System.out.print(i+ " ");
            }
        }
    }
}
```

Output:



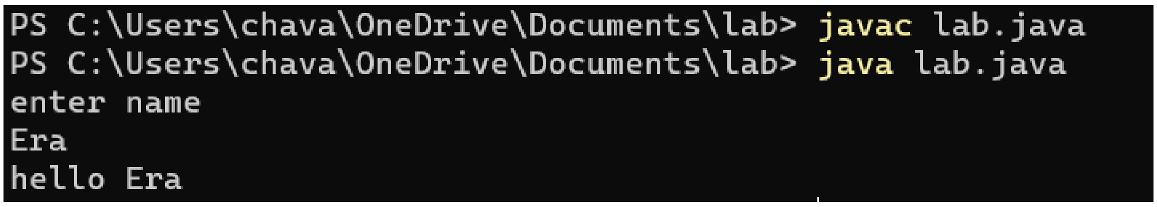
```
PS C:\Users\chava\OneDrive\Documents\lab> javac io.java
PS C:\Users\chava\OneDrive\Documents\lab> java io.java
Insert the number n:
15
Numbers divisible by 3 are:
3 6 9 12 15
 Numbers divisible by 5 are:
5 10 15
 Numbers divisible by 3 & 5 are:
```

3. Write a class named Greeter that prompts the user for his or her name, and then prints
a personalized greeting. As an example, if the user entered &quot;Era&quot;, the program should
respond &quot;Hello Era!&quot;.

Code:

```
class Greeter{

    public static void main(String [] args)
    {

       Scanner obj=new Scanner(System.in);
       System.out.println("enter name");
       String name=obj.nextLine();

       System.out.println("hello"+name);

    }
  }
```

Output:



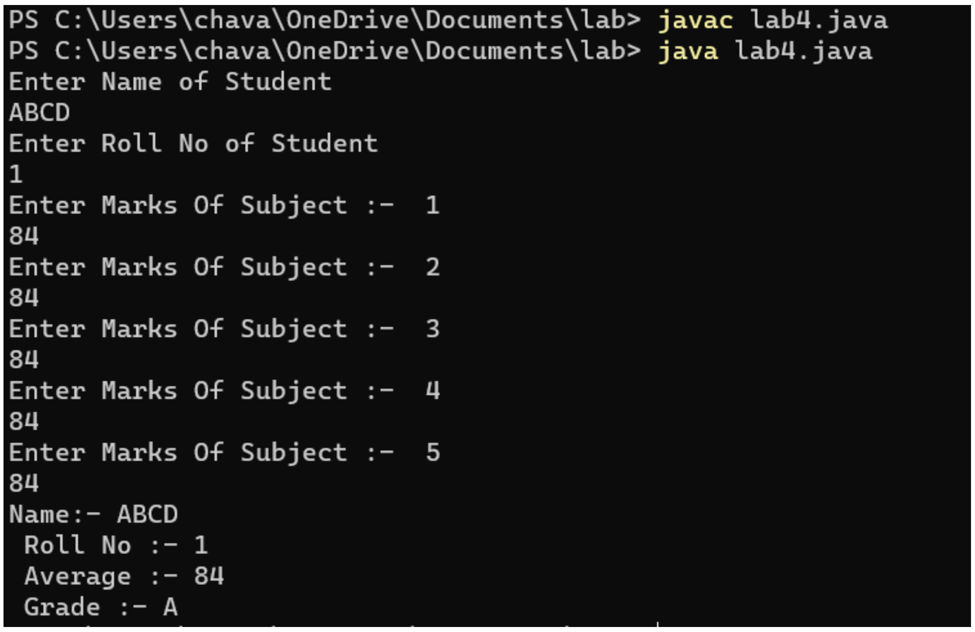
```
PS C:\Users\chava\OneDrive\Documents\lab> javac lab.java
PS C:\Users\chava\OneDrive\Documents\lab> java lab.java
enter name
Era
hello Era
```

4. Write a Java program that takes Name, Roll No and marks of 5 subjects as input and
Gives a formatted output as:
Name: ABCD
Roll No. : 1
Average: 84
Also display the grade (e.g. A, B, C…etc.) using the average.

Code:

```
import java.util.*;
class Prog4
{
        public static void main(String[] args) {
                Scanner sc = new Scanner(System.in);
                        System.out.println("Enter Name of Student ");
                String name = sc.nextLine();
                                        System.out.println("Enter Roll No of Student
");

                int RollNo = sc.nextInt();
                int mrk[]= new int[5];

                for(int i =0;i<5;i++){
                        System.out.println("Enter Marks Of Subject :-  "+(i+1));
                        mrk[i]=sc.nextInt();
                }

                int avg = (mrk[0]+mrk[1]+mrk[2]+mrk[3]+mrk[4])/5;
      char grade='F';
      if(avg >= 8){
         grade = 'A';
      }else if(avg >= 50){
         grade = 'B';
      }else if(avg >35){
         grade = 'C';
      }
                System.out.println("Name:- "+name+ "\n Roll No :- "+RollN+"\n
Average :- "+avg+"\n Grade :- "+grade);

}
}
```

Output:



```
PS C:\Users\chava\OneDrive\Documents\lab> javac lab4.java
PS C:\Users\chava\OneDrive\Documents\lab> java lab4.java
Enter Name of Student
ABCD
Enter Roll No of Student
1
Enter Marks Of Subject :-  1
84
Enter Marks Of Subject :-  2
84
Enter Marks Of Subject :-  3
84
Enter Marks Of Subject :-  4
84
Enter Marks Of Subject :-  5
84
Name:- ABCD
 Roll No :- 1
 Average :- 84
 Grade :- A
```

5. Calculate and return the sum of all the even numbers present in the numbers array passed to the method calculateSumOfEvenNumbers. Implement the logic inside calculateSumOfEvenNumbers() method.Test the functionalities using the main() method of the Tester class.

Sample Input and Output:

Sample Input	**Sample Output**
{68,79,86,99,23,2,41,100}	256
{1,2,3,4,5,6,7,8,9,10}	30

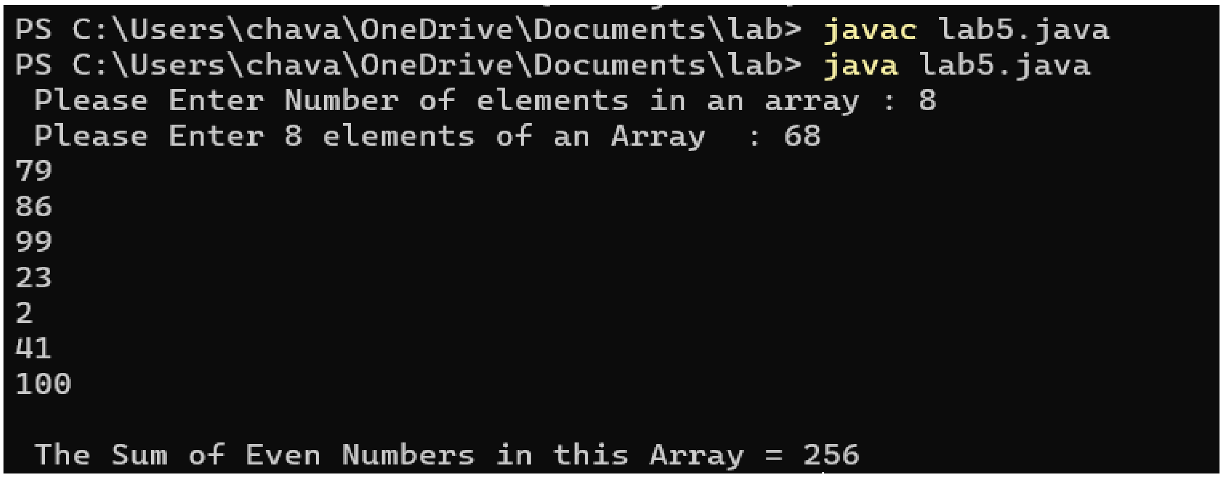
Code:

```
import java.util.Scanner;

 class SumOfEvens1 {

        public static void main(String[] args)
        {
                int Size, i, EvenSum = 0;
                Scanner sc = new Scanner(System.in);

                System.out.print(" Please Enter Number of elements in an array : ");
                Size = sc.nextInt();

                int [] a = new int[Size];

                System.out.print(" Please Enter " + Size + " elements of an Array  : ");
                for (i = 0; i < Size; i++)
                {
                        a[i] = sc.nextInt();
                }

                for(i = 0; i < Size; i++)
                {
                        if(a[i] % 2 == 0)
                        {
                                EvenSum = EvenSum + a[i];
                        }
                }
                System.out.println("\n The Sum of Even Numbers in this Array = " +
EvenSum);
        }
```

}

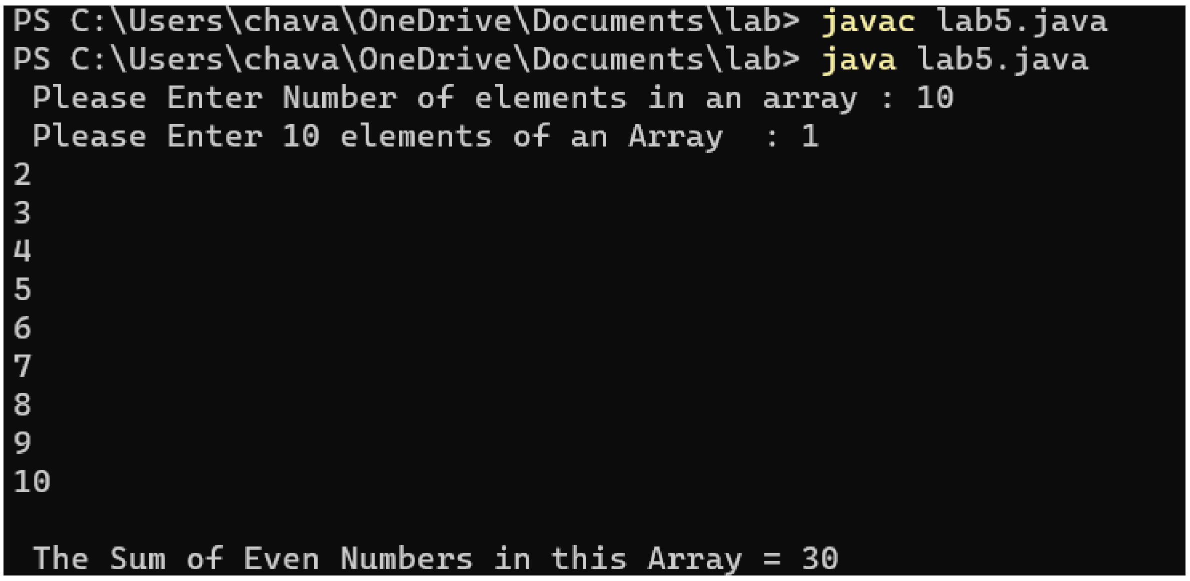
Output:

Sample input 1:



```
PS C:\Users\chava\OneDrive\Documents\lab> javac lab5.java
PS C:\Users\chava\OneDrive\Documents\lab> java lab5.java
 Please Enter Number of elements in an array : 8
 Please Enter 8 elements of an Array  : 68
79
86
99
23
2
41
100

 The Sum of Even Numbers in this Array = 256
```

Sample input 2:



```
PS C:\Users\chava\OneDrive\Documents\lab> javac lab5.java
PS C:\Users\chava\OneDrive\Documents\lab> java lab5.java
 Please Enter Number of elements in an array : 10
 Please Enter 10 elements of an Array  : 1
2
3
4
5
6
7
8
9
10

 The Sum of Even Numbers in this Array = 30
```

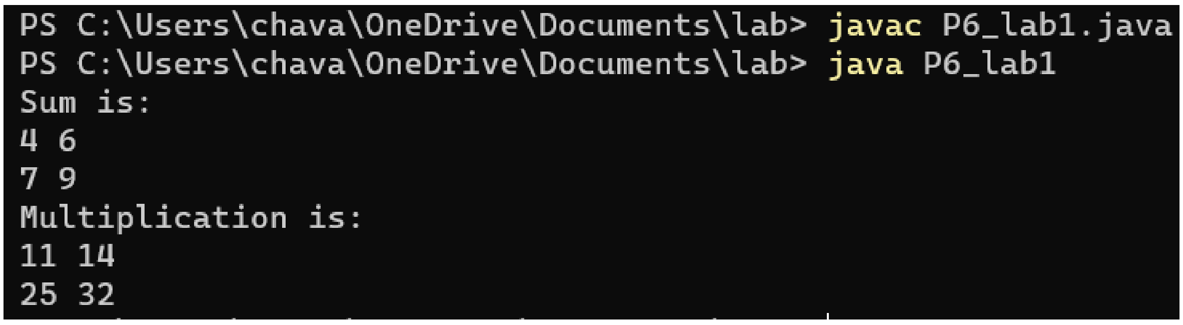
6. Write a program to perform matrix addition and matrix multiplication on two given matrices. Use for-each form of for loop to display the matrices.

Code:

```
class Calc{
	void sum(int a[][], int b[][])
	{
		int s[][] = new int [a.length][b[0].length];
		for(int i=0 ; i<a.length ; i++)
		{
			for(int j=0 ; j<b.length ; j++)
			{
				s[i][j] = a[i][j]+b[i][j];
			}
		}

		for(int[] i : s)
		{
			for(int j : i)
			{
				System.out.print(j+" ");
			}
			System.out.println();
		}
	}

	void multiplication(int a[][], int b[][])
	{
		int m[][] = new int [a.length][b[0].length];
		for(int i=0; i<a.length ; i++)
		{
			for(int j=0 ; j<b[0].length ; j++)
			{
				for(int k=0 ; k<b.length ; k++)
				{
					m[i][j] += a[i][k]*b[k][j];
				}
			}
		}

		for(int[] i : m)
		{
			for(int j : i)
```

```
                        {
                                System.out.print(j+" ");
                        }
                        System.out.println();
                }
        }
}
class P6_lab1{
        public static void main(String[] args)
        {
                int arr1[][] ={
                        {1,2},
                        {3,4}};
                int arr2[][] ={
                        {3,4},
                        {4,5}};
                System.out.println("Sum is: ");
                new Calc().sum(arr1,arr2);
                System.out.println("Multiplication is: ");
                new Calc().multiplication(arr1,arr2);
        }
}
```

Output:



