Laboratory Work

Subject: Java Technologies

Branch: B.Tech. (CE)

Semester: IV

Batch: __**A4**____

Student Roll No: _____**CE079**____

Student Name: _____**DOBARIYA VRUND GHANSHYAMBHAI**___

Department of Computer Engineering,

Faculty of Technology,

Dharmsinh Desai University, Nadiad 387001. –

Gujarat, INDIA.

Laboratory Work

Subject: Java Technologies

Branch: B.Tech. (CE)

Semester: IV

Batch: A4

Student Roll No: **CE079**

Student Name: **DOBARIYA VRUND GHANSHYAMBHAI**

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”.

```
class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
```

```
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.

```
import java.util.*;
class Second {
  static void fun(int n) {
          for(int i=1;i<=n;i++) {
                  if(i%3==0 || i%5==0)
                          System.out.println(i);
          }
  }

  public static void main(String[] args) {
          Scanner s = new Scanner(System.in);
          int n = s.nextInt();
          Second.fun(n);
  }
}
```

```
C:\Users\Vrund\Downloads\lab1-20231216T031712Z-001\lab1>java Second
19
3
5
6
9
10
12
15
18
```

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 “Era”, the program should
Respond “Hello Era!”.

```
import java.util.*;
class Third {
  public static void main(String[] args) {
          Scanner s = new Scanner(System.in);
          String str = s.next();
          System.out.println("Hello " + str + "!");
  }
}
```

```
C:\Users\Vrund\Downloads\lab1-20231216T031712Z-001\lab1>java Third
Vrund
Hello Vrund!
```

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.

```
import java.util.*;
class Fourth {
  static int sum;
  public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        String str = s.nextLine();
        int rollNumber = s.nextInt();
        int[] arr = new int [5];
        for(int i=0;i<5;i++) {
              arr[i] = s.nextInt();
              sum += arr[i];
        }
        double avg = sum/5d;
        char grade;
        if(avg > 85)
              grade='A';
        if(avg > 75)
              grade='B';
        else
              grade='C';
        System.out.println(str + "\n" + rollNumber + "\n" + avg + "\n"
+ grade);

  }
}
```

```
C:\Users\Vrund\Downloads\lab1-20231216T031712Z-001\lab1>java Fourth
vrund dobariya
79
90 89 88 98 91
vrund dobariya
79
91.2
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:
{68,79,86,99,23,2,41,100} 256
{1,2,3,4,5,6,7,8,9,10} 30

```
import java.util.*;
class Solution {
  int calculateSumOfEvenNumbers(int[] arr) {
        int s = 0;
        for(int i=0;i<arr.length;i++) {
              if(arr[i]%2 == 0)
                    s+=arr[i];
        }
        return s;
  }
}

class Tester {
  public static void main(String[] args) {
        Solution sol = new Solution();
        Scanner s = new Scanner(System.in);
        int n = s.nextInt();
        int arr[] = new int[n];
        System.out.println(sol.calculateSumOfEvenNumbers(arr));
  }
}
```

```
C:\Users\Vrund\Downloads\lab1-20231216T031712Z-001\lab1>java Tester
4
1 2 3 4
6
```

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.

```
import java.util.*;
class Sixth {
  static int sum;
  public static void main(String[] args) {
        Scanner s = new Scanner(System.in);

        int n1 = s.nextInt();
        int n2 = s.nextInt();
        int m1 = s.nextInt();
        int m2 = s.nextInt();
        int[][] a1 = new int [n1][n2];
        int[][] a2 = new int [m1][m2];
        int [][] mul = new int [n2][m1];
        for(int i=0;i<n1;i++) {
                for(int j=0;j<n2;j++) {
                        a1[i][j]=s.nextInt();
                }
        }

        for(int i=0;i<m1;i++) {
                for(int j=0;j<m2;j++) {
                        a2[i][j]=s.nextInt();
                }
        }

        if(n2!=m1) {
                System.out.println("multiplication not possible!");
                return;
        }
        for(int i=0;i<n1;i++) {
                for(int j=0;j<m2;j++) {
                        mul[i][j]=0;
                        for(int k=0;k<m1;k++) {
                                mul[i][j] += a1[i][k]*a2[k][j];
                        }
                }
        }
        //multiplication
        for(int i=0;i<n1;i++) {
                for(int j=0;j<m2;j++) {
                        System.out.print(mul[i][j]);
                        System.out.println(" ");
                }
                System.out.println("");
        }
        //addition
        for(int i=0;i<n1;i++) {
                for(int j=0;j<m2;j++) {
                        System.out.print(a1[i][j]+a2[i][j]);
                        System.out.println(" ");
                }
                System.out.println("");
        }
  }
}
```

```
C:\Users\Vrund\Downloads\lab1-20231216T031712Z-001\lab1>java Sixth
1 2
2 1
11 22
33
45
1353
```

Laboratory Work

Subject: Java Technologies

Branch: B.Tech. (CE)

Semester: IV

Batch: A4

Student Roll No: **CE079**

Student Name: **DOBARIYA VRUND GHANSHYAMBHAI**

Department of Computer Engineering,

Faculty of Technology,

Dharmsinh Desai University, Nadiad – 387001.

Gujarat, INDIA.

LAB 2

Topics: String, StringBuffer, StringBuilder, array of objects, this keyword, constructor overloading

1. Write a program that returns the number of times that the string “hi” appears anywhere in the given string.

```
import java.util.*;
class Driver {
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        String str = s.nextLine();
        int c=0;
        for(int i=0;i<str.length()-1;i++) {
            if(str.startsWith("hi", i))
                c++;
        }
        System.out.println(c);
    }
}
```

```
C:\Users\Vrund\Downloads\JT_LAB_2>java Driver
hiHellohiHelloHowAreYou
2
```

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.util.*;
class Driver {
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        String str = s.nextLine();
        str = str.toUpperCase();
        boolean f=true;
        for(int i=0;i<str.length();i++) {
            if(str.charAt(i) != str.charAt(str.length()-i-1)) {
                f = false;
                break;
            }
        }
        System.out.println(f);
    }
}
```

```
C:\Users\Vrund\Downloads\JT_LAB_2>java Driver
Refer
true
```

3. 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.util.*;
class Driver {
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
```

```
        String str = s.nextLine();
        String[] arr = str.split(" ",3);
        System.out.println(arr[0].charAt(0)+". "+arr[1].charAt(0)+".
"+arr[2]);
    }
}
```

```
C:\Users\Vrund\Downloads\JT_LAB_2>java Driver
Vrund Ghanshyambhai Dobariya
V. G. Dobariya
```

4. 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.util.*;
class Tester {
    static String removeWhiteSpaces(String str) {
        return str.replace(" ","");
    }
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        String str = s.nextLine();
        System.out.println(removeWhiteSpaces(str));
    }
}
```

