Thangal Kunju Musaliar College of Engineering
Kollam, Kerala

CSL203 – OBJECT ORIENTED PROGRAMMING LAB IN JAVA

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

2020 - 2021

Thangal Kunju Musaliyar College of Engineering
Kollam, Kerala

CSL203

OBJECT ORIENTED PROGRAMMING LAB IN JAVA

CERTIFICATE

Certified that this is a bonafide record of the work done by **Dinoy Raj K** of **Third** semester, roll no **TKM19CS021** in the OBJECT ORIENTED PROGRAMMING Laboratory during the academic year 2020 - 2021

Reg no: **190723**

Name of the Examination: **Third** Semester B.Tech Degree Examination

Staff in Charge

External Examiner

VISION

To be a centre of excellence imparting quality education in Computer Science and Engineering and transforming students to critical thinkers and lifelong teams capable of developing environment friendly and economically feasible solutions to real world problems

MISSION

- To provide a strong foundation in Computer Science and Engineering, prepare students for professional career and higher education, and inculcate research interest.
- To be abreast of the technological advances in a rapidly changing world.
- To impart skills to come up with socially acceptable solutions to real world problems, upholding ethical values.

PROGRAMME EDUCATIONAL OBJECTIVES(PEOs)

PEO 1: Excel in professional career by acquiring knowledge in mathematics, science and, engineering and applying the knowledge in the design of hardware and software solutions for challenging problems of the society, adapting to the current tends by engaging in lifelong learning

PEO 2: Pursue higher studies and research in the area of Computer Science and Engineering

PEO 3: Ability to provide socially acceptable and economically feasible computer-oriented solutions to real world problems with teamwork, while maintaining environmental balance, quality and cognizance of the underlying principles of ethics

PROGRAM SPECIFIC OUTCOMES

- **PSO 1**: Apply mathematical and algorithmic principles, data structure concepts, software and hardware techniques in designing and developing optimized and secure computer-based solutions.

- **PSO 2**: Design and develop system software and provide exposure to various tools and programming languages to facilitate efficient computing environment which adds to the case of human life.

- **PSO 3**: Use the knowledge of various data processing, communication and intelligent systems to provide solutions to new ideas and innovations.

PROGRAMME OUTCOMES

- PO 1 - **Engineering Knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problem.

- PO 2 - **Problem Analysis**: Identify formulate, review research literature and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering.

- PO 3 - **Design Development of Solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specific needs with appropriate consideration for the public health and safety, and the cultural, societal, environmental considerations.

- PO 4 - **Conduct Investigation of Complex Problem**: Use research-based knowledge and reached including design of experiments analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

- PO 5 - **Modern Tool Usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

- PO 6 - **The Engineer and Society**: Apply reasoning informed by the contextual knowledge to asses societal, health, safety. legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

- PO 7 - **Environment and Sustainability**: Understand the impact of professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

- PO 8 - **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

- PO 9 - **Individual and Team Work**: Function effectively as an individual and as a member or leader in diverse and in multidisciplinary settings.

- PO 10 - **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

- PO 11 - **Project Management and Finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these ta one's own work, member and leader in a team, to manage multidisciplinary environment.

- PO 12 - **Life-Long Learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-tong learning in the broadest context of technological change.

Index

Lab-1

Experiment 1

Aim : Write a Java program to print the Fibonacci list upto a given number

Concepts Used : Scanner Class To Take Input
While Loop For Finding The Numbers

Algorithm

Steps :

Start

1. Create Class Fibonacci with function fibo
2. Instantiate obj1 object
3. Input the limit
4. Call fibo() with object reference limit as parameter
5. fibo()

6. Declare variable n1,n2
7. n1=0,n2=1
8. while(n1<limit) goto step 9 Else To 12
9. Print n1
10. n1=n1+n2
11. n2=n2-n1
12. Endwhile