```
PS C:\Users\chava\OneDrive\Documents\lab> javac P6_lab1.java
PS C:\Users\chava\OneDrive\Documents\lab> java P6_lab1
Sum is:
4 6
7 9
Multiplication is:
11 14
25 32
```

Laboratory Work

Subject: Java Technologies
Branch: B.Tech. (CE)
Semester: IV
Batch: IV

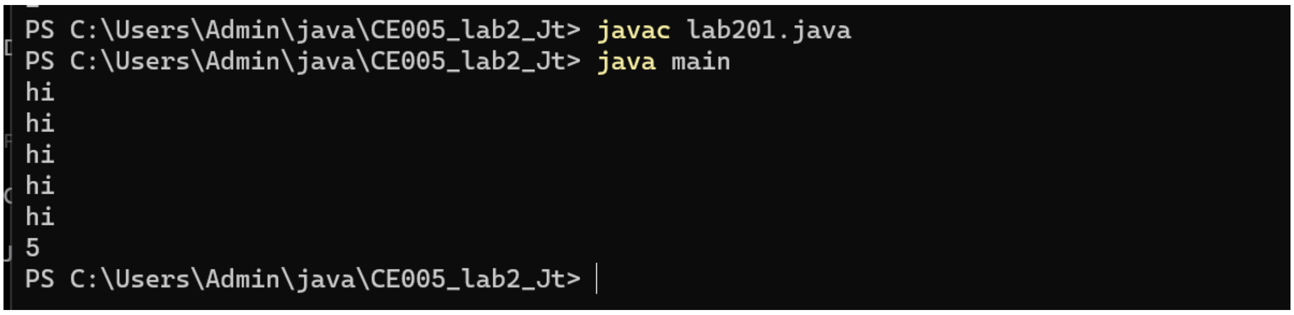
Student Roll No: CE005
Student Name: Chaudhari Krish Lalbhai

Department of Computer Engineering,
Faculty of Technology,
Dharmsinh Desai University,
Nadiad – 387001.
Gujarat, INDIA.

LAB-2

Que1 :-**Write a program that returns the number of times that the string "hi" appears anywhere in the given string**.

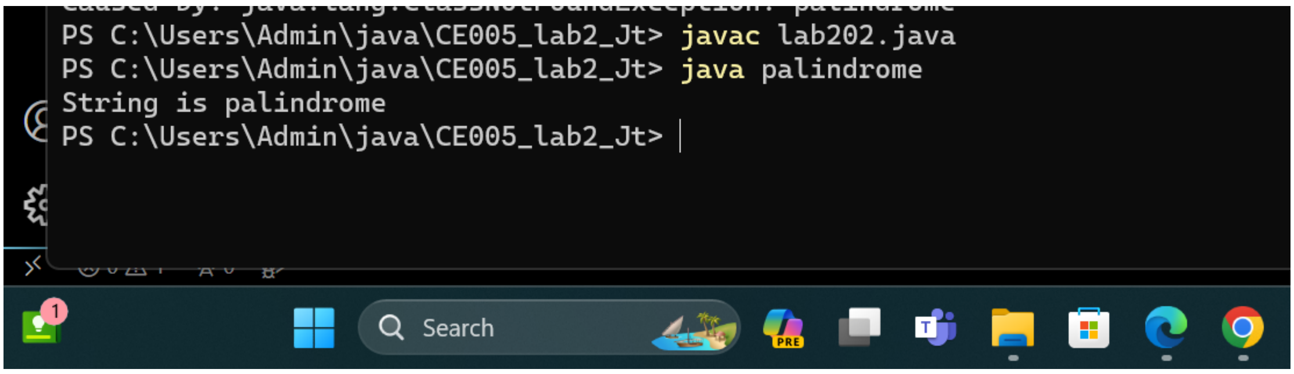
```
class main{
 public static void main(String[] args){
 String s="hi hi hii";
 String key="hi";
 int cnt=0;
 String[] tokens=s.split(" ");
 for(String :tokens){
System.out.println(t); if(t.equals(key)){ cnt++; } }
System.out.println(cnt); } }
```



```
PS C:\Users\Admin\java\CE005_lab2_Jt> javac lab201.java
PS C:\Users\Admin\java\CE005_lab2_Jt> java main
hi
hi
hi
hi
hi
5
PS C:\Users\Admin\java\CE005_lab2_Jt>
```

Que 2 : - **Write a program which checks whether the input string is palindrome or not and then**
display an appropriate message [e.g. “Refer” is a palindrome string].

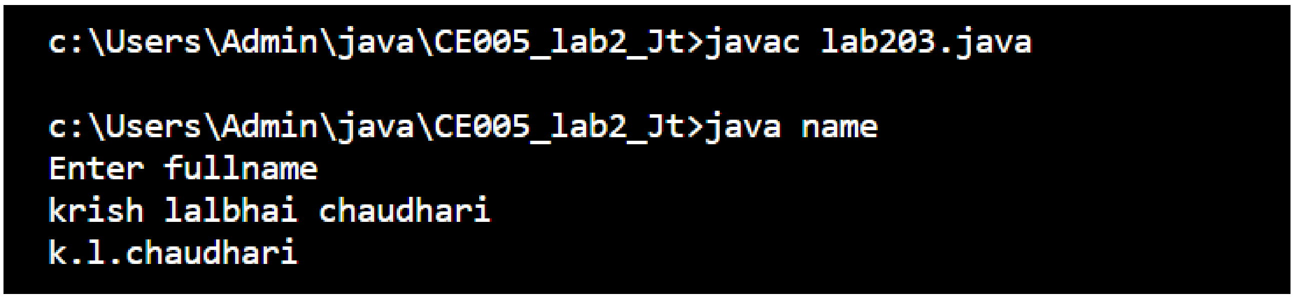
```
import java.lang.*;
import java.io.*;
import java.util.*;
class main{
public static boolean isPalindrome(String s){
StringBuilder input1 = new StringBuilder();
input1.append(s);
input1.reverse();
String r= input1.toString();
boolean ans=false;
if(s.equals(r)){
ans=true;
}
return ans;
}
public static void main(String[] args){
String s="abbA";
s=s.toLowerCase();
boolean ans=isPalindrome(s);
if(ans==true){
System.out.println("String is palindrome");
}
else
System.out.println("String is not palindrome");
}
}
```



```
PS C:\Users\Admin\java\CE005_lab2_Jt> javac lab202.java
PS C:\Users\Admin\java\CE005_lab2_Jt> java palindrome
String is palindrome
PS C:\Users\Admin\java\CE005_lab2_Jt>
```

Que3:- **Write a program that takes your full name as input and displays the abbreviations of the**
first and middle names except the last name which is displayed as it is. For example, if
your name is Robert Brett Roser, then the output should be R.B.Roser.

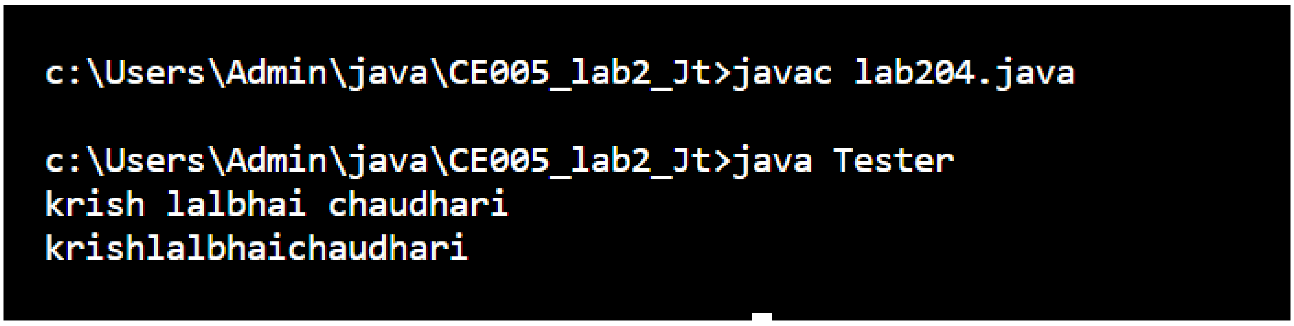
```
import java.lang.*;
import java.io.*;
 import java.util.*;
 class name{
 public static void main(String[] args){
 Scanner myObj = new Scanner(System.in);
 System.out.println("Enter fullname");
String name = myObj.nextLine();
 String[] token = name.split(" ");
 String output = new String();
output=token[0].charAt(0)+"."+token[1].charAt(0)+"."+token[2];
System.out.println(output);
 }
 }
```



```
c:\Users\Admin\java\CE005_lab2_Jt>javac lab203.java

c:\Users\Admin\java\CE005_lab2_Jt>java name
Enter fullname
krish lalbhai chaudhari
k.l.chaudhari
```

Que4 :**Write a method String removeWhiteSpaces(String str) method that removes all the**
white spaces from the string passed to the method and returns the modified string. Test
the functionalities using the main() method of the Tester class.

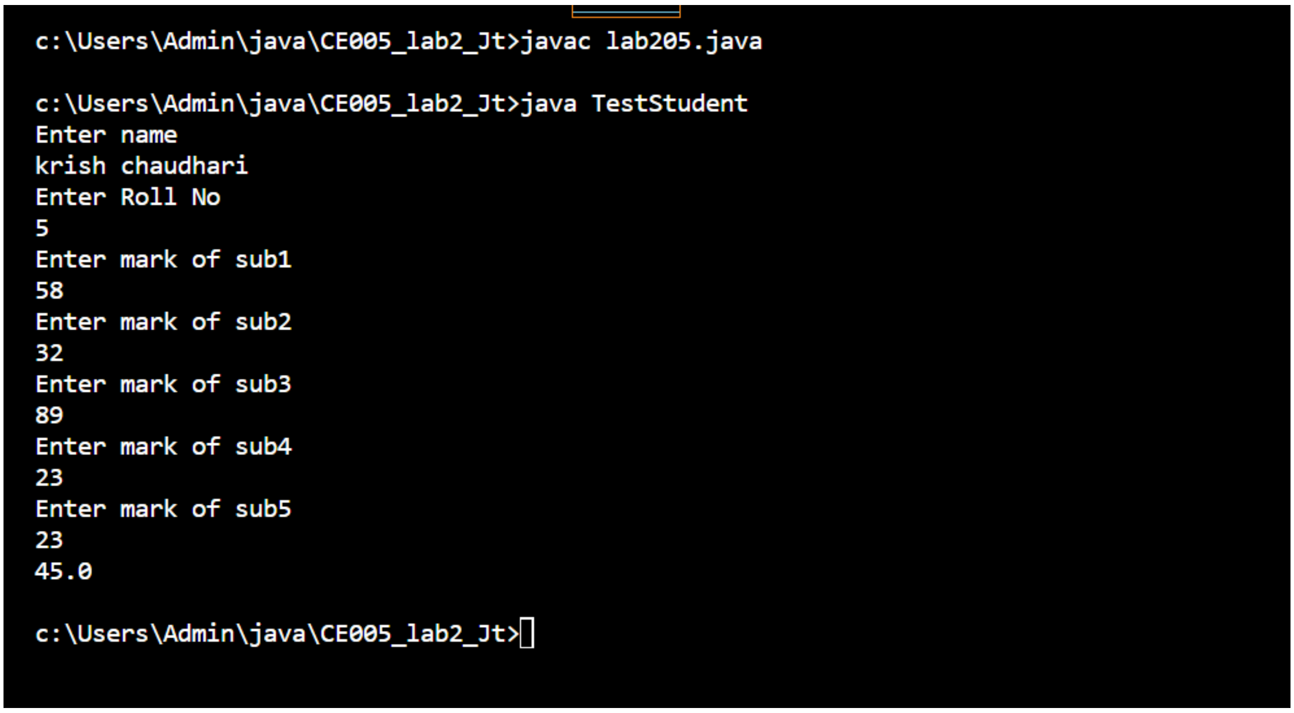
```
import java.lang.*;
 import java.io.*;
 import java.util.*;
 class Remove{
String removeWhiteSpaces(String str){
 String[] tokens = str.split(" ");
 StringBuilder output= new StringBuilder();
for(String t:tokens){
 output.append(t);
 }

String ans=output.toString();
return ans;
}
 }
 class Tester{
 public static void main(String[] args){
 Scanner myObj = new Scanner(System.in);
 String s = myObj.nextLine();
Remove obj1 = new Remove();
 String sp=obj1.removeWhiteSpaces(s);
 System.out.println(sp);
 }
 }
```



```
c:\Users\Admin\java\CE005_lab2_Jt>javac lab204.java

c:\Users\Admin\java\CE005_lab2_Jt>java Tester
krish lalbhai chaudhari
krishlalbhaichaudhari
```

Que5 : **Write a class Student with member variables int roll_no, String name and an array to store**
marks of 5 subjects. Demonstrate constructor overloading and use this keyword. Write a
findAverage() method that returns double value. Write a TestStudent class containing
main() method to do the following:
a) Store the details of one student by creating one object of Student class and display
them.
b) Store the details of 3 students by creating an array of objects of Student class and
display the details of the student who has the highest average amongst the three students.