```
C:\Users\Vrund\Downloads\JT_LAB_2>java Tester
Hey! How are you?
Hey!Howareyou?
```

5. 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.*;
class Student {
    int rollNo;
    String name;
    int[] marks;
    Student(int rollNo, String name, int[] marks) {
        this.rollNo = rollNo;
        this.name = name;
        this.marks = marks;
    }
    Student(int rollNo) {
        this.name = "N/A";
        this.rollNo = rollNo;
        this.marks = new int[5];
    }
    Student() {
        this.name = "N/A";
        this.rollNo = -1;
        this.marks = new int[5];
    }
```

```
    float findAverage() {
        int sum=0;
        for(int i=0;i< marks.length;i++) {
            sum+= marks[i];
        }
        return sum/5f;
    }
    void display() {
        System.out.print("Roll No.: ");
        System.out.println(rollNo);
        System.out.println("Name: ");
        System.out.println(name);
        System.out.println("Marks are:");
        for(int i=0;i<5;i++) {
            System.out.print(marks[i]);
            System.out.print(" ");
        }
        System.out.println("");
    }
}
class TestStudent {
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        int marks[] = {34,34,92,56,37};
        Student s1 = new Student(19,"Jack sparrow",marks);

        // Array of objects
        Student[] arr = new Student[5];

        for(int i=0;i<3;i++) {
            System.out.print("Enter the details of student:\n");
            int r = s.nextInt();
            String n = s.next();
            int[] m = new int[5];
            for(int j=0;j<5;j++) {
                m[j] = s.nextInt();
            }
            arr[i] = new Student(r,n,m);
        }
        float max = 0f;
        Student stuWithMaxMarks = new Student();
        for(int i=0;i<3;i++) {
            arr[i].display();
            if(arr[i].findAverage() > max) {
                max = arr[i].findAverage();
                stuWithMaxMarks = arr[i];
            }
        }
        System.out.println("Student with max marks: ");
        stuWithMaxMarks.display();
    }
}
```

```
C:\Users\Vrund\Downloads\JT_LAB_2>java TestStudent
Enter the details of student:
11
jack
89 92 90 87 76
Enter the details of student:
19
rob
34 56 89 78 90
Enter the details of student:
13
leo
89 77 58 56 49
Roll No.: 11
Name:
jack
Marks are:
89 92 90 87 76
Roll No.: 19
Name:
rob
Marks are:
34 56 89 78 90
Roll No.: 13
Name:
leo
Marks are:
89 77 58 56 49
Student with max marks:
Roll No.: 11
Name:
jack
Marks are:
89 92 90 87 76
```

Laboratory Work

Subject: Java Technologies Branch:

B.Tech. (CE) Semester: IV

Batch **A4**

Student Roll No: **CE079**

Student Name: **DOBARIYA VRUND GHANSHYAMBHAI**

Department of Computer Engineering,

Faculty of Technology,

Dharmsinh Desai University, Nadiad – 387001.

Gujarat, INDIA.

LAB 3

Topics: Inheritance, Polymorphism(method overriding), static keyword

1. Write a Java program that checks for prime number using the object oriented approach.
[Hint: create a class NumberClass with a member value and method isPrimeNumber()]

NumberClass

```
class NumberClass {
int value;

       public NumberClass(int value) {
super();
          this.value = value;
       }
       public boolean isPrimeNumber() {
for(int i=2;i<value;i++) {
if(value==i) return false;
if(value%i==0) return false;
          }
          return true;
       }
    }
```

First

```
import java.util.Scanner; public
class First {
    public static void main(String[] args) {
// TODO Auto-generated method stub
Scanner s = new Scanner(System.in);        int
k = s.nextInt();
       NumberClass n = new NumberClass(k);
       System.out.println(n.isPrimeNumber());
       System.out.print();
    }

}
```

output:

```
56
false


Process finished with exit code 0
```

2. Create two classes: class Person
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 number of student objects created
- 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.

Person

```
public class Person {
String name;
int age;
    public Person(String name, int age) {
super();         this.name = name;
this.age = age;
    }
    public Person() {
super();
    }
    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 "Person [name=" + name + ", age=" + age + "]";
    }

}
```

Student

```
import java.util.Arrays;

public class Student extends Person {
static int count;
    @Override
    public String toString() {
        return “Student [count=” + count + “, rollno=” + rollno
+ “, marks=” + Arrays.toString(marks) + “]”;
    }
static {
count=0;
    }
{
        count++;
}
int rollno;
double[] marks;
    public Student(int rollno, double[] marks) {
super();
        this.rollno = rollno;
this.marks = marks;
    }
    public Student() {
super();
    }
    public Student(String name, int age, int rollno, double[]
marks) {
        super(name, age);
this.rollno = rollno;
this.marks = marks;
    }
    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;
    }
    public void displayDetails() {
        System.out.println(“RollNo = “+rollno);
        System.out.println(“Name = “+name);
```

```
        System.out.println("Age = "+rollno);
System.out.print("Marks = ");
for(double x:marks) {
System.out.print(x+" ");
        }
        System.out.println();
    }
    public double getAvgMarks() {
double avg =0;
for(int i=0;i<5;i++) {
avg+=marks[i];
        }
        return avg/5d;
    }
    public static int getCount() {
return count;
}
public static void setCount(int count) {
        Student.count = count;
    }
}
```

```
import java.util.*; public
class TestStudent {

    public static void main(String[] args) {
Student[] a = new Student[3];
        Scanner s = new Scanner(System.in);
         double avg=0,
maxAvg=0;         for(int
i=0;i<3;i++) {
avg=0;
            int r = s.nextInt();
String n = s.next();              int
ag = s.nextInt();             double
m[]=new double[5];
for(int j=0;j<5;j++) {
m[j] = s.nextDouble();
avg += m[j];
            }
            a[i] = new Student(n,ag,r,m);
avg=avg/5d;
             if(maxAvg < avg)
{                 maxAvg =
avg;
            }
            a[i].displayDetails();
        }
        for(int i=0;i<3;i++) {
            if(a[i].getAvgMarks() == maxAvg) {
                System.out.println("\nstudent with max. avg
marks:");
                a[i].displayDetails();
            }
        }
        System.out.println("\nTotal no. of students:");
        System.out.println(Student.getCount());
    }

}
```

output:

```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-java
1
jack
21
23 45 67 77 0
RollNo = 1
Name = jack
Age = 1
Marks = 23.0 45.0 67.0 77.0 0.0
2
hegg
21
90 90 99 54 98
RollNo = 2
Name = hegg
Age = 2
Marks = 90.0 90.0 99.0 54.0 98.0
3
jim
22
67 78 88 90 12
RollNo = 3
Name = jim
```

```
Age = 3
Marks = 67.0 78.0 88.0 90.0 12.0

student with max. avg marks:
RollNo = 2
Name = hegg
Age = 2
Marks = 90.0 90.0 99.0 54.0 98.0

Total no. of students:
3

Process finished with exit code 0
```

Laboratory Work

Subject: Java Technologies

Branch: B.Tech. (CE)

Semester: IV

Batch: A4

Student Roll No: CE079

Student Name: DOBARIYA VRUND GHANSHYAMBHAI

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

LAB 4

Topics: Interface, Exception Handling
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.

DivideByZero.class

```
import java.util.Scanner;
public class DivideByZero {
	public static void main(String[] args) {
		int n,k;
		Scanner s = new Scanner(System.in);
		n= s.nextInt();
		k= s.nextInt();
		try {
			int c=n/k;
		}
		catch(Exception e) {
			System.out.println(e);
		}
	}
}
```

output:

```
<terminated> DivideByZero [Java Application] /usr/lib/jvm/jdk-17/bin/java  (27-Dec-2023, 5:09:17 pm – 5:09:24 pm) [pid: 10492]
12
0
java.lang.ArithmeticException: / by zero
```

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.