Stop

CODE

```
import java.util.*;

class Fibonacci {

    public void fibo(int num) {

        System.out.println("\n The Fibnacci Series Upto " + num + " Is
:\n");

        int n1 = 0, n2 = 1;

        while (n1 < num) {

            System.out.println(n1);

            n2 = n1 + n2;
```

```
            n1 = n2 - n1;

        }

    }

    public static void main(String args[]) {

        Fibonacci obj1 = new Fibonacci();

        System.out.println("Enter The Limit : ");
        Scanner sc = new Scanner(System.in);

        obj1.fibo(sc.nextInt());

    }

}
```

Output

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-1\Fibonacci>javac Fibonacci.java

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-1\Fibonacci>java Fibonacci
Enter The Limit :
10

The Fibnacci Series Upto 10 Is :

0
1
1
2
3
5
8

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-1\Fibonacci>java Fibonacci
Enter The Limit :
30

The Fibnacci Series Upto 30 Is :

0
1
1
2
3
5
8
13
21

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-1\Fibonacci>java Fibonacci
Enter The Limit :
0

The Fibnacci Series Upto 0 Is :

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-1\Fibonacci>java Fibonacci
Enter The Limit :

-13

The Fibnacci Series Upto -13 Is :

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-1\Fibonacci>java Fibonacci
Enter The Limit :
50

The Fibnacci Series Upto 50 Is :

0
1
1
2
3
5
8
13
21
34

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-1\Fibonacci>

Result

Fibonacci Numbers Upto Integer N Is Found And Printed

Experiment 2

Aim : Write a Java program to display the transpose of a given matrix

Concepts Used : Scanner Class To Take Input (.nextInt())
For Loop For Iterating Through The Matrix

Algorithm

Steps :

Start

int row=3, column=3
int matrix[][] = {{1,2,3},{4,5,6},{7,8,9}}
int transpose[column][row]

for i=0 till row do
for j=0 till column do
transpose[j][i] = matrix[i][j]
end for
end for
for j=0 till column do
for i=0 till row do
print transpose[j][i]
end for
end for

Stop

CODE

```
import java.util.*;

public class Transpose {

    public static void main(String args[]) {
        int row, col;

        System.out.println("Enter The Number Of Column : ");
        Scanner sc = new Scanner(System.in);

        col = sc.nextInt();

        System.out.println("Enter The Number Of Rows : ");
        row = sc.nextInt();
```

```
        System.out.println("Enter The Matrix : ");

        int matrix[][] = new int[row][col];

        for (int i = 0; i < row; i++) {
            for (int j = 0; j < col; j++) {

                matrix[i][j] = sc.nextInt();

            }
            System.out.println();
        }

        System.out.println("The Matrix Is : ");

        for (int i = 0; i < row; i++) {
            for (int j = 0; j < col; j++) {

                System.out.print(matrix[i][j] + " ");

            }
            System.out.println();
        }

        System.out.println("Transpose Of The Matrix Is : ");

        int matrixt[][] = new int[col][row];

        for (int i = 0; i < col; i++) {
            for (int j = 0; j < row; j++) {

                matrixt[i][j] = matrix[j][i];
                System.out.print(matrixt[i][j] + " ");

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

```
            }

        }

}
```

Output

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-1\Transpose>javac Transpose.java

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-1\Transpose>java Transpose
Enter The Number Of Column :
4
Enter The Number Of Rows :
4
Enter The Matrix :
1
2
3
4

5
5
6
7

8

8
9
0

1
1
23

3

The Matrix Is :
1 2 3 4
5 5 6 7
8 8 9 0
1 1 23 3
Transpose Of The Matrix Is :
1 5 8 1
2 5 8 1
3 6 9 23
4 7 0 3

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-1\Transpose>java Transpose
Enter The Number Of Column :
2
Enter The Number Of Rows :
2
Enter The Matrix :
1
2

3
4

```
The Matrix Is :
1 2
3 4
Transpose Of The Matrix Is :
1 3
2 4

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-1\Transpose>java Transpose
Enter The Number Of Column :
5
Enter The Number Of Rows :
5
Enter The Matrix :
3
2
6
3
7