```
import java.util.Scanner;
 class Student{
 int roll_no;
String name;
 int[] marks= new int[5];
 Student(){
}
Student(int roll,String name, int[] marks){
roll_no=roll;
 this.name=name;
this.marks=marks;
 }
double findAvg(){
 double sum=0;
 for(int i=0;i<5;i++){
 sum+=marks[i];
 }
return sum/5;
 }
 }
 class TestStudent{
 public static void main(String[] args){
 int roll; String name;
int[] marks= new int[5];
 Scanner myObj = new Scanner(System.in);
 System.out.println("Enter name");
name = myObj.nextLine();
System.out.println("Enter Roll No");
 roll = myObj.nextInt();
 for(int i=1;i<=5;i++){
 System.out.println("Enter mark of sub" + i);
 marks[i-1] = myObj.nextInt();
 }
 Student s= new Student(roll,name,marks);
 System.out.println(s.findAvg());
 }
}
```



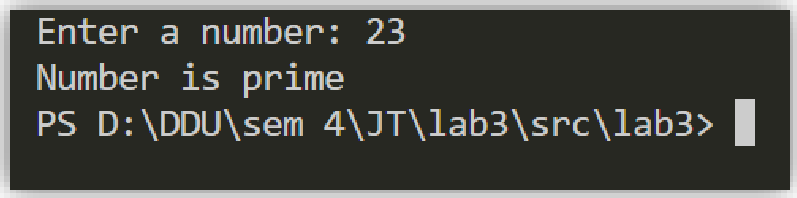
```
c:\Users\Admin\java\CE005_lab2_Jt>javac lab205.java

c:\Users\Admin\java\CE005_lab2_Jt>java TestStudent
Enter name
krish chaudhari
Enter Roll No
5
Enter mark of sub1
58
Enter mark of sub2
32
Enter mark of sub3
89
Enter mark of sub4
23
Enter mark of sub5
23
45.0

c:\Users\Admin\java\CE005_lab2_Jt>
```

LAB-3

1. Write a Java program that checks for prime number using the object oriented approach.

```
package lab3;
import java.util.Scanner;

class NumberClass{
        int value = 0;
        NumberClass(int value)
        {
                this.value = value;
        }

        public void prime()
        {
                if(value == 0 || value == 1)
                {
                        System.out.println("O/1 are unique number!");
                        return;
                }
                else
                {
                        for(int i=2; i<=value/2; i++)
                        {
                                if(value % i == 0)
                                {
                                        System.out.println("Number is not prime");
                                        break;
                                }
                                else
                                {
                                        System.out.println("Number is prime");
                                        break;
                                }
                        }
                }
        }
}


public class P1 {

        public static void main(String[] args) {

                Scanner sc = new Scanner(System.in);
                System.out.print("Enter a number: ");
                int num = sc.nextInt();

                NumberClass obj = new NumberClass(num);
                obj.prime();
        }

}
```

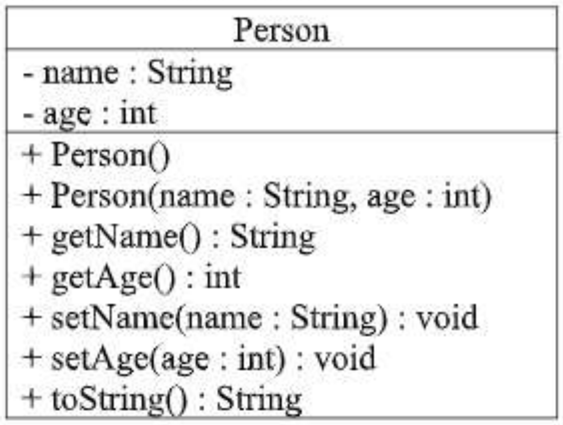
Output:-



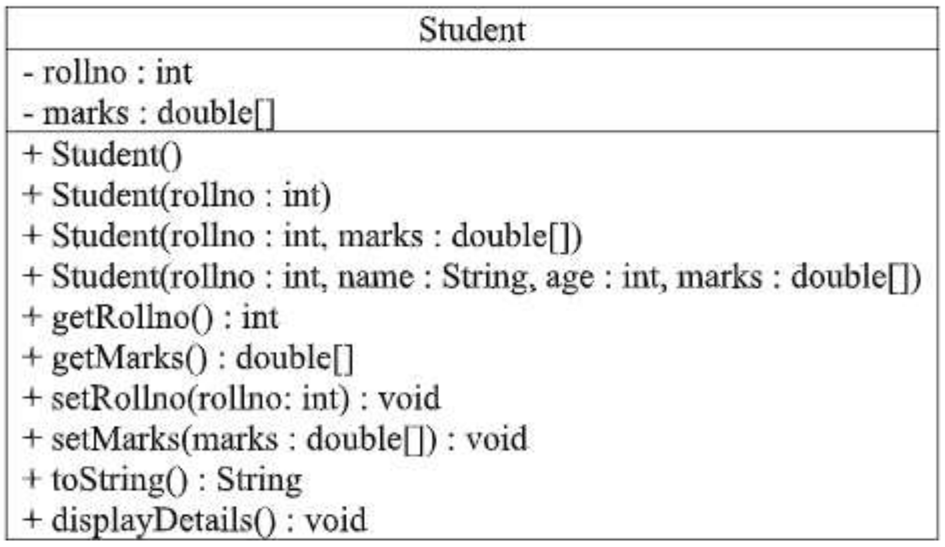
```
Enter a number: 23
Number is prime
PS D:\DDU\sem 4\JT\lab3\src\lab3>
```

2.

Derive a class Student from class Person.



```
Person
- name : String
- age : int
+ Person()
+ Person(name : String, age : int)
+ getName() : String
+ getAge() : int
+ setName(name : String) : void
+ setAge(age : int) : void
+ toString() : String
```



```
Student
- rollno : int
- marks : double[]
+ Student()
+ Student(rollno : int)
+ Student(rollno : int, marks : double[])
+ Student(rollno : int, name : String, age : int, marks : double[])
+ getRollno() : int
+ getMarks() : double[]
+ setRollno(rollno: int) : void
+ setMarks(marks : double[]) : void
+ toString() : String
+ displayDetails() : void
```

Add the following to Student class:

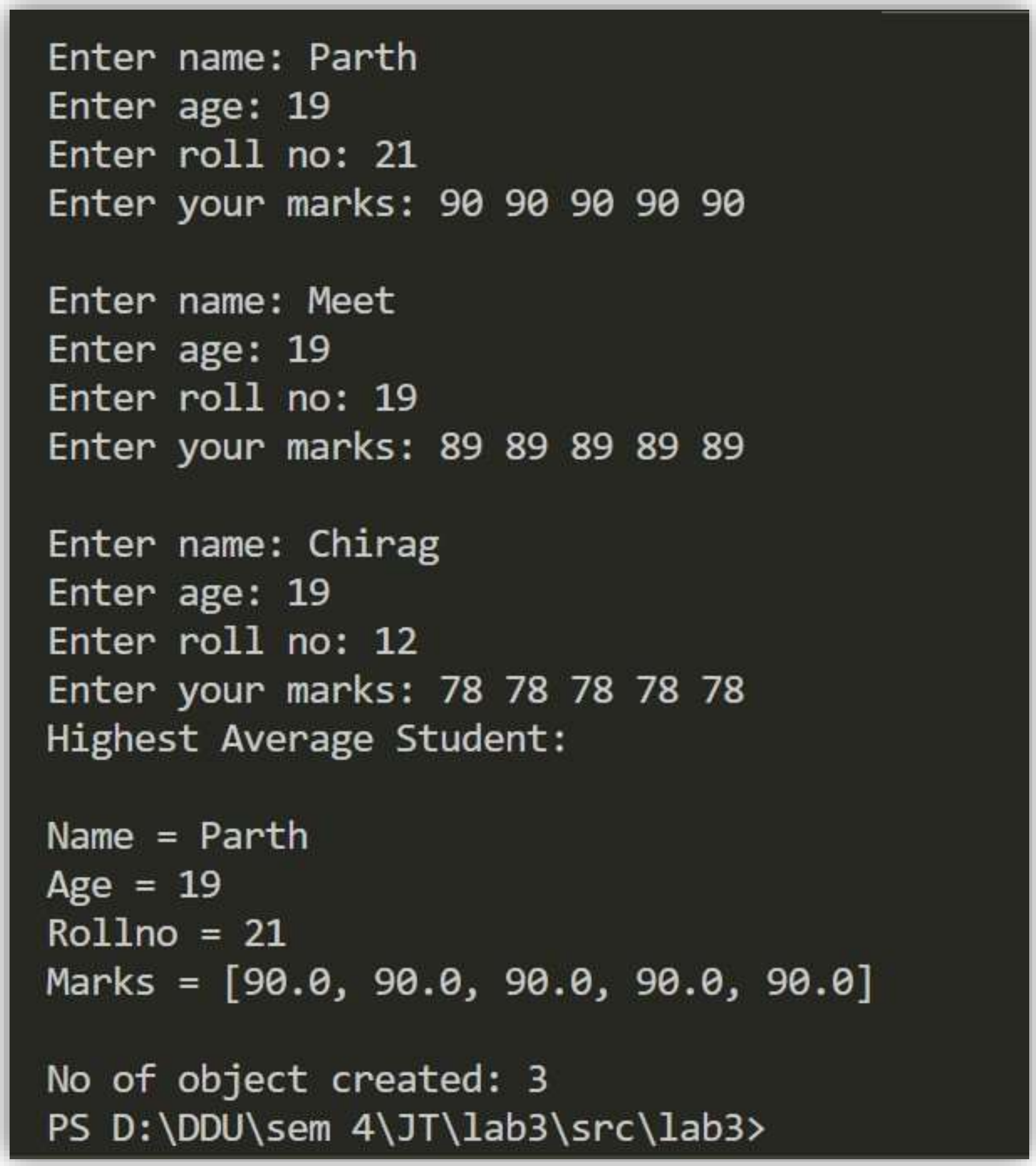
- a static variable count(to count the number of objects)
- a static block to initialize count variable to zero
- a static method String getCount() that returns the no of student objects ceated
- Write a TestStudent class containing the main() method.
- Store the details of 3 students by creating an array of objects of Student class and display the student who has highest average amongst the three students as follows using displayDetails() method for that object:
  e.g.
  RollNo = 100
  Name = ABC
  Age = 20
  Marks=78 86 88 67 92
- Create one more object of the Student class and then call the getCount() to display the number of Student objects created.

```
package lab3;

import java.util.Arrays;
import java.util.Scanner;

class Person{
	private String name;
	private int age;

	public Person()
	{}

	public Person(String name, int age) {
		this.name = name;
		this.age = age;
	}

	public String getName() {
		return name;
	}

	public void setName(String name) {
		this.name = name;
	}

	public int getAge() {
		return age;
	}

	public void setAge(int age) {
		this.age = age;
	}

	@Override
	public String toString() {
		return "\nName = " + name + "\nAge = " + age;
	}
}

class Student extends Person{
	private int rollno;
	private double marks[];
	static int count;
	int avg;

	static{ //Static block
```

```
		count = 0;
	}

	{ //initializer block
		count++;
	}

	static String getCount()
	{
		System.out.println("\nNo of object created: "+count);
		return ""+count;
	}
	public Student()
	{}

	public Student(int rollno, double[] marks, String name, int age,int avg) {
		super(name,age);
		this.rollno = rollno;
		this.marks = marks;
		this.avg = avg;
	}

	public int getRollno() {
		return rollno;
	}
	public void setRollno(int rollno) {
		this.rollno = rollno;
	}
	public double[] getMarks() {
		return marks;
	}
	public void setMarks(double[] marks) {
		this.marks = marks;
	}


	@Override
	public String toString() {
		System.out.println(super.toString());
		return "Rollno = " + rollno + "\nMarks = " + Arrays.toString(marks);
	}

	public void displayDetail() {
		System.out.println(toString());
	}
}
```

```
public class P2 {

        public static void main(String[] args) {
                Scanner sc = new Scanner(System.in);
                Student[] arr_obj = new Student[3];
                Student highest = null;
                for(int j=0; j<3; j++)
                {
                        System.out.print("\nEnter name: ");
                        String name = sc.next();
                        System.out.print("Enter age: ");
                        int age = sc.nextInt();
                        System.out.print("Enter roll no: ");
                        int rollno = sc.nextInt();
                        System.out.print("Enter your marks: ");
                        double[] marks = new double[5];
                        int avg = 0;
                        for(int i=0; i<5; i++)
                        {
                                marks[i] = sc.nextDouble();
                                avg += marks[i];
                        }
                        avg/=5;
                        arr_obj[j] = new Student(rollno,marks,name,age,avg);
                        if(highest == null)
                        {
                                highest = arr_obj[j];
                        }
                        else
                        {
                                if(avg > highest.avg)
                                {
                                        highest = arr_obj[j];
                                }
                        }

                }
                System.out.println("Highest Average Student:");
                highest.displayDetail();
                Student.getCount();
        }
}
```

Output:-



```
Enter name: Parth
Enter age: 19
Enter roll no: 21
Enter your marks: 90 90 90 90 90

Enter name: Meet
Enter age: 19
Enter roll no: 19
Enter your marks: 89 89 89 89 89

Enter name: Chirag
Enter age: 19
Enter roll no: 12
Enter your marks: 78 78 78 78 78
Highest Average Student:

Name = Parth
Age = 19
Rollno = 21
Marks = [90.0, 90.0, 90.0, 90.0, 90.0]

No of object created: 3
PS D:\DDU\sem 4\JT\lab3\src\lab3>
```

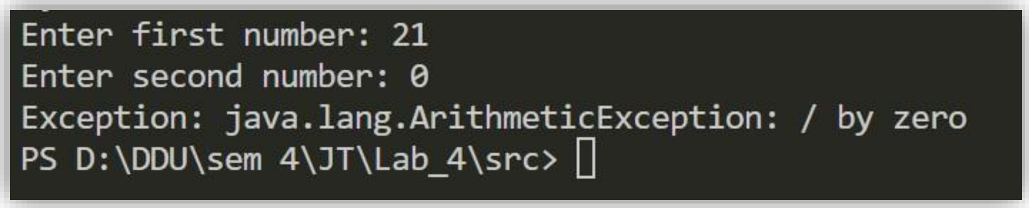
Java Technology

Lab – 4

1. Write a program that catches the divide-by-zero exception using the try-catch mechanism Take a numeric value and perform division by zero. Catch the ArithmeticException.