CatchExcercise.class

```
import java.util.Scanner;
public class CatchExercise {
	public static void main(String[] args) {
		try {
			int arr[] = new int[5];
			arr[5]=3/0;
		}
		catch(ArithmeticException e) {
			System.out.println(e);
		}
		catch(ArrayIndexOutOfBoundsException  e) {
```

```
                System.out.println(e);
            }
        } }
```

output:

```
<terminated> CatchExercise [Java Application] /usr/lib/jvm/jdk-17/bin/java (27-Dec-2023, 5:12:58 pm – 5:12:58 pm) [pid: 10609]
java.lang.ArithmeticException: / by zero
```

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

A.class

```
import java.util.Scanner;
public class A {
        public static void main(String[] args) {
            Scanner s = new Scanner(System.in);
            int n = 25;
            int m = 5;
            try {
                int t=n/m;
            }
            catch(ArithmeticException e) {
                System.out.println(e);
            }
            catch(NullPointerException e) {
                System.out.println(e);
            }
            finally {
                System.out.println("Executed Finally block!");
            }
        }
}
```

B.class

```
import java.util.Scanner;
public class B {
        public static void main(String[] args) {
            Scanner s = new Scanner(System.in);
            int n = 25;
            int m = 0;
            try {
                int t=n/m;
            }
```

```
            catch(NullPointerException e) {
                  System.out.println(e);
            }
            finally {
                  System.out.println("Executed Finally block!");
            }
      }
}
```

C.class

```
import java.util.Scanner;
public class C {
      public static void main(String[] args) {
            Scanner s = new Scanner(System.in);
            int n = 25;
            int m = 0;
            try {
                  int t=n/m;
            }
            catch(ArithmeticException e) {
                  System.out.println(e);
            }
            finally {
                  System.out.println("Executed Finally block!");
            }
      }
}
```

output:

```
<terminated> A [Java Application] /usr/lib/jvm/jdk-17/bin/java  (27-Dec-2023, 5:15:54 pm – 5:15:54 pm) [pid: 10676]
Executed Finally block!
```

```
<terminated> B [Java Application] /usr/lib/jvm/jdk-17/bin/java  (27-Dec-2023, 5:17:03 pm – 5:17:03 pm) [pid: 10730]
Executed Finally block!
Exception in thread "main" java.lang.ArithmeticException: / by zero
        at B.main(B.java:10)
```

```
<terminated> C [Java Application] /usr/lib/jvm/jdk-17/bin/java (27-Dec-2023, 5:17:41 pm – 5:17:41 pm) [pid: 10772]
java.lang.ArithmeticException: / by zero
Executed Finally block!
```

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.

Account

```
public interface Account {
	void deposit(int amt);
	void withdraw(int amt);
}
```

SavingsAccount

```
public class SavingsAccount implements Account {
	int amt;
	{
		amt=0;
	}
	public SavingsAccount() {
		super();
	}
	public SavingsAccount(int amt) {
		super();
		this.amt = amt;
	}
	public int getAmt() {
		return amt;
	}
	public void setAmt(int amt) {
		this.amt = amt;
	}
	@Override
	public void deposit(int amt) {
		this.amt+=amt;
	}
	@Override
	public void withdraw(int amt) {
		try {
			if(this.amt < amt) {
				CustomException e = new CustomException();
				throw e;
			}
		}
```

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

CustomException

```
public class CustomException extends Exception {
      @Override
      public String toString() {
            return "Exception [Account Balance is to low!]";
      }
      public CustomException() {
            super();
            // TODO Auto-generated constructor stub
      }
      public CustomException(String message, Throwable cause, boolean
enableSuppression, boolean writableStackTrace) {
            super(message, cause, enableSuppression, writableStackTrace);
            // TODO Auto-generated constructor stub
      }
      public CustomException(String message, Throwable cause) {
            super(message, cause);
            // TODO Auto-generated constructor stub
      }
      public CustomException(String message) {
            super(message);
            // TODO Auto-generated constructor stub
      }
      public CustomException(Throwable cause) {
            super(cause);
            // TODO Auto-generated constructor stub
      }

}
```

Fourth

```
import java.util.Scanner;
public class Fourth {
      public static void main(String[] args) {
            // TODO Auto-generated method stub
            Scanner s = new Scanner(System.in);
            int amount = s.nextInt();
            SavingsAccount a = new SavingsAccount(amount);
            a.withdraw(1200);
      }
}
```

output:

```
<terminated> Fourth [Java Application] /usr/lib/jvm/jdk-17/bin/java  (27-Dec-2023, 5:22:10 pm – 5:22:16 pm) [pid: 10847]
121
Exception [Account Balance is to low!]
```

Laboratory Work

Subject: Java Technologies Branch:

B.Tech. (CE) Semester: IV

Batch **A4**

Student Roll No: **CE079**

Student Name: **DOBARIYA VRUND GHANSHYAMBHAI**

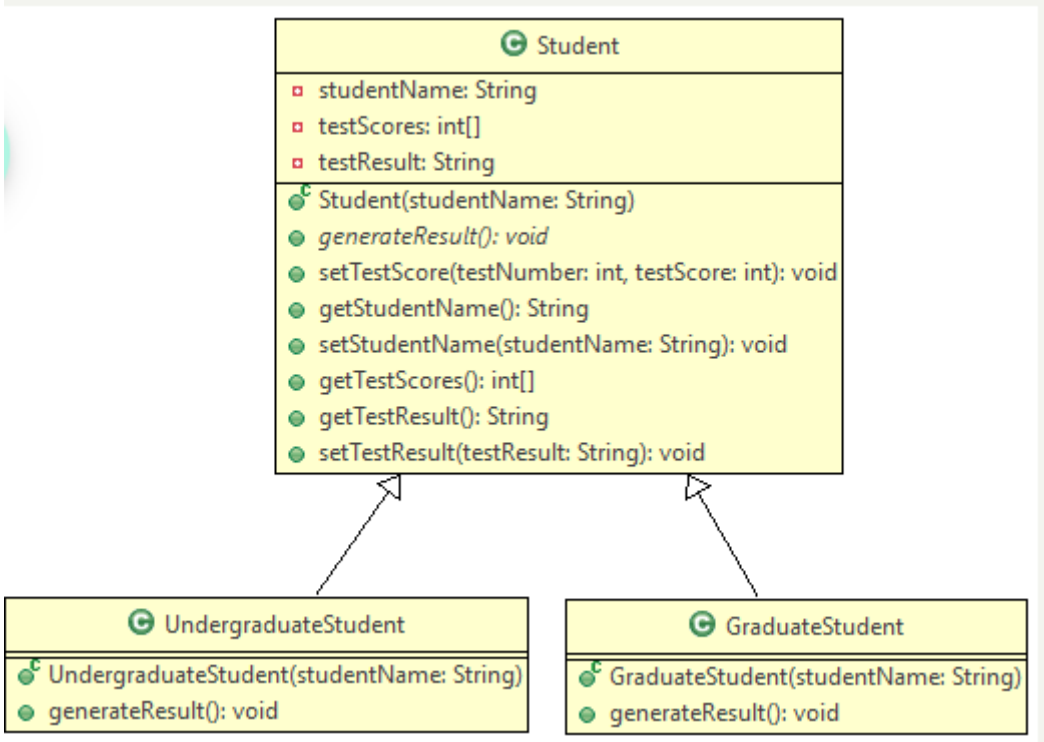
Department of Computer Engineering,

Faculty of Technology,

Dharmsinh Desai University, Nadiad – 387001.