5
7
5
4
7

6
4
3
5
7

5
7
```

6
4
3

3
7
8
7
6

The Matrix Is :
3 2 6 3 7
5 7 5 4 7
6 4 3 5 7
5 7 6 4 3
3 7 8 7 6
Transpose Of The Matrix Is :
3 5 6 5 3
2 7 4 7 7
6 5 3 6 8
3 4 5 4 7
7 7 7 3 6

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-1\Transpose>

Result

Transpose Of The Given Matrix Is Found By Interchanging Column And Rows

Experiment 3

Aim : Write a Java Program to find the frequency of a given character in a given string

Concepts Used : Scanner Class To Take Input (.nextLine())
For Loop For Iterating Through Character Array
.toCharArray() to Convert String To Character Array
.charAt() To Find The Character At Specific Position Of String
.toLowerCase() To Ignore Case While Comparing

Algorithm

Steps :

Start

```
String str="hello world !"
 char ch='l'
 int count=0
  for i=0 till string_length(str) do
 if str[i] == ch then
  Count++
  end if
end for
print coun
```

Stop

CODE

```
import java.util.*;

public class Frequency {

    public static void main(String args[]) {
        int count = 0;
        System.out.println("Enter The String :");

        Scanner sc = new Scanner(System.in);
        String s = sc.nextLine();

        System.out.println("Enter The Character To Find Frequency :");

        String ch = sc.nextLine();

        for (char c : s.toCharArray()) {
            if (Character.toLowerCase(c) ==
Character.toLowerCase(ch.charAt(0)))
                count++;
        }

        System.out.println("Frequency : " + count);

    }

}
```

Output

```
C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-1\Frequency>javac Frequency.java

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-1\Frequency>java Frequency
Enter The String :
diidggsidishh
Enter The Character To Find Frequency :
D
Frequency : 3

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-1\Frequency>java Frequency
Enter The String :
dddjjahaghajha
Enter The Character To Find Frequency :
K
Frequency : 0

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-1\Frequency>java Frequency
Enter The String :
AHHBHABHA AJJAHHAN AJBJAB
Enter The Character To Find Frequency :
a
Frequency : 8
```

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-1\Frequency>

Result

frequency of a given character in a given string Is Found By Traversing Through The String By Converting It To Character Array

Lab-2

Experiment 4

Aim : Design a class to represent a bank account. Include the following members.

Data Members: Name of the depositor
Account Number
Type of Account
Balance amount in the account

Methods : To deposit an amount
To withdraw an amount after checking balance
To display the name and balance

Incorporate default and parameterized constructor to provide initial values

Concepts Used : Constructor chaining.

Algorithm

Steps :

Start

Algorithm deposit(x)

1.accBalance = accBalance + x

Algorithm withdraw(x)

1.accBalance = accBalance –x

Algorithm display()

1.Print accName

2.Print accBalance

3.Print accType

4.Print accNo

Algorithm Bank() //default constructor

1.accName = NIL

2.accBalance = 0

3.accType = NIL

4.accNo = 0

Algorithm Bank(a, b, c, d) //parameterized constructor

1.accName = a

2.accBalance = b

3.accType = c

4.accNo = d

Stop

CODE

```
public class BankAccount {

    private String accHolder;
    private String accNumber;
    private String accType;
    private Double accBalance;
```

```
BankAccount() {
    System.out.println("Welcome To Dinoy Bank");
    accinfo();
}

BankAccount(String name, String acc, String type, Double balance) {
    System.out.println("Welcome To Dinoy Bank");

    accHolder = name;
    accNumber = acc;
    accType = type;
    accBalance = balance;
    accinfo();
}

public void accinfo() {
    System.out.println("\n\n||||| Dino Bank Welcome You |||||");
    System.out.println("\n:: Account Holder Name : " + accHolder);
    System.out.println(":: Account Number : " + accNumber);
    System.out.println(":: Account Type : " + accType);
    System.out.println(":: Account Balance : " + accBalance);
    System.out.println(":: Happy Banking :) \n");
}

public void deposit(Double depo) {
    accBalance += depo;
    System.out.println("New Balance " + accBalance);