```
import java.util.Scanner;
public class P1 {

        public static void main(String[] args) {
                try
                {
                        Scanner sc = new Scanner(System.in);
                        System.out.print("Enter first number: ");
                        int num1 = sc.nextInt();
                        System.out.print("Enter second number: ");
                        int num2 = sc.nextInt();
                        num1 = num1/num2;
                        System.out.println("Divison is: " + num1);
                        sc.close();
                }
                catch(ArithmeticException e)
                {
                        System.out.println("Exception: " + e);
                }
        }
}
```

Output:-

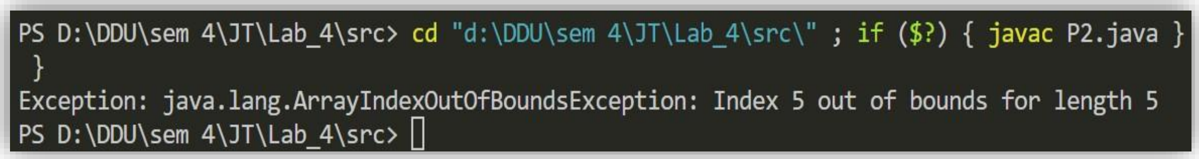


```
Enter first number: 21
Enter second number: 0
Exception: java.lang.ArithmeticException: / by zero
PS D:\DDU\sem 4\JT\Lab_4\src> 
```

2. Write a java program using multiple catch blocks. Create a class CatchExercise, inside the try block declare an array a[] with size of 5 elements and initialize with value a[5] =30/5 . Using Multiple catch blocks handle ArithmeticException and ArrayIndexOutOfBoundsException.

```
public class P2 {

        public static void main(String[] args) {
                try
                {
                        int a[] = new int[5];
                        a[5] = 30/5; // only one catch block will get executed for each try
                        a[0] = 30/0; // a[5] throws error then it will redirected to catch lock
                                      // 30/0 wont get executed
                }
                catch(ArithmeticException e)
                {
                        System.out.println("Exception: " + e);
                }
                catch(ArrayIndexOutOfBoundsException e)
                {
                        System.out.println("Exception: " + e);
                }
        }
}
```

Output:-



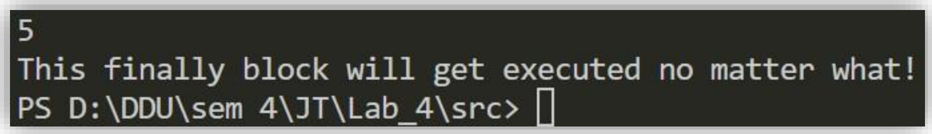
```
PS D:\DDU\sem 4\JT\Lab_4\src> cd "d:\DDU\sem 4\JT\Lab_4\src\" ; if ($?) { javac P2.java }
 }
Exception: java.lang.ArrayIndexOutOfBoundsException: Index 5 out of bounds for length 5
PS D:\DDU\sem 4\JT\Lab_4\src> 
```

3. Write a program that demonstrates use of finally block. Observe the output of your program for different cases as mentioned below.

- Case A: exception does not occur. Perform 25/5 mathematical operation. Catch the NullPointerException.

```
public class P3 {
        public static void main(String[] args)
        {
                try
                {
                        int num = 25/5;
                        System.out.println(num);
                }
                catch(NullPointerException e)
                {
                        System.out.println("Exception: " + e);
                }
                finally
                {
                        System.out.println("This finally block will get executed no matter
what!");
                }
        }
}
```

Output:-

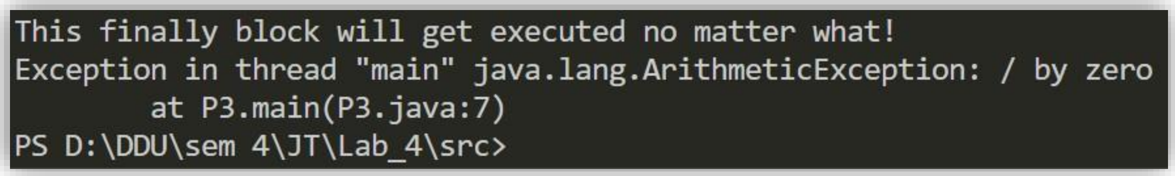


```
5
This finally block will get executed no matter what!
PS D:\DDU\sem 4\JT\Lab_4\src> 
```

- Case B: exception occurs but not handled. Perform 25/0 mathematical operation. Catch NullPointerException.

```
public class P3 {
        public static void main(String[] args)
        {
                try
                {
                        int num = 25/0;
                        System.out.println(num);
                }
                catch(NullPointerException e)
                {
                        System.out.println("Exception: " + e);
                }

                finally
                {
                        System.out.println("This finally block will get executed no matter
what!");
                }
        }
}    // Finally block will be executed and program will be terminated because error
occurred but not handled
```

Output:-

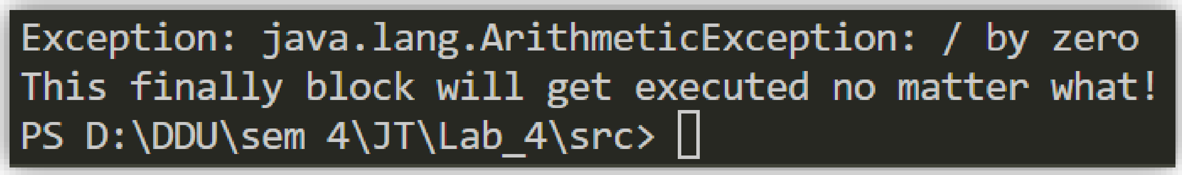


```
This finally block will get executed no matter what!
Exception in thread "main" java.lang.ArithmeticException: / by zero
        at P3.main(P3.java:7)
PS D:\DDU\sem 4\JT\Lab_4\src>
```

- Case C: exception occurs and handled. Perform 25/0 mathematical operation. Catch ArithmeticException

```
public class P3 {
        public static void main(String[] args)
        {
                try
                {
                        int num = 25/0;
                        System.out.println(num);
                }
                catch(ArithmeticException e)
                {
                        System.out.println("Exception: " + e);
                }
                finally
                {
                        System.out.println("This finally block will get executed no
matter what!");
                }
        }
}
```

Output:-



```
Exception: java.lang.ArithmeticException: / by zero
This finally block will get executed no matter what!
PS D:\DDU\sem 4\JT\Lab_4\src> 
```

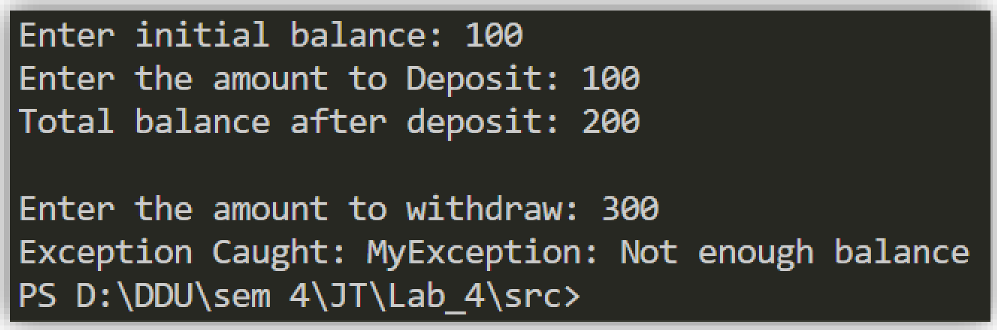
Java Technology

4. Create an interface Account with two methods: deposit and withdraw. Create class SavingsAccount which implements the interface. Write a custom Exception handler class CustomException for SavingsAccount to handle the scenarios when the withdrawn amount is larger than the balance in the account.

```
import java.util.Scanner;
interface Account
{
        int deposit(int amt, int bal);
        int withdraw(int amt, int bal);
}
class MyException extends Exception
{
        public MyException(String message)
        {
                super(message);
        }

        @Override
        public String toString() {
                return "MyException: " + getMessage();
        }

}
class SavingAccount implements Account
{

        @Override
        public int deposit(int amt,int bal) {
                bal += amt;
                System.out.println("Total balance after deposit: " + bal);
                return bal;
        }

        @Override
        public int withdraw(int amt,int bal) {
                try
                {
                        bal -= amt;
                        if(bal < 0)
                        {
                                throw new MyException("Not enough balance");
                        }
                        System.out.println("Total balance after withdrawl: " + bal);

                }
```

```
		catch(MyException e)
		{
			System.out.println("Exception Caught: " + e);
		}
		return bal;
	}
}

public class P4
{
	public static void main(String[] args)
	{
		Scanner sc = new Scanner(System.in);
		SavingAccount a1 = new SavingAccount();
		int amt,bal;

		System.out.print("Enter initial balance: ");
		bal = sc.nextInt();

		System.out.print("Enter the amount to Deposit: ");
		amt = sc.nextInt();
		bal = a1.deposit(amt,bal);

		System.out.print("\nEnter the amount to withdraw: ");
		amt = sc.nextInt();
		bal = a1.withdraw(amt,bal);
		sc.close();
	}
}
```

Output:-



```
Enter initial balance: 100
Enter the amount to Deposit: 100
Total balance after deposit: 200

Enter the amount to withdraw: 300
Exception Caught: MyException: Not enough balance
PS D:\DDU\sem 4\JT\Lab_4\src>
```

Laboratory Work

Subject: Java Technologies
Branch: B.Tech. (CE)
Semester: IV
Batch: A1

Student Roll No: CE005

Student Name: Chaudhari Krish L.

Department of Computer Engineering,
Faculty of Technology,

Dharmsinh Desai University, Nadiad – 387001.

Gujarat, INDIA.

LAB 05

Topics :- Abstract class , Interface , Multithreading

- **AIM: Anchor College offers both UnderGraduate and PostGraduate programs. The college**

stores the names of the students, their test scores and the final result for eachstudent.
Each student has to take 4 tests in total. You need to create an application forthe college
by implementing the classes based on the class diagram and description givenbelow.

```
abstract class Student {

        private String studentName;

        private int[] testScores=new int[4];private
        String testResult;
        Student(){

        }

        public String getStudentName() {
                return studentName;
        }

        public void setStudentName(String studentName) {
                this.studentName = studentName;
        }

        public String getTestResult() {
                return testResult;
        }

        public void setTestResult(String testResult) {
                this.testResult = testResult;
        }

        public Student(String studentName) {
                super();
                this.studentName = studentName;
                for(int i=0;i<4;i++) {
                        testScores[i]=0;

                }

        }

        abstract void generateResult();

        public void setTestScore(int testNumber,int testScore) {
                this.testScores[testNumber-1]=testScore;

        }

        public int[] getTestScores() {
                return testScores;
        }

}



class UndergraduateStudent extends Student{
        UndergraduateStudent(){

        }
        public UndergraduateStudent(String studentName) {
                super(studentName);
                // TODO Auto-generated constructor stub
        }

        void generateResult() {
                double avg_score=0;
                int[] scores=getTestScores();

                for(int i:scores) {

                        avg_score+=i;

                }

                avg_score=avg_score/4;
                if(avg_score>=60) {
                        setTestResult("Pass");

                }

                else {

                }
                                                }

setTestResult("Fail");

 }

 public class GraduateStudent extends Student {
        GraduateStudent(){

        }
```

```
          public  GraduateStudent(String  studentName) {
                    super(studentName);
                    // TODO Auto-generated constructor stub

          }


          void generateResult() {
                    double avg_score=0;
                    int[] scores=getTestScores();


                    for(int i:scores) {

                              avg_score+=i;

                    }

                    avg_score=avg_score/4;
                    if(avg_score>=75) {
                              setTestResult("Pass");

                    }

                    else {


                    }
                                                  }




setTestResult("Fail");


 }


 public class testing {

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

```
                    String name="Jerry";
                    UndergraduateStudent s1=new
UndergraduateStudent(name);
                    s1.setTestScore(1,70);
                    s1.setTestScore(2,69);

                    s1.setTestScore(3,71);
                    s1.setTestScore(4,55);
                    s1.generateResult();
                    String
                    under_res=s1.getTestResult();
                    System.out.println(under_res);

                    String nm="Tom";
                    GraduateStudent  s2=new  GraduateStudent(nm);
                    s2.setTestScore(1,70);
                    s2.setTestScore(2,75);
                    s2.setTestScore(3,80);
                    s2.setTestScore(4,85);
                    s2.generateResult();
                    String
                    post_res=s2.getTestResult();
                    System.out.println(post_res);
               }


     }
```

<u>OUTPUT:</u>

```
Pass
Pass
```

• AIM: Write a Java program as per the given description to demonstrate use of interface.

• Define an interface RelationInterface.

Write three abstract methods: isGreater, isLess and isEqual.
All methods have a return type of boolean and take an argument of type Line with which the caller object will be compared.

• Define the Line class implements the RelationInterface interface.

• It has 4 double variables for the x and y coordinates of the line.

• Define a constructor in Line class that initializes these 4 variables.

• Define a method getLength() that computes length of the line.
[double length = Math.sqrt((x2-x1)*(x2-x1)+(y2-y1)*(y2-y1))].