Lab-5
Topics : Abstract class , Interface , Multithreading

1. Anchor College offers both UnderGraduate and PostGraduate programs. The college stores the names of the students, their test scores and the final result for each student. Each student has to take 4 tests in total. You need to create an application for the college by implementing the classes based on the class diagram and description given below.



Method Description

Implement the getter and setter methods appropriately.

1) **Student(Class)**
   I. **Student(String studentName)**
   - Initialize the instance variable studentName with the value passed to the constructor and other instance variables to the default values.

   II. **setTestScore(int testNumber, int testScore)**
   - Set the value of the testScore in the appropriate position of testScores array based on the testNumber.

2) **UndergraduateStudent(Class)**
   I. **UndergraduateStudent(String studentName)**
   - Initialize the instance variable studentName with the value passed to the constructor and other instance variables to the default values.

   II. **generateResult()**

- Implement the abstract method of Student class by setting the value of testResult based on the below details.

Average Score	**Result**
>=60	Pass
<60	Fail

Sample Input and Output For UndergraduateStudent

Input:-

Instance Variable	**Values**
name	**Jerry**
testScores	**{70,69,71,55 }**

Output:-

Student Name : Jerry

Result : Pass

3) PostGraduateStudent(Class)

I. PostgraduateStudent(String studentName)

- Initialize the instance variable studentName with the value passed to the constructor and other instance variables to the default values.

II. generateResult()

- Implement the abstract method of Student class by setting the value of testResult based on the below details.

Average Score	**Result**
>=75	Pass
<75	Fail

Sample Input and Output For PostgraduateStudent

Input:

Instance Variable	Values
name	Tom
testScores	{70,75,80,85}

Output: Student Name : Tom

Result : Pass

Student.class

```
import java.util.Arrays;

public abstract class Student {
    private String studentName;
    private int[] testScores;
    private String testResult;

    {
       int[] a=new int[4];
       for(int i=0;i<4;i++) a[i]=0;
       testScores=a;
       testResult="Not generated yet!";
    }

    //constructors
    public Student() {
       super();
    }
    public Student(String studentName) {
       super();
       this.studentName = studentName;
    }
    //getters and setters
    public String getStudentName() {
       return studentName;
    }

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

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

```java
    }

    public String getTestResult() {
        return testResult;
    }

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

    @Override
    public String toString() {
        return "studentName= " + studentName + "\ntestResult= "
                + testResult;
    }
    abstract public void generateResult();
    public void setTestScore(int testNumber, int testScore) {
        testScores[testNumber] = testScore; //zero based indexing
    }
}
```

GraduateStudent.class

```java
public class GraduateStudent extends Student {
    //constructor

    public GraduateStudent(String studentName) {
        super(studentName);
    }

    @Override
    public void generateResult() {
        double avgMarks = 0;
        for(int i=0;i<4;i++) {
            avgMarks+=getTestScores()[i];
        }
        avgMarks=avgMarks/4d;
        if(avgMarks>=75) setTestResult("Pass");
        else setTestResult("Fail");
    }

}
```

UndergraduateStudent.class

```
public class UndergraduateStudent extends Student {
    //constructor
    public UndergraduateStudent(String studentName) {
        super(studentName);
    }
    @Override
    public void generateResult() {
        double avgMarks = 0;
        for(int i=0;i<4;i++) {
            avgMarks+=getTestScores()[i];
        }
        avgMarks=avgMarks/4d;
        if(avgMarks>=60) setTestResult("Pass");
        else setTestResult("Fail");
    }
}
```

Driver1.class

```
import java.util.*;
public class Driver1 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        UndergraduateStudent s1 = new
UndergraduateStudent(sc.next());

        for(int i=0;i<4;i++) {
            s1.setTestScore(i, sc.nextInt());
        }
        s1.generateResult();
        System.out.println(s1);

        GraduateStudent s2 = new GraduateStudent(sc.next());

        for(int i=0;i<4;i++) {
            s2.setTestScore(i, sc.nextInt());
        }
        s2.generateResult();
        System.out.println(s2);
    }

}
```

Output:

```
Javadoc  Declaration  Console ×  Terminal
<terminated> Driver [Java Application] /usr/lib/jvm/jdk-17/bin/java (10-Jan-2024, 4:44:30 pm – 4:44:56 pm) [pid: 9186]
Jarry
70 69 71 55
studentName= Jarry
testResult= Pass
Tom
70 75 80 85
studentName= Tom
testResult= Pass
```

2. Write a Java program as per the given description to demonstrate use of interface.
   I. 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.
   II. 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
   III. In class **CompareLines**.Java, create two objects of Line class, call the three methods to compare the lengths of the lines.

Line.class

```java
public class Line implements RelationalInterface {
    double x1,x2,y1,y2;

    public Line(double x1, double x2, double y1, double
y2) {
        super();
        this.x1 = x1;
        this.x2 = x2;
        this.y1 = y1;
        this.y2 = y2;
    }
    @Override
    public boolean isGreater(Line l) {
        return this.getLength()>l.getLength();
    }

    @Override
    public boolean isLess(Line l) {
```

```
        return this.getLength()<l.getLength();
    }

    @Override
    public boolean isEqual(Line l) {
        return this.getLength()==l.getLength();
    }

    double getLength() {
        return Math.sqrt((x2-x1)*(x2-x1)+(y2-y1)*(y2-y1));
    }
}
```

CompareLines

```
public class CompareLines {
    public static void main(String[] args) {
        Line l1 = new Line(1,2,3,4);
        Line l2 = new Line(3,4,5,6);
        System.out.println(l1.isEqual(l2));
        System.out.println(l1.isGreater(l2));
        System.out.println(l1.isLess(l2));
    }

}
```

RelationalInterface

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

Output:

```
Javadoc  Declaration  Console ×
<terminated> CompareLines [Java Application] /usr/lib/jvm/jdk-17/bin/java  (10-Jan-2024, 5:01:50 pm – 5:01:50 pm) [pid:
true
false
false
```

3. In the producer–consumer problem, the producer and the consumer share 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 all necessary classes to achieve this.

Buffer

```
package cp;
public class Buffer {
    String bufferData="DATA";
    boolean isSet=false;
    synchronized void put(String s) {
       while(isSet) {
            try {
                wait();
            } catch (InterruptedException e) {
                throw new RuntimeException(e);
            }
        }
       bufferData = s;
       System.out.println("produced "+bufferData);
       isSet=true;
       notify();
    }
    synchronized String get() {
       while(!isSet) {
            try {
                wait();
            } catch (InterruptedException e) {
                throw new RuntimeException(e);
            }
        }
       System.out.println("consumed "+bufferData);
       isSet=false;
       notify();
       return bufferData;
    }
    public Buffer() {}
}
```

Producer