}

public void withdraw(Double draw) {

    if (draw > accBalance) {
        System.out.println("Cannot Withdraw Out Of Balance ");
        accinfo();
    } else {
```

```
            accBalance -= draw;
            accinfo();
        }

    }

    public static void main(String args[]) {

        // with default values Bankaccount//
        BankAccount bk = new BankAccount();

        // using
        // parameterized//
        BankAccount acc1 = new BankAccount("DINOY RAJ", "AC101432",
"SAVINGS", 100.0);
        acc1.deposit(20000.0);
        acc1.withdraw(1000.0);
        // low Balance
        BankAccount acc2 = new BankAccount("AMAL NATH", "AC101438",
"SAVINGS", 100.0);
        acc2.deposit(2000.0);
        acc2.withdraw(100000.0);

    }

}
```

Output

```
C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-2>javac BankAccount.java

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-2>java BankAccount
Welcome To Dinoy Bank


||||| Dino Bank Welcome You |||||

:: Account Holder Name : null
:: Account Number : null
:: Account Type : null
:: Account Balance : null
:: Happy Banking :)

Welcome To Dinoy Bank


||||| Dino Bank Welcome You |||||

:: Account Holder Name : DINOY RAJ
:: Account Number : AC101432
:: Account Type : SAVINGS
:: Account Balance : 100.0
:: Happy Banking :)

New Balance 20100.0


||||| Dino Bank Welcome You |||||

:: Account Holder Name : DINOY RAJ
:: Account Number : AC101432
```

:: Account Type : SAVINGS
:: Account Balance : 19100.0
:: Happy Banking :)

Welcome To Dinoy Bank

||||| Dino Bank Welcome You |||||

:: Account Holder Name : AMAL NATH
:: Account Number : AC101438
:: Account Type : SAVINGS
:: Account Balance : 100.0
:: Happy Banking :)

New Balance 2100.0
Cannot Withdraw Out Of Balance

||||| Dino Bank Welcome You |||||

:: Account Holder Name : AMAL NATH
:: Account Number : AC101438
:: Account Type : SAVINGS
:: Account Balance : 2100.0
:: Happy Banking :)

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-2>

Result

Two bank account objects with the given fields and methods are created using default and parameterized constructors.

Lab -2

Experiment 5

Aim : Write a Java program which creates a class named 'Employee' having the followingmembers: Name, Age, Phone number, Address, Salary. It also has a method named'printSalary( )' which prints the salary of the Employee. Two classes 'Officer' and'Manager' inherits the 'Employee' class. The 'Officer' and 'Manager' classes have datamembers 'specialization' and 'department' respectively. Now, assign name, age, phonenumber, address and salary to an officer and a manager by making an object of bothof these classes and print the same.

Concepts Used : Inhertitance, Method overloading, this keyword.

Algorithm

Steps :

Start

Class Employee

1. Declare fields name, age, phone, address, salary
2. Define printSalary and printDetails methods

Class Officer inherits Class Employee

1. Declare additional field specialization
2. Define constructors for with and without specialization
3. Define printSpl method

Class Manager inherits Class Employee

1. Declare additional field department
2. Define constructors for with and without department
3. Define printDept method

Stop

CODE

```
public class Employee {

    String name;
    int age;
    int phno;
    Double salary;
    String address;

    public void printSalary() {
        System.out.println("Salary: " + salary);
    }

    public void printInfo() {
```

```
        System.out.println("Name: " + name + "\nage: " + age + "\nphone
number: " + phno + "\nSalary: " + salary
                + "\nAddress: " + address);
    }

    public static void main(String args[]) {

        System.out.println("::Dino Company Employee LIst ::\n");
        Officer officer1 = new Officer("Amal nath", 18, 852634464,
200003.1, "Mangootil House Kannur", "HR");
        officer1.printSalary();
        officer1.printSpec();
        Manager manager1 = new Manager("Dinoy Raj ", 19, 730618539,
100011.1, "Rennugeetham House Pokkunnu", "Design");
        manager1.printSalary();
        manager1.printdep();