• Implement the methods of interface in Line class

• In class CompareLines.Java, create two objects of Line class, call the three methods to compare the lengths of the lines.

```
public interface RelationInterface {
        boolean isGreater(Line l);
        boolean isLess(Line l);
        boolean isEqual(Line l);

}

public class Line implements RelationInterface {
        private double x1;
        private double x2;
        private double y1;
        private double y2;
        public Line() {

        }
        public Line(double x1, double x2, double y1, double y2) {super();
                this.x1 = x1;
                this.x2 = x2;
                this.y1  =  y1;
                this.y2 = y2;
        }

        double getLength() {

                double length=Math.sqrt((x2-x1)*(x2-x1)+(y2-y1)*(y2-y1));
                return length;

        }

        public boolean isGreater(Line l) {

                if(this.getLength()>l.getLength()) {
                        return true;

                }

                else {


                }
                                                }



return false;

        public boolean isLess(Line l) {

                if(this.getLength()<l.getLength()) {



                }

                else {


                }
```

```
                                  }

return true; return  false;

        public boolean isEqual(Line l) {
            if(this.getLength()==l.getLength()) {
                return true;
            }
            else {

            }
                                  }

return false;

    }

    public class CompareLines {

        public static void main(String[] args) {
            Line l1=new Line(10,5,20,17);
            Line l2=new Line(8,15,9,23);
            if(l1.isEqual(l2)) {
                System.out.println("Both length are same");
            }
```

```
                if(l1.isGreater(l2)){
                        System.out.println("L1 length is grater than L2");
                }
                if(l1.isLess(l2)) {
                        System.out.println("L1 Length is less than L2");
                }
        }

}
```

OUTPUT:

```
L1 Length is less than L2
```

- **AIM: In the producer–consumer problem, the producer and the consumershare a common, fixed-size buffer used as a queue. (Take buffer size as 1). The producer's job is to generate data, put it into the buffer. At the same time, the consumer is consuming the data (i.e. removing it from the buffer).
The problem is to make sure that the producer won't try**

to add data into the buffer if it's full and that the consumer won't try to remove data from an empty buffer. Write a Java application consisting of allnecessary classes to achieve this.

```
public class process {
        int n;
        boolean value=false;
        synchronized int consume() {
                while(!value) {
                        try {
```

```
                    wait();
                }
                catch(InterruptedException e) {
                     System.out.println(e+" found");
                }
            }
            System.out.println("consumed "+ n);
            value=false;
            notify();
            return n;


        }
        synchronized int produce(int n) {
            while(value) {
                try {
                    wait();

                }
                catch(InterruptedException e){
                    System.out.println(e+" found");
                }


            }
            this.n=n;
            System.out.println("Produce "+n);
            value=true;
            notify(); return
            n;
        }
}



public class producer implements Runnable {process
        p;


        public producer(process p) {
                super();
                this.p = p;

        }


        public void run() {

                int i=1;
                while(true) {
                        int n1=p.produce(i++);


                }

        }

}



public class consumer implements Runnable {process
        p;
```

```
    public consumer(process p) {
        super();
        this.p = p;

    }


    public void run() {

        while(true) {

            p.consume();


        }

    }
}




public class testing {

    public static void main(String[] args) {
        process q=new process();
        producer p1=new producer(q);
        consumer c1=new consumer(q);
        Thread producerThread = new Thread(p1, "ProducerThread");
   Thread consumerThread = new Thread(c1, "ConsumerThread");


   producerThread.start();
   consumerThread.start();
```

```
                System.out.println("Terminated");

        }

}
```

OUTPUT:

```
Produce 1
consumed 1
Produce 2
consumed 2
Produce 3
consumed 3
Produce 4
consumed 4
Produce 5
consumed 5
Produce 6
consumed 6
Produce 7
consumed 7
```

- **AIM:Write a multithreaded Java application to produce a deadlock condition.**

```
class first {
    synchronized void fun(second b) {

        String name = Thread.currentThread().getName();
        System.out.println(name + " entered first.fun");
        try {

            Thread.sleep(1000);

        } catch (InterruptedException e) {

            System.out.println("first Interrupted");

        }

        System.out.println(name + " trying to call second.last()");
        b.last();
    }

    synchronized void last() {

        System.out.println("Inside first.last");

    }


}

class second {

    synchronized void fun1(first a) {

        String name = Thread.currentThread().getName();
        System.out.println(name + " entered second.fun1");
        try {
            Thread.sleep(1000);

        } catch (InterruptedException e) {

            System.out.println("second  Interrupted");

        }

        System.out.println(name + " trying to call first.last()");
        a.last();
    }

    synchronized void last() {
        System.out.println("Inside second.last");
    }
}


public class deadlock implements Runnable {

        first a=new first();
        second b=new second();

        deadlock() {
      Thread.currentThread().setName("MainThread");
      Thread t = new Thread(this, "RacingThread");
      t.start();
      a.fun(b);
      System.out.println("Back in main thread");
    }


                public void run() {
                    b.fun1(a);
                    System.out.println("Back in other thread");
                 }

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

```
}
```

OUTPUT:

```
MainThread entered first.fun
RacingThread entered second.fun1
MainThread trying to call second.last()
RacingThread trying to call first.last()
```

Java Technology

<u>Laboratory Work</u>

Subject: Java Technologies

Branch: CE

Semester: IV

Batch: ___A1_____

Student Roll No: CE005

Student Name: Chaudhari Krish

Department of Computer Engineering,

Faculty of Technology,

Dharmsinh Desai University, Nadiad – 387001.

Gujarat, INDIA.

Java Technology

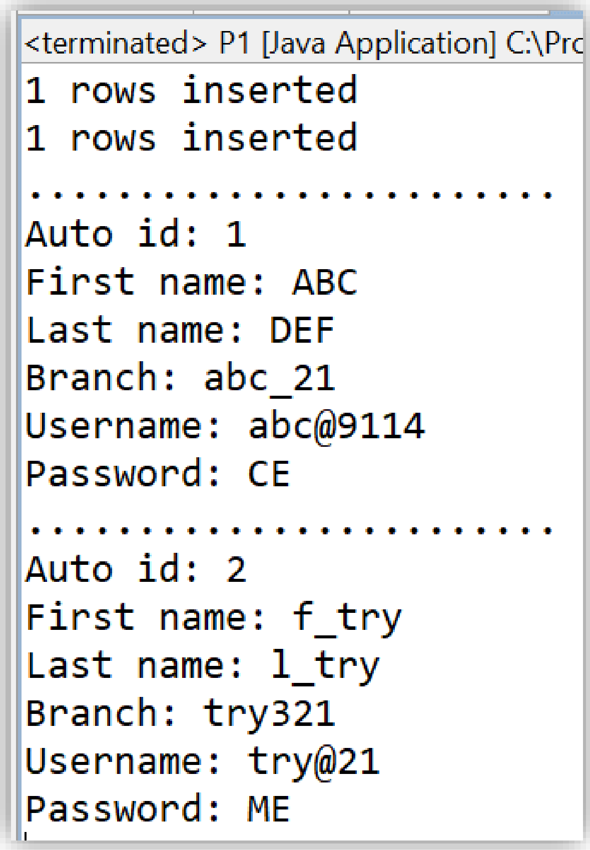
Lab – 6

1. Write a Java application to perform operations for student information like (id[Primary key, Auto increment],firstName, lastName, branch, username and password) from a database using JDBC.
    - Insert two records for student
    - Practice the use of the following methods of the ResultSet interface: absolute(), afterLast(), beforeFirst(), first(), isFirst(), isLast(), last(), previous(), next(), relative().

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
public class P1 {

        public static void main(String[] args) {
                try(Connection con = 
DriverManager.getConnection("jdbc:mysql://localhost:3306/student","root","root"))
                {
                        Statement s = con.createStatement();
                        String qry = "insert into 
`student`(`firstname`,`lastname`,`branch`,`username`,`password`) 
values('ABC','DEF','CE','abc_21','abc@9114')";
                        int i = s.executeUpdate(qry);
                        System.out.println(i + " rows inserted");
                        qry = "insert into 
`student`(`firstname`,`lastname`,`branch`,`username`,`password`) 
values('f_try','l_try','ME','try321','try@21')";
                        i = s.executeUpdate(qry);
                        System.out.println(i + " rows inserted");
                        qry = "select * from `student`";
                        ResultSet rs;
                        rs = s.executeQuery(qry);
                        while(rs.next())
                        {
                                System.out.println("........................");
                                System.out.println("Auto id: " + rs.getInt(1));
                                System.out.println("First name: " + rs.getString(2));
                                System.out.println("Last name: " + rs.getString(3));
                                System.out.println("Branch: " + rs.getString(4));
                                System.out.println("Username: " + rs.getString(5));
                                System.out.println("Password: " + rs.getString(6));
                        }
```

```
                    }
                    catch(Exception e)
                    {
                            System.out.println(e);
                    }
            }
    }
```

Output:-



```
<terminated> P1 [Java Application] C:\Pro
1 rows inserted
1 rows inserted
........................
Auto id: 1
First name: ABC
Last name: DEF
Branch: abc_21
Username: abc@9114
Password: CE
........................
Auto id: 2
First name: f_try
Last name: l_try
Branch: try321
Username: try@21
Password: ME
```

2. Using JDBC API and MySql database perform the following operations.
   I. create a table MOVIES with following columns in the database:
      Id of type INTEGER AUTO INCREMENT,
      Title of type VARCHAR (50),
      Genre of type VARCHAR (50),
      YearOfRelease of type INTEGER.
   II. Define Movie class with following data members
      private Integer id;
      private String title;
      private String genre;
      private Integer yearOfRelease;

   Create getters and setters for the mentioned data members.

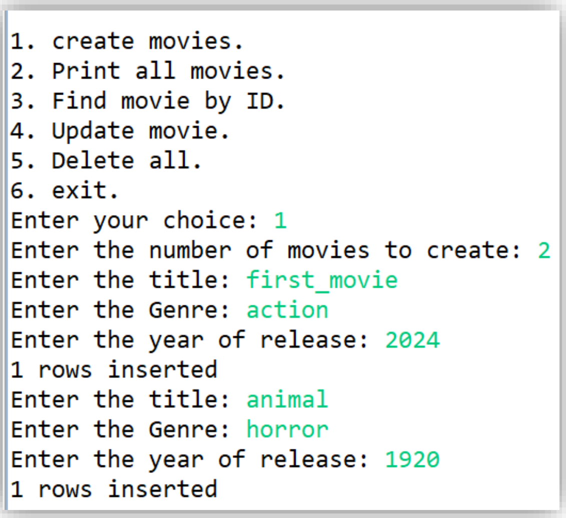
   III. Define following methods in a class, test the methods according to user input
      A. createMovie(Movie m)- it will insert a new record for a movie.
      B. deleteMovie(int MovieID)- it will delete a movie with given MovieID
      C. updateMovieTitle(String title, int id)- it will update the title of a movie with given id.
      D. findMovieById(int MovieId)- it will display all details of a movie with a given MovieId
      E. findAllMovie()- it will display all details of all movies.

```
import java.sql.*;
import java.util.Scanner;

class Movie
{
        private int id;
        private String title;
        private String genre;
        private int year;
        public int getId() {
                return id;
        }
        public void setId(int id) {
                this.id = id;
        }
        public String getTitle() {
                return title;
        }
        public void setTitle(String title) {
                this.title = title;
        }
        public String getGenre() {
                return genre;
        }
```