```
package cp;
public class Producer extends Thread {
    Buffer b;
    Producer(Buffer b) {
       super();
       this.b=b;
    }
    @Override
    public void run() {
       while(true) {
          b.put("DATA");
```

```
            try {
                Thread.sleep(1000);
            }
            catch(Exception e) {
                System.out.println(e);
            }
        }
    }
}
```

Consumer

```
package cp;
public class Consumer extends Thread {
    Buffer b;
    Consumer(Buffer b) {
        super();
        this.b=b;
    }
    @Override
    public void run() {
        while(true) {
            b.get();
            try {
                Thread.sleep(1000);
            }
            catch(Exception e) {
                System.out.println(e);
            }
        }

    }

}
```

Driver2

```
package cp;
import java.lang.Thread;
public class Driver2 {

    public static void main(String[] args) {
        System.out.println("Main thread started executing...");
        Buffer b = new Buffer();
        Consumer c = new Consumer(b);
        Producer p = new Producer(b);
        Thread p2 = new Thread(c);
        Thread p1 = new Thread(p);
        System.out.println("Producer strted executing...");
        p1.start();
        System.out.println("Consumer started executing...");
        p2.start();
    }

}
```

Output:

```
Run    Driver2
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2023.2.5\lib\idea_rt.jar=52866:C
Main thread started executing...
Producer started executing...
Consumer started executing...
Main thread ended execution
produced DATA
consumed DATA
produced DATA
consumed DATA
produced DATA
consumed DATA
produced DATA
```

4. Write a multithreaded Java application to produce a deadlock condition.

Deadlock.class

```
public class Deadlock {
    public static void main(String[] args) {
        final String resource1 = "vrund";
        final String resource2 = "student";

        Thread t1 = new Thread() {
            public void run() {
                synchronized (resource1) {
                    System.out.println("Thread 1: locked resource
1");

                    try { Thread.sleep(100);} catch (Exception e)
{}

                    synchronized (resource2) {
                        System.out.println("Thread 1: locked
resource 2");
                    }
                }
            }
        };

        // t2 tries to lock resource2 then resource1
        Thread t2 = new Thread() {
            public void run() {
                synchronized (resource2) {
                    System.out.println("Thread 2: locked resource
2");

                    try { Thread.sleep(100);} catch (Exception e)
{}

                    synchronized (resource1) {
                        System.out.println("Thread 2: locked
resource 1");
                    }
```

```
                }
            }
        };


        t1.start();
        t2.start();
    }
}
```

Output:

Subject: Java Technologies

Branch: B.Tech. (CE) Semester: IV

Batch: **A4**

Student Roll No: **CE079**

Student Name: **DOBARIYA VRUND GHANSHYAMBHAI**

Department of Computer Engineering,

Faculty of Technology,

Dharmsinh Desai University, Nadiad – 387001.

LAB-6
Topics: JDBC, Generics

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().

ManageStudentDetails

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

public class ManageStudentDetails {
 public static void main(String[] a) {

 Scanner sc = new Scanner(System.in);

 //Creating connection object
 try(Connection c =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:orcl","VGD"
, "vrund3626")) {

 Statement s1 =
c.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
ResultSet.CONCUR_UPDATABLE);
 ResultSet rs = s1.executeQuery("SELECT ID, FIRSTNAME,  LASTNAME,
BRANCH, USERNAME, PASSWORD FROM VGD.STUDENT");

 System.out.println(rs.isBeforeFirst());

 while(rs.next()) {

 if(rs.isFirst()==true)
 System.out.println("Details of the first student");

 if(rs.isLast()==true)
 System.out.println("Details of the last student");

 System.out.println(rs.getString(1));
 System.out.println(rs.getString(2));
 System.out.println(rs.getString(3));
 System.out.println(rs.getString(4));
 System.out.println(rs.getString(5));
 System.out.println("------------------------");
rs.updateString(2,"123");
 }

 System.out.println(rs.isAfterLast());

 rs.first();
 System.out.println(rs.isFirst());
 rs.last();
 System.out.println(rs.isLast());
 PreparedStatement s2 = c.prepareStatement("INSERT INTO
STUDENT(ID, FIRSTNAME, LASTNAME, BRANCH, USERNAME, PASSWORD)
```

```
VALUES(?,?,?,?,?,?)");


 for(int i=0;i<2;i++) {

 s2.setInt(1,sc.nextInt());
 s2.setString(2,sc.next());
 s2.setString(3,sc.next());
 s2.setString(4,sc.next());
 s2.setString(5,sc.next());
 s2.setString(6,sc.next());
 s2.executeUpdate();
 }

 rs = s1.executeQuery("SELECT * FROM STUDENT");

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

Input:

```
5
abc
abc
abc
abc
abc
6
xyz
xyz
zzz
zzz
zzz
```

Output:

```
true
Details of the first student
1
jack
sparrow
CE
pirat
------------------------
2
tony
stark
EC
ironman
------------------------
3
tom
hardy
```

```
ME
venom
------------------------
Details of the last student
4
John
Doe
IT
JDK
------------------------
true
true
true
```

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

Movie.java

```java
public class Movie {
 private int id;
 private String genre;
 private String title;
 private int yearOfRelease;

 public Movie(int id, String title, String genre, int yearOfRelease)
 {
 this.id = id;
 this.genre = genre;
 this.title = title;
 this.yearOfRelease = yearOfRelease;
 }

 public int getId() {
 return id;
 }
 public String getGenre() {
```

```
 return genre;
 }
 public String getTitle() {
 return title;
 }
 public int getYearOfRelease() {
 return yearOfRelease;
 }
}
```

ManageMovies.java

```
import java.sql.*;

public class ManageMovies {
 Connection c;
 ManageMovies() throws SQLException {
 this.c =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:orcl","
VG D","vrund3626");
 }
 void createMovie(Movie m) throws SQLException {
 int id = m.getId();
 String title = m.getTitle();
 String genre = m.getGenre();
 int yearOfRelease = m.getYearOfRelease();
 Statement s = c.createStatement();
 String insQuery = "INSERT INTO MOVIES(ID, TITLE, GENRE,
YEAROFRELEASE) VALUES("+ id +",'"+ title + "','" + genre +"',"
+  yearOfRelease + ")";
// System.out.println(insQuery);
 s.executeUpdate(insQuery);
 }

 void deleteMovie(int movieId) throws SQLException {
Statement s = c.createStatement();
 String delQuery = "DELETE FROM MOVIES WHERE ID=" + movieId;
s.executeUpdate(delQuery);
 System.out.println("\n\nMovie with id= "+movieId+" is
deleted");
 }

 void updateMovieTitle(String title, int id) throws SQLException {
Statement s = c.createStatement();
 String upQuery = "UPDATE MOVIES SET TITLE ='" + title + "'
WHERE ID=" + id;
 s.executeUpdate(upQuery);
 System.out.println("");
 System.out.print("\n"+"Movie title updated for id= "+id);  }

 void findMovieById(int id) throws SQLException {
 Statement s = c.createStatement();
```