    }
}

class Officer extends Employee {

    String spec;

    Officer(String name, int age, int phno, Double salary, String address)
{

        this.name = name;
        this.age = age;
        this.phno = phno;
        this.salary = salary;
        this.address = address;

        this.printInfo();

    }
```

```
    Officer(String name, int age, int phno, Double salary, String address,
String spec) {

        this.name = name;
        this.age = age;
        this.phno = phno;
        this.salary = salary;
        this.address = address;
        this.spec = spec;

        this.printInfo();
        this.printSpec();

    }

    public void printSpec() {
        System.out.println("Specialization: " + spec);
    }
}

class Manager extends Employee {

    String dep;

    Manager(String name, int age, int phno, Double salary, String address)
{

        this.name = name;
        this.age = age;
        this.phno = phno;
        this.salary = salary;
        this.address = address;

        this.printInfo();

    }
```

```
    Manager(String name, int age, int phno, Double salary, String address,
String dep) {

        this.name = name;
        this.age = age;
        this.phno = phno;
        this.salary = salary;
        this.address = address;
        this.dep = dep;

        this.printInfo();
        this.printdep();

    }

    public void printdep() {
        System.out.println("Specialization: " + dep);
    }
}
```

Output

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-2\salary>javac Employee.java

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-2\salary>java Employee
::Dino Company Employee LIst ::

Name: Amal nath
age: 18
phone number: 852634464
Salary: 200003.1
Address: Mangootil House Kannur
Specialization: HR
Salary: 200003.1
Specialization: HR
Name: Dinoy Raj
age: 19
phone number: 730618539
Salary: 100011.1
Address: Rennugeetham House Pokkunnu
Specialization: Design
Salary: 100011.1
Specialization: Design

Result

Objects of both Officer and Manager classes are created and their details are printed.

Lab -3

Experiment 6

Aim : Write two Java classes Employee and Engineer. Engineershould inherit from Employeeclass. Employee class to havetwo methods display() and calcSalary(). Write a programtodisplay the engineer salary and to display from Employeeclass using a single objectinstantiation (i.e., only one objectcreation is allowed). display() only prints the name of●theclass and does not return any value. Ex. " Name of class isEmployee." calcSalary() in●Employee displays "Salary ofemployee is 10000" and calcSalary() in Engineerdisplays"Salary of employee is 20000." ?

Concepts Used : Inheritance, static keyword

Algorithm

Steps :

Start

Class Employee

1.Declare fields Salary and Classname

2.Define display method to print className

3.Define calcSalary method to print salaryClass

Engineer inherits Class Employee

1.Call methods display and calcSalary

2.Create Engineer object and call methods display and calcSalary on the object

Stop

CODE

```
class Employee {

    String Classname;
    Double Salary = 1000.0;

    void calcSalary(Double Salary) {

        System.out.println("Salary Of The Employee Is: " + Salary);

    }
```

```
    void display(String Classname) {
        System.out.println("Name Of The Class Is: " + Classname);
    }
}

public class Engineer extends Employee {

    void calcSalary(Double Salary) {

        System.out.println("Salary Of The Employee Is: " + Salary);

    }

    void calcSalary() {

        super.calcSalary(Salary);


    }

    public static void main(String args[]) {

        Engineer eg = new Engineer();
        eg.display("Employee");
        eg.calcSalary();
        eg.calcSalary(2000.0);