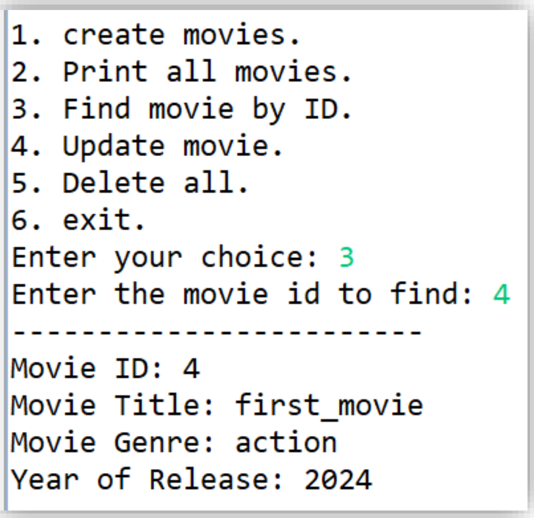
```
	public void setGenre(String genre) {
		this.genre = genre;
	}
	public int getYear() {
		return year;
	}
	public void setYear(int year) {
		this.year = year;
	}

	void createmovie()
	{
		setTitle(title);
		setGenre(genre);
		setYear(year);
		try(Connection con = 
DriverManager.getConnection("jdbc:mysql://localhost:3306/student","root","root"))
		{
			Statement s = con.createStatement();
			String insert = "insert into 
`movies`(`Title`,`Genre`,`YearOfRelease`) values('" + this.getTitle()
+"','"+this.getGenre()+"','" + this.getYear() +"')" ;
			int i = s.executeUpdate(insert);
			System.out.println(i + " rows inserted");
		}
		catch(Exception e)
		{
			System.out.println(e);
		}
	}

	void deletemovie(int movieID)
	{
		try(Connection con = 
DriverManager.getConnection("jdbc:mysql://localhost:3306/student","root","root"))
		{
			Statement s = con.createStatement();
			String qry = "delete from `movies` where `id` = "+movieID;
			int i = s.executeUpdate(qry);
			System.out.println(i + " rows deleted");
		}
		catch(Exception e)
		{
			System.out.println(e);
		}
	}
	void updatemovietitle(String title , int movieId)
	{
		try(Connection con = 
DriverManager.getConnection("jdbc:mysql://localhost:3306/student","root","root"))
		{
			Statement s = con.createStatement();
			String qry = "update `movies` set `Title` = '" + title + "' Where Id = 
" + movieId;
			int i = s.executeUpdate(qry);
			System.out.println(i + " rows updated");
		}
		catch(Exception e)
		{
			System.out.println(e);
		}
	}


	void findmoviebyid(int movieId)
	{
		try(Connection con = 
DriverManager.getConnection("jdbc:mysql://localhost:3306/student","root","root"))
		{
			Statement s = con.createStatement();
			String qry = "select * from `movies` where `Id` = " + movieId;
			ResultSet rs;
			rs = s.executeQuery(qry);
			while(rs.next())
			{
				System.out.println("------------------------");
				System.out.println("Movie ID: " + rs.getInt(1));
				System.out.println("Movie Title: " + rs.getString(2));
				System.out.println("Movie Genre: " + rs.getString(3));
				System.out.println("Year of Release: " + rs.getInt(4));
			}
		}
		catch(Exception e)
		{
			System.out.println(e);
		}
	}

	void  findallmovies()
	{
		try(Connection con = 
DriverManager.getConnection("jdbc:mysql://localhost:3306/student","root","root"))
		{
			Statement s = con.createStatement();
			String qry = "select * from `movies`";
			ResultSet rs;
			rs = s.executeQuery(qry);
			while(rs.next())
			{
				System.out.println("------------------------");
				System.out.println("Movie ID: " + rs.getInt(1));
				System.out.println("Movie Title: " + rs.getString(2));
				System.out.println("Movie Genre: " + rs.getString(3));
				System.out.println("Year of Release: " + rs.getInt(4));
			}
		}
		catch(Exception e)
		{
			System.out.println(e);
		}
	}

	void deleteall()
	{
		try(Connection con = 
DriverManager.getConnection("jdbc:mysql://localhost:3306/student","root","root"))
		{
			Statement s = con.createStatement();
			String qry = "delete from `movies`";
			int i = s.executeUpdate(qry);
			System.out.println(i + "rows deleted");
		}
		catch(Exception e)
		{
			System.out.println(e);
		}
	}
}
public class P2 {
	public static void main(String[] args) {
		Movie m1 = new Movie();
		Scanner sc = new Scanner(System.in);
		int ch = 0;
		while(ch!=6)
		{
```

```
                        System.out.println("\n\n1. create movies. \n2. Print all movies. \n3.
Find movie by ID. \n4. Update movie. \n5. Delete all. \n6. exit.");
                        System.out.print("Enter your choice: ");
                        ch = sc.nextInt();
                        switch(ch)
                        {
                                case 1:
                                        System.out.print("Enter the number of movies to
create: ");
                                        int n = sc.nextInt();
                                        for(int i=0; i<n; i++)
                                        {
                                                System.out.print("Enter the title: ");
                                                m1.setTitle(sc.next());
                                                System.out.print("Enter the Genre: ");
                                                m1.setGenre(sc.next());
                                                System.out.print("Enter the year of release:
");
                                                m1.setYear(sc.nextInt());
                                                m1.createmovie();
                                        }
                                        break;

                                case 2:
                                        m1.findallmovies();
                                        break;

                                case 3:
                                        System.out.print("Enter the movie id to find: ");
                                        int movieid = sc.nextInt();
                                        m1.findmoviebyid(movieid);
                                        break;

                                case 4:
                                        System.out.print("Enter movie id and new title: ");
                                        movieid = sc.nextInt();
                                        String title = sc.nextLine();
                                        m1.updatemovietitle(title, movieid);
                                        break;

                                case 5:
                                        m1.deleteall();
                                        break;

                                case 6:
                                        break;
                                default:
                                        System.out.println("Enter valid choice!");
                        }
                }
                sc.close();
        }
}
```

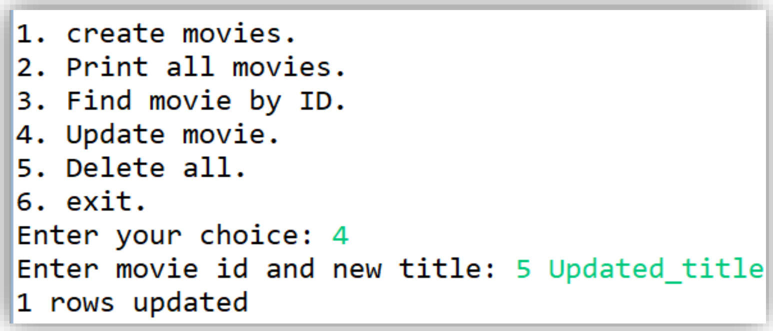
Output:-



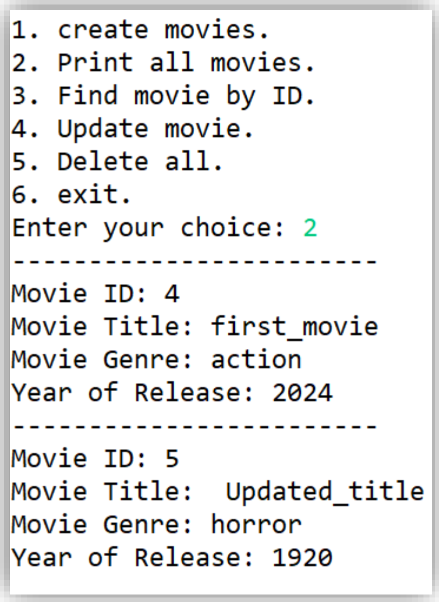
```
1. create movies.
2. Print all movies.
3. Find movie by ID.
4. Update movie.
5. Delete all.
6. exit.
Enter your choice: 1
Enter the number of movies to create: 2
Enter the title: first_movie
Enter the Genre: action
Enter the year of release: 2024
1 rows inserted
Enter the title: animal
Enter the Genre: horror
Enter the year of release: 1920
1 rows inserted
```



```
1. create movies.
2. Print all movies.
3. Find movie by ID.
4. Update movie.
5. Delete all.
6. exit.
Enter your choice: 3
Enter the movie id to find: 4
------------------------
Movie ID: 4
Movie Title: first_movie
Movie Genre: action
Year of Release: 2024
```



```
1. create movies.
2. Print all movies.
3. Find movie by ID.
4. Update movie.
5. Delete all.
6. exit.
Enter your choice: 4
Enter movie id and new title: 5 Updated_title
1 rows updated
```



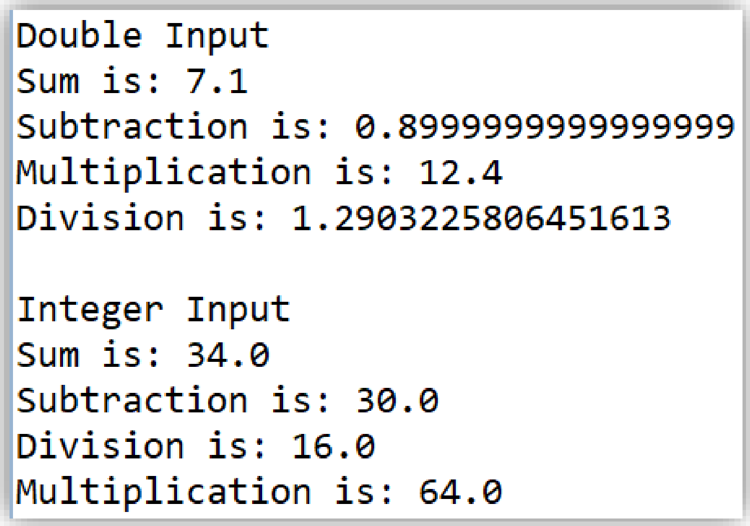
```
1. create movies.
2. Print all movies.
3. Find movie by ID.
4. Update movie.
5. Delete all.
6. exit.
Enter your choice: 2
------------------------
Movie ID: 4
Movie Title: first_movie
Movie Genre: action
Year of Release: 2024
------------------------
Movie ID: 5
Movie Title:  Updated_title
Movie Genre: horror
Year of Release: 1920
```

3. Create a Generic class Calculator which can perform addition, subtraction, multiplication and division. Make sure that Calculator class works for Numeric values only. Write an appropriate main method in TestCalculator class.

```
class Calculator<T extends Number>{
        T num1;
        T num2;

        Calculator(T num1, T num2)
        {
                this.num1 = num1;
                this.num2 = num2;
        }

        void addition()
        {
                System.out.println("Sum is: " + (this.num1.doubleValue() +
this.num2.doubleValue()));
        }

        void subtraction()
        {
                System.out.println("Subtraction is: " + (this.num1.doubleValue() -
this.num2.doubleValue()));
        }

        void multiplication()
        {
                System.out.println("Multiplication is: " + (this.num1.doubleValue() *
this.num2.doubleValue()));
        }

        void division()
        {
                System.out.println("Division is: " + (this.num1.doubleValue() /
this.num2.doubleValue()));
        }
}
public class P3 {

        public static void main(String[] args) {
                Calculator<Double> double_obj = new Calculator<>(4d,3.1d);
                System.out.println("Double Input");
                double_obj.addition();
                double_obj.subtraction();
```

```
                    double_obj.multiplication();
                    double_obj.division();

                    System.out.println("\nInteger Input");
                    Calculator<Integer> int_obj = new Calculator<>(32,2);
                    int_obj.addition();
                    int_obj.subtraction();
                    int_obj.division();
                    int_obj.multiplication();
          }
}
```

Output:-

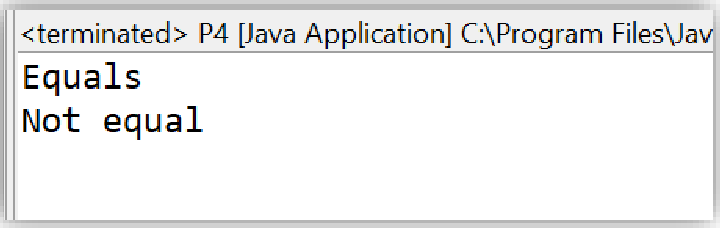


```
Double Input
Sum is: 7.1
Subtraction is: 0.8999999999999999
Multiplication is: 12.4
Division is: 1.2903225806451613

Integer Input
Sum is: 34.0
Subtraction is: 30.0
Division is: 16.0
Multiplication is: 64.0
```

4. Write a Java program to create a generic method that takes two arrays of T type and checks if they have the same elements in the same order.

```
import static java.lang.System.exit;
public class P4 {
        <T>void compare(T a1[], T a2[])
        {
                int i=0;
                if(a1.length == a2.length){
                        for(T ele : a1)
                        {
                                if(!ele.toString().equals(a2[i].toString()))
                                {
                                        System.out.println("Not equal");
                                        exit(i);
                                }
                                i++;
                        }
                        System.out.println("Equals");
                }
                else{System.out.println("Not equal");}
        }

        public static void main(String[] args)
        {
                P4 obj= new P4();
                Double[] num1 = {1.0d,2.0d,3.0d,4.0d};
                Float[] num2 = {1.0f,2.0f,4.0f,3.0f};
                obj.compare(num1, num2);
                String []str1 = {"1", "2", "3", "4", "5"};
                String []str2 = {"1", "2", "3", "4", "5"};
                obj.compare(str1, str2);
        }
}
```

Output:-



```
<terminated> P4 [Java Application] C:\Program Files\Jav
Equals
Not equal
```

Java Technology

<u>Laboratory Work</u>

Subject: Java Technologies

Branch: B.Tech. (CE)

Semester: IV

Batch: ___A1_____

Student Roll No: CE-005

Student Name: krish chaudhari

Department of Computer Engineering,

Faculty of Technology,

Dharmsinh Desai University, Nadiad – 387001.

Gujarat, INDIA.

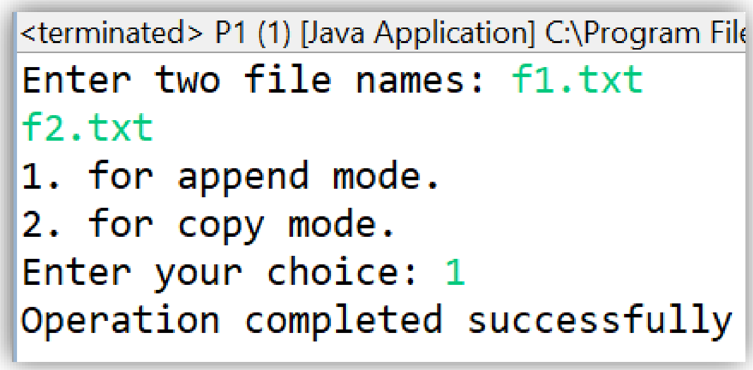
Java Technology

Lab – 7

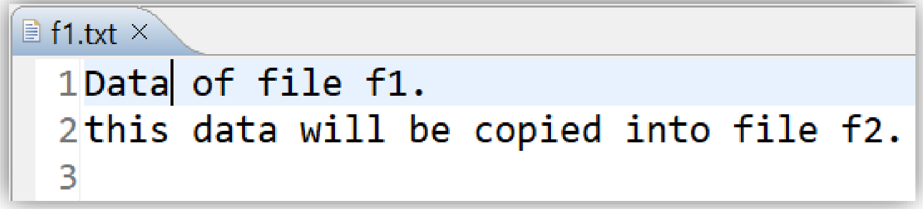
1. Write a Java program that accepts two filenames. Based on the user's choice to copy or append. Copy the first file into the second file or append the content of first file to the second file.

```
package lab_7;
import java.io.*;
import java.util.Scanner;

public class P1 {
    public static void main(String[] args) throws IOException {
            Scanner sc = new Scanner(System.in);
            System.out.print("Enter two file names: ");
            String file1 = sc.nextLine();
            String file2 = sc.nextLine();

            FileInputStream in = null;
            FileOutputStream out = null;

            System.out.println("1. for append mode. \n2. for copy mode.");
            System.out.print("Enter your choice: ");
            int ch = sc.nextInt();
            switch(ch) {
                    case 1:
                            try {
                                    in = new FileInputStream(file1);
                                    out = new FileOutputStream(file2, true);
                                    int c;
                                    while((c=in.read()) != -1)
                                    {
                                            out.write(c);
                                    }

                                    System.out.println("Operation completed successfully");
                            }
                            catch(Exception e)
                            {
                                    System.out.println(e);
                            }
                            finally
                            {
                                    if(in != null) { in.close();}
                                    if(out != null) { out.close();}
                            }
                    break;

                case 2:
                    try {
                        in = new FileInputStream(file1);
                        out = new FileOutputStream(file2);
                        int c;
                        while((c=in.read()) != -1)
                        {
                            out.write(c);
                        }
                        System.out.println("Operation completed successfully");
                    }
                    catch(Exception e)
                    {
                        System.out.println(e);
                    }
                    finally
                    {
                        if(in != null) { in.close();}
                        if(out != null) { out.close();}
                    }

                    break;

                default:
                    System.out.println("Enter valid choice!");
            }
            sc.close();
        }
    }
```