```
 String Query = "SELECT ID,TITLE,GENRE,YEAROFRELEASE FROM MOVIES
WHERE ID=" + id;
 s.executeQuery(Query);
 ResultSet rs = s.executeQuery(Query);
 if(rs==null) {
 System.out.println("Movie not found");
 return;
 }
 rs.next();
 System.out.println("\n");
 System.out.println("found the movie with id "+id);
System.out.print(rs.getInt(1)+" ");
 System.out.print(rs.getString(2)+" ");
 System.out.print(rs.getString(3)+" ");
 System.out.print(rs.getString(4)+" ");
 }

 void findAllMovie() throws SQLException {
 Statement s = c.createStatement();
 String Query = "SELECT ID,TITLE,GENRE,YEAROFRELEASE FROM
MOVIES";
 ResultSet rs = s.executeQuery(Query);
 while(rs.next()) {
 System.out.println("");
 System.out.print(rs.getInt(1)+" ");
      System.out.print(rs.getString(2)+"      ");
System.out.print(rs.getString(3)+"            ");
System.out.println(rs.getString(4));
 System.out.print("----------------------");  }
 }

}
```

Driver.java

```
import java.sql.SQLException;

public class Driver {
 public static void main(String[] a) throws SQLException {
Movie m1 = new Movie(1,"Oppenheimer","Thriller",2023);  Movie m2
= new Movie(2,"12th fail","Drama",2023);  Movie m3 = new
Movie(3,"Marry Christmas","Thriller",2024);  Movie m4 = new
Movie(4,"Dunki","Comedy",2023);  ManageMovies mm = new
ManageMovies();
 mm.createMovie(m1);
 mm.createMovie(m2);
 mm.createMovie(m3);
 mm.createMovie(m4);
 mm.findAllMovie();
 mm.findMovieById(4);
 mm.updateMovieTitle("12th Fail", 2);
 mm.deleteMovie(4);
 mm.findAllMovie();
```

```
 }
}
```

Output:

```
1 Oppenheimer Thriller 2023
----------------------
2 12th Fail Drama 2023
----------------------
3 Marry Christmas Thriller 2024
----------------------
4 Dunki Comedy 2023
----------------------

found the movie with id 4
4 Dunki Comedy 2023

Movie title updated for id= 2

Movie with id= 4 is deleted

1 Oppenheimer Thriller 2023
----------------------
2 12th Fail Drama 2023
----------------------
3 Marry Christmas Thriller 2024
----------------------
Process finished with exit code 0
```

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.

Calculator.java

```
public class Calculator<T extends Number> {
 double addition(T op1, T op2) {
 return op1.doubleValue() + op2.doubleValue();
 }

 double subtraction(T op1, T op2) {
 return op1.doubleValue() - op2.doubleValue();
 }

 double multiplication(T op1, T op2) {
 return op1.doubleValue() * op2.doubleValue();
 }

 double division(T op1, T op2) throws Exception {
 if(op2.doubleValue()==0)
 throw new Exception("divide by zero!");
 return op1.doubleValue()/op2.doubleValue();
 }
```

```
}
```

CalculatorTester.java

```
public class CalculatorTester {
 public static void main(String[] abc) {
 Calculator<Integer> c = new Calculator<Integer>();
Integer op1 = new Integer(23);
 Integer op2 = new Integer(2);

 System.out.println(c.addition(op1,op2));
System.out.println(c.subtraction(op1,op2));
System.out.println(c.multiplication(op1,op2));  try {
 System.out.println(c.division(op1, op2));  }
 catch (Exception e) {
 System.out.println(e.getMessage());  }

 Calculator<Double> d = new Calculator<Double>();
Double a = new Double(345.3);
 Double b = new Double(45);

 System.out.println(d.addition(a,b));
System.out.println(d.subtraction(a,b));
System.out.println(d.multiplication(a,b));  try {
 System.out.println(d.division(a,b));  }
 catch (Exception e) {
 System.out.println(e.getMessage());  }

 }
}
```

Output:

```
25.0

21.0

46.0

11.5

390.3

300.3

15538.5

7.673333333333334
```

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.

CheckSameArr.java

```
public class CheckSameArr<T> {
 T[] a;
 T[] b;
```

```
 public CheckSameArr(T[] a, T[] b) {
 this.a = a;
 this.b = b;
 boolean f=true;
 if(a.length != b.length)
 f=false;
 for(int i=0;i<Math.min(a.length,b.length);i++) {
if(!a[i].equals(b[i]))
 f=false;
 }
 if(!f)
 System.out.println("Not same");
 else
 System.out.println("Same");
 }
}
```

Driver2.java

```
public class Driver2 {
 public static void main(String[] ae) {
 Integer[] a = new Integer[3];
 Integer[] b = new Integer[3];
 a[0]=new Integer(1); b[0]= new Integer(1);
 a[1]=new Integer(2); b[1]= new Integer(2);
 a[2]=new Integer(999); b[2]= new Integer(999);
 CheckSameArr<Integer> ck = new CheckSameArr<>(a,b);   }
}
```

Output:

```
Same
```

LAB 7

Topics : Collection , I/O

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 the first file to the second file.

FileOps2.java

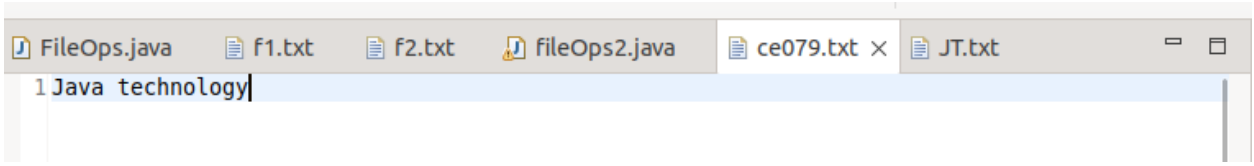
```
import java.util.*;
import java.io.*;
public class fileOps2 {
      public static void main(String[] args) throws IOException {
            Scanner s = new Scanner(System.in);
            String file1 = s.next(), file2 = s.next(), op =
s.next();
            boolean flag = true;
            if(op == "copy")
                  flag=false;

            FileInputStream source = null;
            FileOutputStream destination = null;
            try {
                  source = new FileInputStream(file1);
                  destination = new FileOutputStream(file2,flag);
                  int c;
                  while((c=source.read()) != -1) {
                        destination.write(c);
                  }
            }
            finally {
                  if(source != null)
                        source.close();
                  if(destination != null)
                        destination.close();
            }

      }
}
```

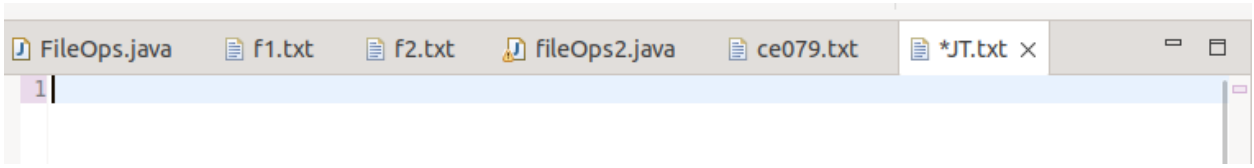
Input:

ce079.txt

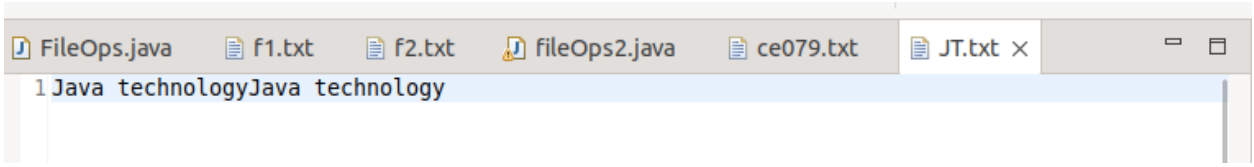
JT.txt

Append





Output:



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.

Second.java