    }

}
```

Output

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-3\Engineer>javac Engineer.java

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-3\Engineer>java Engineer
Name Of The Class Is: Employee
Salary Of The Employee Is: 1000.0
Salary Of The Employee Is: 2000.0

Result

methods of parent class Employee are accessed with super keyword

Lab -3

Experiment 7

Aim : Write a java program to create an abstract class named Shapethat contains an emptymethod named numberOfSides( ).Provide three classes named Rectangle, TriangleandHexagon such that each one of the classes extends the classShape. Each one of theclasses contains only the methodnumberOfSides( ) that shows the number of sides in thegivengeometrical structures

Concepts Used : Inheritance, Method overriding, Data abstraction using abstract keyword

Algorithm

Steps :

Start

Declare abstract method numOfSidesClass
Triangle inherits Class Shape
1.Override numOfSides method and Print 3
Class Rectangle inherits Class Shape
1.Override numOfSides method and print 4
Class Hexagon inherits Class Shape
1.Override numOfSides method and print 6

Stop

CODE

```
abstract class Shape {

    abstract void numberOfSides();
}

class Triangle extends Shape {

    void numberOfSides() {

        System.out.println("Triangle : 3");
    }
}

class Rectangle extends Shape {

    void numberOfSides() {

        System.out.println("Rectangle : 4");
    }
}

class Hexagon extends Shape {

    void numberOfSides() {
```

```
        System.out.println("Hexagon : 6");
    }
}

public class polygon {

    public static void main(String args[]) {
        Triangle T1 = new Triangle();
        Rectangle R1 = new Rectangle();
        Hexagon H1 = new Hexagon();

        T1.numberOfSides();
        R1.numberOfSides();
        H1.numberOfSides();

    }

}
```

Output

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-3\Shapes>javac polygon.java

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-3\Shapes>java polygon
Triangle : 3
Rectangle : 4
Hexagon : 6

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-3\Shapes>

Result

Data abstraction and method overriding are performed on different shapes.

Lab -4

Experiment 8

Aim : Create a package ‘studpack’ which incorporates Student class and Sports interface. Create a Result class that extends Student class and implements a sports interface to display the total marks. The details of the classes and interfaces described below. Use appropriate access specifier as per the requirement. (*method, - variable) Create a java program Hybrid.java that imports Result class from studpack and display the total for 5 students.

Concepts Used : Package, Interface

Algorithm

Steps :

Start

Package studpack

1.Interface Sports with public static final field grade and public abstract method displayGrade

2.Class Student with fields name, rollNo, mark1, mark2, mark3 and required constructors

3.Class Result inherits Class Student and implements Interface Sports with field total, required constructors, definition for overriden method displayGrade, and displayTotal methodClass

Hybrid

1.Import package studpack

2.Create Result object and call displayTotal method on i

Stop

CODE

```
package studpack;

interface Sports
{
    int grade = 100;

    void displayGrade();
}

class Student
{
    String name;
    int rollNo;
    float mark1, mark2, mark3;

    Student()
    {
        name = "NONE";
        rollNo = 0;
        mark1 = mark1 = mark3 = 0;
    }
```

```
    Student(String name, int rollNo, float mark1, float mark2, float
mark3)
    {
        this.name = name;
        this.rollNo = rollNo;
        this.mark1 = mark1;
        this.mark2 = mark2;
        this.mark3 = mark3;
    }
}

public class Result extends Student implements Sports
{
    float total;

    public void displayGrade()
    {
        System.out.println("Sports Grade = "+grade);
    }

    public void displayTotal()
    {
        System.out.println("Roll No = "+rollNo);
        displayGrade();
        System.out.println("Total obtained = "+total+"/400\n");
    }

    public Result()
    {
        super();
        total = 0;
    }

    public Result(String name, int rollNo, float mark1, float mark2, float
mark3)
    {
        super(name, rollNo, mark1, mark2, mark3);
```