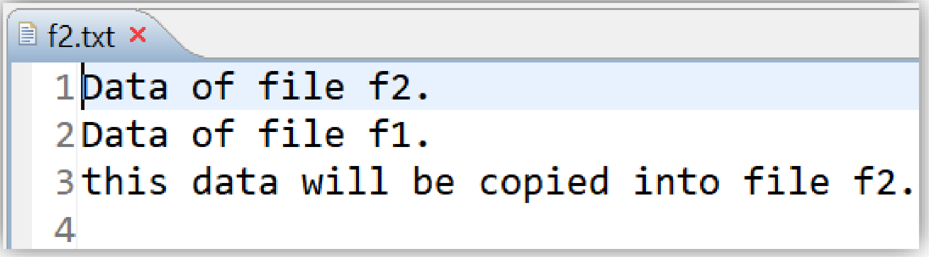
Output:-



```
<terminated> P1 (1) [Java Application] C:\Program File
Enter two file names: f1.txt
f2.txt
1. for append mode.
2. for copy mode.
Enter your choice: 1
Operation completed successfully
```



```
f1.txt
Data of file f1.
this data will be copied into file f2.
```

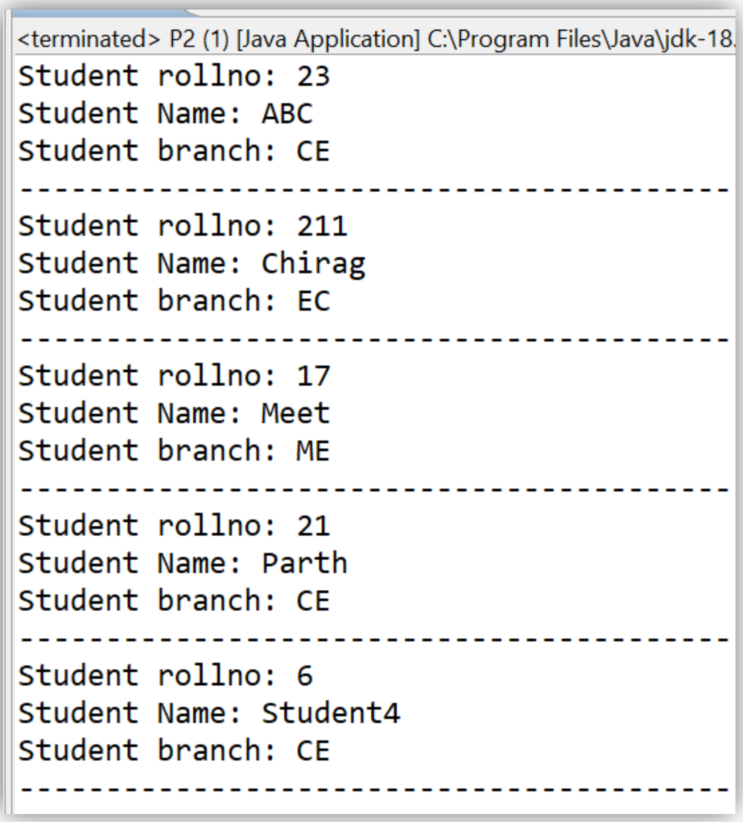


```
f2.txt
Data of file f2.
Data of file f1.
this data will be copied into file f2.
```

2. Write a java program to generate a linked list of some five students (Student objects) and display the list of students in a sorted order as per name of students.

```
package lab_7;
import java.util.*;

class Student implements Comparable<Student>{
    private String name;
    private int rollno;
    private String branch;

    Student(String name,int rollno,String branch){
            this.name = name;
            this.rollno = rollno;
            this.branch = branch;
    }

    public String getName() {
            return name;
    }

    public void setName(String name) {
            this.name = name;
    }

    public int getRollno() {
            return rollno;
    }

    public void setRollno(int rollno) {
            this.rollno = rollno;
    }

    public String getBranch() {
            return branch;
    }

    public void setBranch(String branch) {
            this.branch = branch;
    }

    @Override
    public int compareTo(Student s1) {
            return this.name.compareTo(s1.getName());
    }
}
public class P2 {

    public static void main(String[] args) {
            List<Student> my_list = new LinkedList<Student>();
            my_list.add(new Student("Parth",21,"CE"));
            my_list.add(new Student("Chirag",211,"EC"));
            my_list.add(new Student("Meet",17,"ME"));
            my_list.add(new Student("Student4",006,"CE"));
            my_list.add(new Student("ABC",23,"CE"));
            Collections.sort(my_list);
            for (Student student : my_list) {
                    System.out.println("Student rollno: " + student.getRollno());
                    System.out.println("Student Name: " + student.getName());
                    System.out.println("Student branch: " + student.getBranch());
                    System.out.println("-----------------------------------------");
        }
    }
}
```

Output:-



```
<terminated> P2 (1) [Java Application] C:\Program Files\Java\jdk-18.
Student rollno: 23
Student Name: ABC
Student branch: CE
-----------------------------------------
Student rollno: 211
Student Name: Chirag
Student branch: EC
-----------------------------------------
Student rollno: 17
Student Name: Meet
Student branch: ME
-----------------------------------------
Student rollno: 21
Student Name: Parth
Student branch: CE
-----------------------------------------
Student rollno: 6
Student Name: Student4
Student branch: CE
-----------------------------------------
```

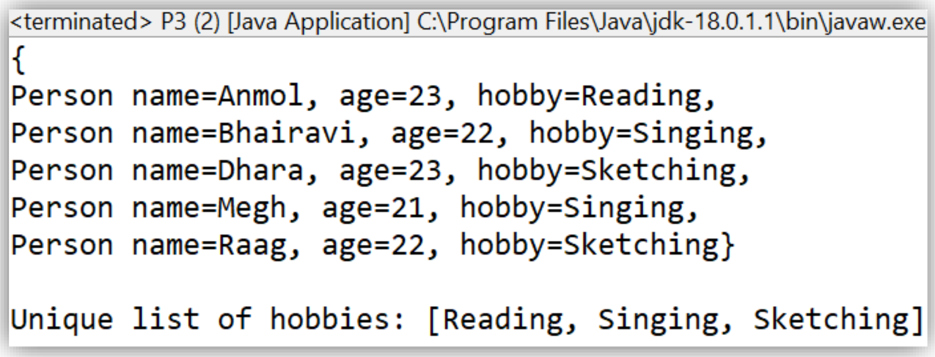
3. Write a Java program to map Person objects to string hobby using TreeMap class. This mapping should store Person objects in ascending sorting by their name.

Persons	hobby
Name: Bhairavi Age: 22	Singing
Name: Dhara Age:23	Sketching
Name: Anmol Age: 23	Reading
Name: Megh Age 21	Singing
Name: Raag Age:22	Sketching

Find the unique list/set of all the hobbies that are mapped in this collection and display it.

```
package lab_7;
import java.util.*;
class Person implements Comparable<Person>{
    private String name;
    private int age;

    Person(String name, int age)
    {
            this.name = name;
            this.age = age;
    }
    public String getName() {
            return name;
    }
    public void setName(String name) {
            this.name = name;
    }
    public int getAge() {
            return age;
    }
    public void setAge(int age) {
            this.age = age;
    }

    @Override
    public String toString() {
            return "\nPerson name=" + this.name + ", age=" + this.age + ", hobby";
    }

    @Override
    public int compareTo(Person p1) {
            return this.getName().compareTo(p1.getName());
    }
}
public class P3 {
    public static void main(String[] args) {
            int i=0;
            Map<Person, String> my_map = new TreeMap<>();
            String hobby[] = {"Singing", "Sketching", "Reading", "Singing", "Sketching"};

            Person obj[] = new Person[5];

            obj[0] = new Person("Bhairavi",22);
            obj[1] = new Person("Dhara",23);
            obj[2] = new Person("Anmol",23);
            obj[3] = new Person("Megh",21);
            obj[4] = new Person("Raag",22);

            for(Person p1: obj)
            {
                    my_map.put(p1, hobby[i]);
                    i++;
            }

            System.out.println(my_map);
            HashSet<String> my_list = new HashSet<>();
            my_list.addAll(Arrays.asList(hobby));

            System.out.println("\nUnique list of hobbies: "+my_list);
    }
}
```

Output:-



```
<terminated> P3 (2) [Java Application] C:\Program Files\Java\jdk-18.0.1.1\bin\javaw.exe
{
Person name=Anmol, age=23, hobby=Reading,
Person name=Bhairavi, age=22, hobby=Singing,
Person name=Dhara, age=23, hobby=Sketching,
Person name=Megh, age=21, hobby=Singing,
Person name=Raag, age=22, hobby=Sketching}

Unique list of hobbies: [Reading, Singing, Sketching]
```

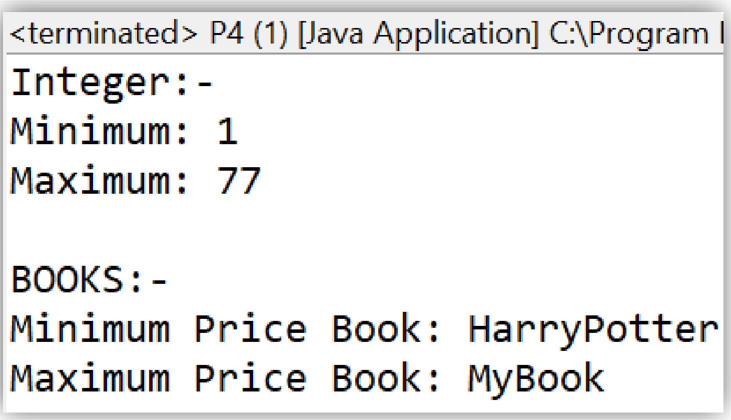
4. Write a generic interface MinMax having two methods findMin() and findMax().Create a class named MyClass which implements the above interface. Write an appropriate demo class to test your classes and interfaces. Create an object of MyClass which stores an array of Book and find books having minimum and maximum price.

```
package lab_7;

interface MinMax<T>{
    T findmin();
    T findmax();
}

class MyClass<T extends Comparable<T>> implements MinMax<T>{

    T arr[];

    MyClass(T[] arr){
            this.arr = arr;
    }

    @Override
    public T findmin() {
            T min = arr[0];
            for(int i=1 ; i<arr.length ; i++)
            {
                    if(arr[i].compareTo(min) < 0)
                    {
                            min = arr[i];
                    }
            }
            return min;
    }

    @Override
    public T findmax() {
            T max = arr[0];
            for(int i=1 ; i<arr.length ; i++)
            {
                    if(arr[i].compareTo(max) > 0)
                    {
                            max = arr[i];
                    }
            }
            return max;
    }
}
```

```
class Book implements Comparable<Book>
{
    String book;
    int price;

    Book(String book, int price)
    {
            this.book = book;
            this.price = price;
    }

    @Override
    public int compareTo(Book obj) {
            if(this.price < obj.price)
            {
                    return -1;
            }
            else if(this.price == obj.price)
            {
                    return 0;
            }
            else {
                    return 1;
            }
    }
    @Override
    public String toString() {

            return this.book;
    }
}

class P4{
    public static void main(String[] args) {
            Integer arr[] = {1,23,56,77,3,12};
            MyClass<Integer> int_obj = new MyClass<Integer>(arr);

            System.out.println("Integer:- \nMinimum: "+ int_obj.findmin());
            System.out.println("Maximum: "+ int_obj.findmax());


            Book books[] = {new Book("HarryPotter",340),
                                            new Book("LearnJAVA",500),
                                            new Book("GameDevelopment", 400),
                                            new Book("MyBook", 800)};
            MyClass<Book> books_obj = new MyClass<Book>(books);
```

```
		System.out.println("\nBOOKS:- \nMinimum Price Book: " + books_obj.findmin());
		System.out.println("Maximum Price Book: " + books_obj.findmax());
	}
}
```

Output:-



```
<terminated> P4 (1) [Java Application] C:\Program
Integer:-
Minimum: 1
Maximum: 77

BOOKS:-
Minimum Price Book: HarryPotter
Maximum Price Book: MyBook
```

Java Technology

<u>Laboratory Work</u>

Subject: Java Technologies

Branch: B.Tech. (CE)

Semester: IV

Batch: ___<u>A1</u>_____

Student Roll No: <u>CE-005</u>

Student Name: <u>krish chaudhari</u>

Department of Computer Engineering,

Faculty of Technology,

Dharmsinh Desai University, Nadiad – 387001.

Gujarat, INDIA.