```
import java.util.*;
public class Second {
	public static void main(String[] args) {
		LinkedList<Student> l = new LinkedList<> ();
		l.push(new Student("dbc",11));
		l.push(new Student("ccw",1));
		l.push(new Student("bba",8));
		l.push(new Student("bbc",23));
		l.push(new Student("abc",78));
//		Comparator<Student> c = new Comparator<> ();
		Collections.sort(l);

		for(Student x : l) {
			System.out.println(x);
		}

	}
}
```

Student.java

```
public class Student implements Comparable<Student> {
	String name;
	int rollNo;
	Student(String s, int e) {
		this.name=s; this.rollNo=e;
	}
	@Override
	public int compareTo(Student o) {
		if(this.name.compareTo(o.name) > 0)
			return 1;
		else if(this.name.compareTo(o.name) < 0)
			return -1;
		else
			return 0;
	}
	@Override
	public String toString() {
		return "Student [name=" + name + ", rollNo=" + rollNo + "]";
	}

}
```

Output:

```
Student [name=abcw, rollNo=1]
Student [name=abc, rollNo=8]
Student [name=abc, rollNo=11]
Student [name=abc, rollNo=23]
Student [name=abc, rollNo=78]
```

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.

Person.class

```java
public class Person implements Comparable<Person> {
   String name;
   int age;
   public Person(String name, int age) {
       super();
       this.name = name;
       this.age = age;
   }

   @Override
   public int compareTo(Person o) {
       return (this.name).compareTo(o.name);
   }
}
```

Main.class

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

public class Main {
   public static void main(String[] args) {
       Scanner s = new Scanner(System.in);
       TreeMap<Person, String> m = new TreeMap<>();

       m.put(new Person("Bhairavi", 22), "Singing");
       m.put(new Person("Dhara", 23), "Sketching");
       m.put(new Person("Anmol", 23), "Reading");
       m.put(new Person("Megh", 21), "Singing");
       m.put(new Person("Raag", 22), "Sketching");

       Set<String> st = new HashSet<>();

       for(Map.Entry<Person, String> e: m.entrySet()) {
           st.add(e.getValue());
       }

       st.forEach(System.out::println);
```

```
    }
}
```

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.

MinMax.java

```
public interface MinMax<T> {
    void findMin();
    void findMax();
}
```

Book.java

```
public class Book {
   double p;

   public double getP() {
       return p;
   }

   public void setP(double p) {
       this.p = p;
   }

   public Book(double p) {
       this.p = p;
   }
}
```

Myclass.java

```
import java.util.Arrays;
import java.util.Comparator;
public class MyClass implements MinMax {
```

```java
    Book [] b;

    @Override
    public void findMin() {
        C c = new C();
        System.out.println(Arrays.stream(b).max(c));
    }

    @Override
    public void findMax() {
        C c = new C();
        System.out.println(Arrays.stream(b).min(c));
    }
}
```

C.java

```java
import java.util.Comparator;

public class C implements Comparator<Book> {

    @Override
    public int compare(Book o1, Book o2) {
        if(o1.getP() < o2.getP())
            return -1;
        else if(o1.getP() == o2.getP())
            return 0;
        else
            return 1;
    }
}
```

Laboratory Work

Subject: Java Technologies

Branch: B.Tech. (CE)

Semester: IV

Batch: A4

Student Roll No: CE079

Student Name: DOBARIYA VRUND GHANSHYAMBHAI

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

Lab-8

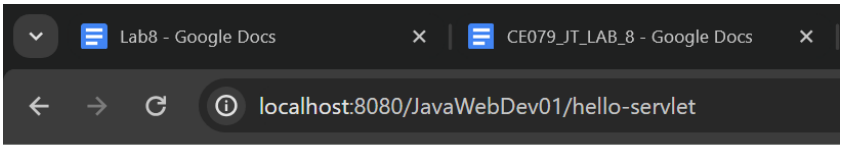
Topics: Servlet, JSP

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

HelloServlet.java

```
package com.example.javawebdev01;

import java.io.*;
import jakarta.servlet.http.*;
import jakarta.servlet.annotation.*;

@WebServlet(name = "helloServlet", value =
"/hello-servlet")
public class HelloServlet extends HttpServlet {
   private String message;

   public void init() {
       message = "Welcome!";
   }

   public void doGet(HttpServletRequest request,
HttpServletResponse response) throws IOException {
       response.setContentType("text/html");

       // Hello
       PrintWriter out = response.getWriter();
       out.println("<html><body>");
       out.println("<h1>" + message + "</h1>");
       out.println("</body></html>");
   }

   public void destroy() {
   }
}
```

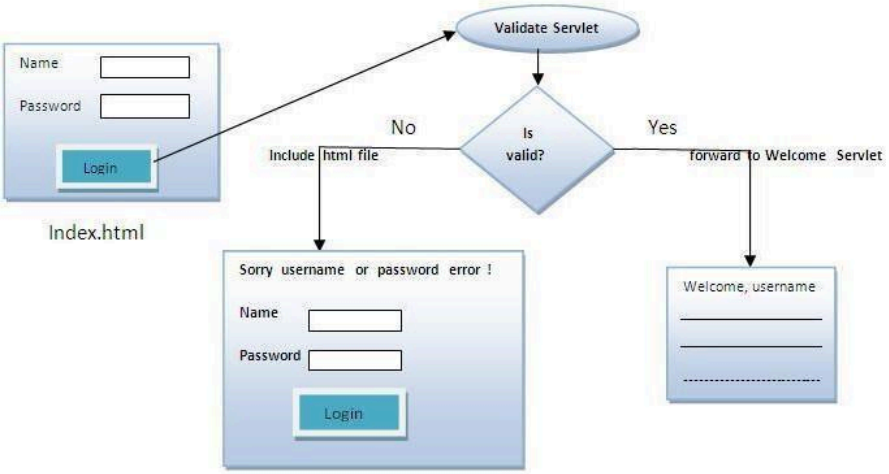
Output:



Welcome!

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>Insert title here</title>
</head>
<body>
<form action='submit' method='post'>
   <label for='name'>
       username
   </label>
   <input type='text' id='name' name='name' >
   <br>
   <br>
   <label for='pass'>
       password
   </label>
   <input type='password' id='pass' name='pass'>
   <br>
   <br>
   <input type='submit' value='log in'>

</form>
</body>
</html>
```

login.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>
hello
<% String p=request.getParameter("name");
   out.print(p);
%>
</body>
</html>
```

validateServlete.java

```
import java.sql.*;
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;

public class validateServlet extends HttpServlet {
   private static final long serialVersionUID = 1L;

    public validateServlet() {
       super();
   }


   protected void doGet(HttpServletRequest request,
HttpServletResponse response) throws ServletException,
IOException {

       response.sendRedirect("index.html");

   protected void doPost(HttpServletRequest request,
HttpServletResponse response) throws ServletException,
IOException {

       try (Connection con =
DriverManager.getConnection("jdbc:mysql://192.168.29.150:3306/c
e4_79", "ce4_79", "ce4_79")){
```

```
            String query="SELECT * FROM `student` WHERE 1";
            PreparedStatement p=con.prepareStatement(query);
        }
        catch(Exception e) {

        }

        String name=request.getParameter("name");
        String password=request.getParameter("pass");
        if(name.equals("vrund") && password.equals("1234")) {
            RequestDispatcher
rd=request.getRequestDispatcher("login.jsp");
            rd.forward(request,response);
        }
        else {
            response.sendRedirect("index.html");
        }

    }

}
```

Laboratory Work

Subject: Java Technologies

Branch: B.Tech. (CE)

Semester: IV

Batch: A4

Student Roll No: CE079

Student Name: DOBARIYA VRUND GHANSHYAMBHAI

Department of Computer Engineering,
Faculty of Technology,

Dharmsinh Desai University, Nadiad – 387001.
Gujarat, INDIA.

LAB-9

Topics: JSP, Servlet, Session Management in web application

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)

LoginVerifierServlet.java

```
package com.example.javawebdev01;
import java.io.*;
import jakarta.servlet.RequestDispatcher;
import jakarta.servlet.ServletException;
import jakarta.servlet.http.*;
import jakarta.servlet.annotation.*;

@WebServlet(name = "LoginVerifierServlet", value =
"/login")
public class LoginVerifierServlet extends
HttpServlet {
   private String message;

   public void init() {
   }

   public void doPost(HttpServletRequest request,
HttpServletResponse response) throws IOException,
ServletException {
       response.setContentType("text/html");
       PrintWriter o = response.getWriter();
       String uname =
request.getParameter("username");
       String pass =
request.getParameter("password");
```

```
        if(uname.equals("vrund") &&
pass.equals("1234")) {
            RequestDispatcher rd =
request.getRequestDispatcher("first.jsp");
            HttpSession s = request.getSession();
            s.setAttribute("uname", "vrund");
            s.setAttribute("pass", "1234");
            rd.forward(request, response);
        }
        else {
            o.println("<p>invalid
credentials!</p>");
            RequestDispatcher rd =
request.getRequestDispatcher("index.jsp");
            rd.include(request, response);
        }

    }

    public void destroy() {
    }
}
```

index.jsp

```
<%@ page contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8" %>
<!DOCTYPE html>
<html>
<head>
 <meta charset="UTF-8">
 <title>login</title>
</head>
<body>
<form method="POST" action="login">
 <div class="container">
   <label>Username : </label>
   <input type="text" placeholder="Enter Username"
name="username" required>
   <label>Password : </label>
   <input type="password" placeholder="Enter
Password" name="password" required>
   <button type="submit">Login</button>
 </div>
</form>
</body>
```

```
</html>
```

first.jsp

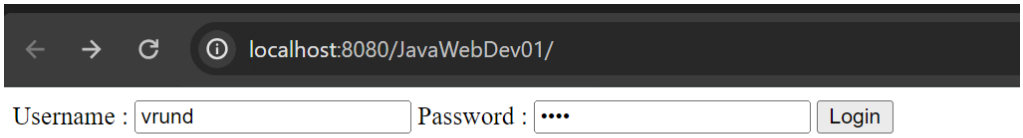
```
<%@ page language="java" contentType="text/html;
charset=UTF-8"
        pageEncoding="UTF-8" %>
<!DOCTYPE html>
<html>
<head>
   <meta charset="UTF-8">
   <title>Insert title here</title>
</head>
<body>
<h1>

   <%

       if
(request.getSession(false).getAttribute("uname") ==
null) {
           RequestDispatcher rd =
request.getRequestDispatcher("index.jsp");
           rd.forward(request, response);
       }

   %>
   ${ "Hello " }
   ${ uname }
   <br>
   ${ "password: "} ${ pass }
   <br>
   <a href="second.jsp">second.jsp</a>
</h1>
</body>
</html>
```

second.jsp

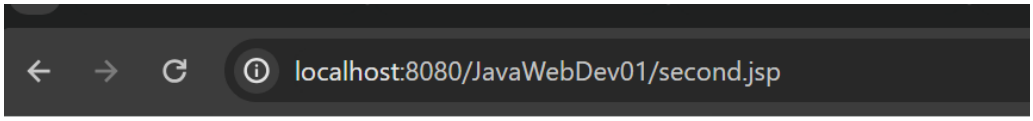
```
<%@ page contentType="text/html;charset=UTF-8"
language="java" %>
<html>
<head>
   <title>second</title>
</head>
```

```
<body>
<p>username: ${ uname } and password: ${ pass } are
also available here... </p>
</body>
</html>
```

OUTPUT :



Hello vrund
password: 1234
second.jsp



username: vrund and password: 1234 are also available here...

invalid credentials!

Username : Enter Username Password : Enter Password Login

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.

CalcBean.java

```
package beans;
public class CalcBean {
   int num1;
   int num2;
   char op;
   int result;
   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 char getOp() {
       return op;
   }
   public void setOp(char op) {
       this.op = op;
   }
   public int getResult() {
       if(getOp()=='+') {
           setResult(getNum1()+getNum2());
       }
       else if(getOp()=='-') {
           setResult(getNum1()-getNum2());
       }
       else if(getOp()=='*') {
           setResult(getNum1()*getNum2());
       }
       else{
           setResult(getNum1()/getNum2());
       }
       return result;
   }
   public void setResult(int result) {
```

```
        this.result = result;
    }
    @Override
    public String toString() {
         return "CalcBean [num1=" + num1 + ", num2=" + num2 +
", op=" + op + ",result=" + result + "]";
    }
}
```

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>
   <input type="number" name="n1" value='<%=
request.getParameter("n1") %>'>
   <br>
   <br>
   <select name="op" >
       <option value='+'  <%
if("+".equals(request.getParameter("op"))) {
%>selected<% } %> >+</option>
       <option value='-'  <%
if("-".equals(request.getParameter("op"))) {
%>selected<% } %>>-</option>
       <option value='*'  <%
if("*".equals(request.getParameter("op"))) {
%>selected<% } %>>*</option>
```

```
        <option value='/'  <%
if("/".equals(request.getParameter("op"))) {
%>selected<% } %>>/</option>
   </select>
   <br>
   <br>
   <input type="number" name="n2"  value='<%=
request.getParameter("n2") %>'>
   <br>
   <br>
   <input type="submit" value="submit">
   <br>
   <br>
</form>
<% if(request.getParameter("op")!=null){ %>

<jsp:useBean id="c" class="beans.CalcBean"
            scope="request"></jsp:useBean>
<jsp:setProperty property="num1" name="c" param="n1" />
<jsp:setProperty property="num2" name="c" param="n2" />
<jsp:setProperty property="op" name="c" param="op" />

<h1>Result:  <jsp:getProperty property="result" name="c"/>
</h1>

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

localhost:8080/JavaWebDev01/calc.jsp?n1=34&op=%2B&n2=23

34

+

23

submit

Result: 57