Java Technology

Lab – 8

1. **Create a Simple JAVA web application to display the welcome message using JSP or servlet.**

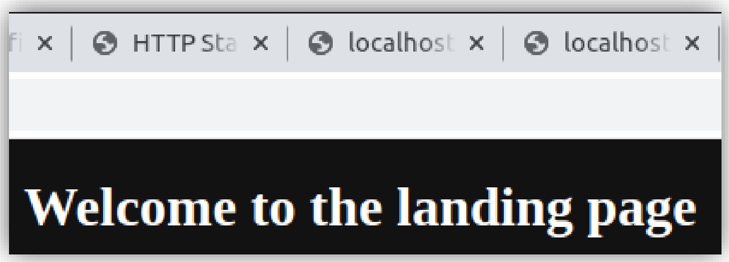
Index.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
      <head>
                  <meta charset="UTF-8">
                  <title>Welcome page</title>
      </head>
      <body>
                  <center><h1>Welcome to the landing page</h1></center>
      </body>
</html>
```

Web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="https://jakarta.ee/xml/ns/jakartaee" xmlns:web="http://xmlns.jcp.org/xml/ns/javaee"
xsi:schemaLocation="https://jakarta.ee/xml/ns/jakartaee
https://jakarta.ee/xml/ns/jakartaee/web-app_5_0.xsd" id="WebApp_ID" version="5.0">
  <display-name>lab_8</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
    <welcome-file>index.jsp</welcome-file>
    <welcome-file>index.htm</welcome-file>
    <welcome-file>default.html</welcome-file>
    <welcome-file>default.jsp</welcome-file>
    <welcome-file>default.htm</welcome-file>
  </welcome-file-list>
  </web-app>
```

Output:-



Welcome to the landing page

2. **Write a Java web application for a login module which contains the following components:**
   - **index.html: for getting input from the user.**
   - **ValidateServlet.java: a servlet class for validating the user. If it is a valid user (validate from a database using PreparedStatement), it will forward the request to the WelcomeServlet. If the user is not validated then it displays an Error message along with the response from index.html.**
   - **Welcome.jsp: a JSP file for displaying the welcome message.**

Index.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>login</title>
</head>
<body>
<div style="border: 1px solid white;">
    <form action="Myservlet" method="post">
            Username: <input type="text" placeholder="Enter Username" name="username"
/><br>
            Password: <input type="password" placeholder="Enter password" name="pass"
/><br>
            <input type="submit" value="Login" name="submit" />
    </form>
</div>
</body>
</html>
```

Myservlet.java

```
import jakarta.servlet.RequestDispatcher;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.sql.*;
import java.io.PrintWriter;

public class Myservlet extends HttpServlet {
   private static final long serialVersionUID = 1L;
   public Myservlet() {
      super();
```

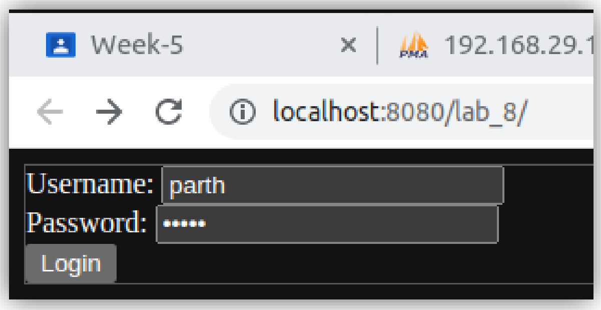
```
        }
         protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
                response.getWriter().append("Served at: ").append(request.getContextPath());
                response.setContentType("text/html");

                String username = request.getParameter("username");
                String pass = request.getParameter("pass");

                try
                {
                        // DriverManager.registerDriver(new com.mysql.jdbc.Driver ());
                        Class.forName("com.mysql.jdbc.Driver");
                        Connection con =
DriverManager.getConnection("jdbc:mysql://192.168.29.150/ce4_21","ce4_21","ce4_21");

                        PreparedStatement qry = con.prepareStatement("select * from `users`
where username = ? and pass = ?");
                        qry.setString(1,username);
                        qry.setString(2, pass);
                        ResultSet rs = qry.executeQuery();

                        if(rs.next())
                        {
                                request.getRequestDispatcher("home.jsp").forward(request,
response);

                        }
                        else
                        {
                                response.getWriter().append("\nEnter valid username &
password");
                        }
                }
                catch(Exception e)
                {
                        System.out.println(e);
                }
        }
        protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
                doGet(request, response);
        }
```

```
}
```

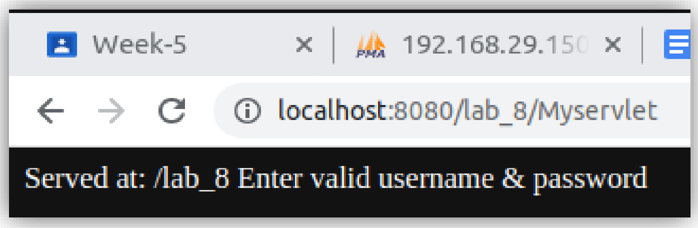
Home.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
      <h1>Welcome to the homepage,
      <% String name = request.getParameter("username"); %>
      <% out.print(name); %></h1>
</body>
</html>
```

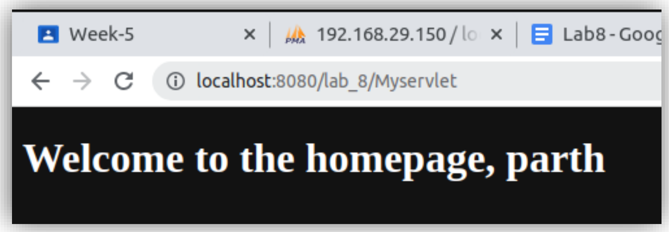
Output:-



Username: parth
Password: •••••



Served at: /lab_8 Enter valid username & password



Welcome to the homepage, parth

Java Technology

<u>Laboratory Work</u>

Subject: Java Technologies

Branch: B.Tech. (CE)

Semester: IV

Batch: ___A1_____

Student Roll No: CE-005

Student Name: Choudhary Krish L.

Department of Computer Engineering,

Faculty of Technology,

Dharmsinh Desai University, Nadiad – 387001.

Gujarat, INDIA.

Lab – 9

1. **Write a Java web application using HttpSession which allows only logged in users to access the other JSPs/Servlets of the application. Write the following components:**
    1. **Login.html allows users to provide username and password and send them as request parameters toLoginVerifierServlet.**
    2. **LoginVerifierServlet takes username and password from login.html and verifies it. If credentials are correct then it creates a session. It displays welcome message along with username and links to first.jsp and second.jsp.**
    3. **first.jsp and second.jsp display some text with username and can be accessed if the user is logged in. (you should delegate to Login.html if the user is not logged in)**

Login.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <form action="LoginVerifierServlet" method="post">
            Username: <input type="text" name="username" placeholder="Enter Username"
/><br>
            Password: <input type="password" name="pass" placeholder="Enter Password"
/><br>
            <input type="submit" value="Login" name="submit" />
    </form>
</body>
</html>
```

LoginVerifierServlet.java

```
import jakarta.servlet.RequestDispatcher;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import jakarta.servlet.http.HttpSession;

import java.io.IOException;

/**
 * Servlet implementation class LoginVerifierServlet
 */
public class LoginVerifierServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

   /**
    * @see HttpServlet#HttpServlet()
    */
   public LoginVerifierServlet() {
       super();
       // TODO Auto-generated constructor stub
   }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
                response.setContentType("text/html");
                if(request.getParameter("username").equals("admin") &&
request.getParameter("pass").equals("admin"))
                {
                        HttpSession session = request.getSession();
                        session.setAttribute("username",request.getParameter("username"));
                        response.getWriter().append("\n<h1>Welcome,
"+request.getParameter("username")+"</h1>");
                        response.getWriter().append("<a href='first.jsp'><h1>first.jsp</h1></a>");
                        response.getWriter().append("<a
href='second.jsp'><h1>Second.jsp</h1></a>");
                }
                else
                {
                        response.getWriter().append("Invalid username or password!");
                        RequestDispatcher dis = request.getRequestDispatcher("login.html");
                        dis.include(request, response);
                }
    }

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
                // TODO Auto-generated method stub
                doGet(request, response);
    }
}
```

first.jsp

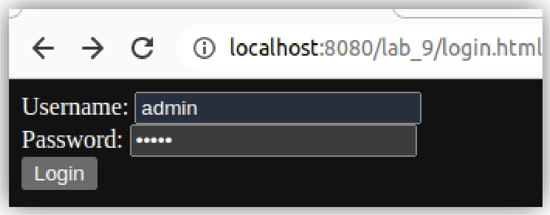
```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8" session="false"%>

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>first</title>
</head>
<body>
<%
    HttpSession ses = request.getSession(false);
    if(ses.getAttribute("username") != null)
    {
            out.print("<h3>Welcome, "+ ses.getAttribute("username") +"<br>this is the first
jsp page</h3>");
    }
    else
    {
            response.sendRedirect("login.html");
    }

%>
</body>
</html>
```

second.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8" session="false"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>first</title>
</head>
<body>
<%
    HttpSession ses = request.getSession(false);
    if(ses.getAttribute("username") != null)
    {
            out.print("<h3>Welcome, "+ ses.getAttribute("username") +"<br>this is the
second jsp page</h3>");
    }
```

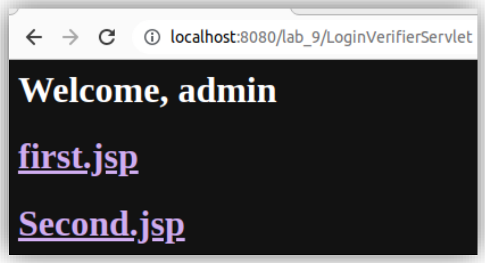
```
        else
        {
                response.sendRedirect("login.html");
        }

    %>
    </body>
    </html>
```

Output:-



localhost:8080/lab_9/login.html

Username: admin
Password: •••••
Login

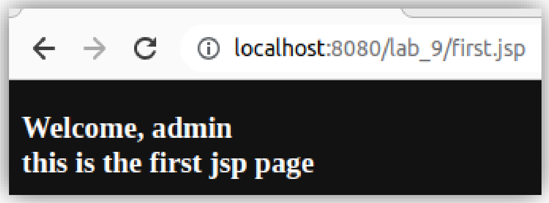


localhost:8080/lab_9/LoginVerifierServlet

Welcome, admin

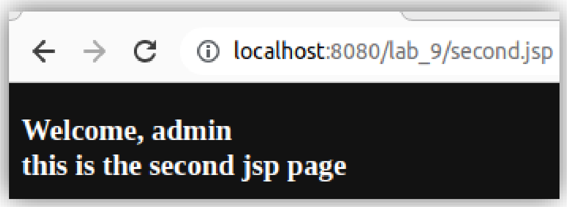
first.jsp

Second.jsp



localhost:8080/lab_9/first.jsp

Welcome, admin
this is the first jsp page



localhost:8080/lab_9/second.jsp

Welcome, admin
this is the second jsp page

2. **Write a web based java application containing a JSP which performs the simple arithmetic calculation. Take the necessary operands and operators in textboxes. Write your JSP code using jsp:useBean action tag.**

calc.jsp

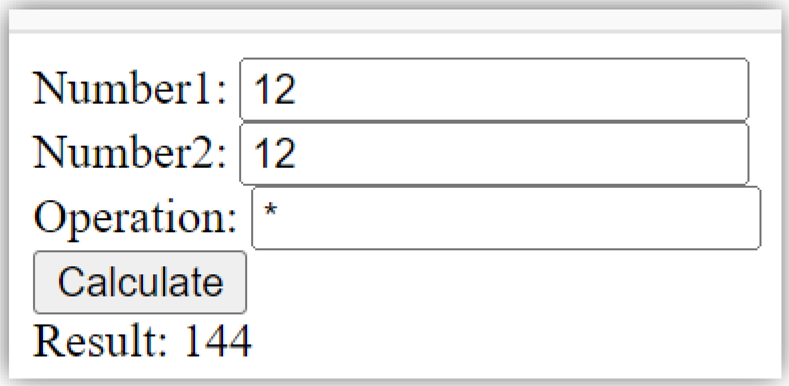
```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
     <form action="#" method="post">
     Number1: <input type="number" name="num1" placeholder="Enter first number"
/><br>
     Number2: <input type="number" name="num2" placeholder="Enter second number"
/><br>
     Operation: <input type="text" name="ope" placeholder="Enter the operator" /><br>
     <input type="submit" value="Calculate" />
     </form>
     <jsp:useBean class="lab9.Calculator" id="calc" scope="request" ></jsp:useBean>
     <jsp:setProperty name="calc" property="num1" param="num1" />
     <jsp:setProperty name="calc" property="num2" param="num2" />
     <jsp:setProperty  name="calc" property="ope" param="ope" />

     Result: <%= calc.calculate() %>
</body>
</html>
```

Calculator.java

```
public class Calculator{
     private int num1;
     private int num2;
     private String ope;

     public Calculator()
     {
             this.num1 = 0;
             this.num2 = 0;
             this.ope = "+";
     }
```

```
        public int getNum1() {
                return num1;
        }
        public void setNum1(int num1) {
                this.num1 = num1;
        }
        public int getNum2() {
                return num2;
        }
        public void setNum2(int num2) {
                this.num2 = num2;
        }
        public String getOpe() {
                return ope;
        }
        public void setOpe(String ope) {
                this.ope = ope;
        }

        public int calculate()
        {
                String op = this.getOpe();
                int ans=0;
                switch(op)
                {
                        case "+":
                                ans = this.getNum1() + this.getNum2();
                                break;
                        case "*":
                                ans = this.getNum1() * this.getNum2();
                                break;
                        case "/":
                                ans = this.getNum1() / this.getNum2();
                                break;
                        case "-":
                                ans = this.getNum1() - this.getNum2();
                                break;
                }

                return ans;
        }
    }
```

Output:-



Number1: 12
Number2: 12
Operation: *

Result: 144