```
        total = mark1 + mark2 + mark3 + grade;
    }
}
```

```
import studpack.*;

class Hybrid
{
    public static void main(String arg[])
    {
        Result st1 = new Result("Abhiram", 04, 88.75f, 91, 95);
        Result st2 = new Result("Ajith", 07, 89, 90.5f, 96);
        Result st3 = new Result("Emil", 22, 90, 92.5f, 99);
        Result st4 = new Result("Dinoy", 21, 92.5f, 89, 98);
        Result st5 = new Result("Jijo", 29, 90, 90.5f, 97.5f);

        st1.displayTotal();
        st2.displayTotal();
        st3.displayTotal();
        st4.displayTotal();
        st5.displayTotal();
    }
}
```

Output

Roll No = 4

Sports Grade = 100

Total obtained = 374.75/400

Roll No = 7

Sports Grade = 100

Total obtained = 375.5/400

Roll No = 22

Sports Grade = 100

Total obtained = 381.5/400

Roll No = 21

Sports Grade = 100

Total obtained = 379.5/400

Roll No = 29

Sports Grade = 100

Total obtained = 378.0/400

Result

Package studpack is imported to another package and its classes are used there

Lab -5

Experiment 9

Aim : Write a Java program that shows the usage of try, catch, throws and finally.

(a)---Try-Finally example

(b)---Multiple catch example (3 catches for a single try)

(c)---Nested Try (3 levels of nesting)

(d)---Throw an exception when there are no sufficient arguments passed into command line as input for adding two numbers.

(e)---Throws example (for handling two exceptions in a method)

Concepts Used : Exception handling, parseInt() method,charAt() method

Algorithm

Steps :

Start

Algorithm tryFinally(index)

1.try block –operation a/b

2.catch block –Print ArithmeticException object
3.finally block –Print a Message

Algorithm multipleCatch(index)

1.try block – Conversion String to int , division of two after conversion,charcter extraction,file input
2.catch block
 1 –Print NumberFormatException object
3.catch block
 1 –Print ArithmeticException object
4.catch block
 1 –Print ArrayOutOfBoundException object
4.catch block
 1 –Print FileNotFoundException object

Algorithm nestedTry(str, index)

1.try –
2.args[1].charAt(3)
3.try –
4.obj = null
5.Print obj.dinoy()
6.catch –Print NullPointerException object
7.catch –Print IndexOutOfBoundsException object

objectAlgorithm throwEg(input[])

1.if(input.length != 1)
2.throw RuntimeException("Input One Number Through Command Line")
3.else
4. Try Convert Integer.parseInt(input[0])
5.catch NumberFormatException object

Algorithm throwsEg

1.throws FileNotFoundException, Arithmetic Exception

2.Read any file

Stop

CODE

Try Finally - 1

```
import java.util.*;

public class TryFinallyEx {

    public static void main(String args[]) {

        System.out.println("Enter The Value Of A And B : ");
        Scanner sc = new Scanner(System.in);

        int a = sc.nextInt();
        int b = sc.nextInt();

        try {

            int c = a / b;
            System.out.println(c);

        } catch (ArithmeticException e) {
```

```
            System.out.println("Exception Found : " + e);
        } finally {
            System.out.println("This One Will Excecute");
        }

    }

}
```

Try Finally - 2

```
import java.util.*;

public class TryFinallyEx2 {

    public static void main(String args[]) {
        int arr[] = new int[4];
        System.out.println("Enter The 4 Numbers :");
        Scanner sc = new Scanner(System.in);

        int i = 0;
        while (i < 4) {
            arr[i] = sc.nextInt();
            i++;
        }

        try {
            System.out.println("Enter Index To Acess : ");
            System.out.println(arr[sc.nextInt() - 1]);
        } catch (ArrayIndexOutOfBoundsException e) {

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

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

        } finally {

            System.out.println("\nThis One Will Excecute");
            i = 0;
            while (i < 4) {
                System.out.println(arr[i]);
                i++;
            }

        }

    }

}
```

Multiple Catch

```
import java.io.*;

public class MultiCatch1 {

    public static void main(String args[]) {

        // String s="dinoy";
        try {

            int c = Integer.parseInt(args[0]);
            c = Integer.parseInt(args[1]) / Integer.parseInt(args[2]);
            char g = args[3].charAt(10);
            FileInputStream st = new FileInputStream("input.txt");
```

```
        } catch (NumberFormatException e) {
            System.out.println(e.getMessage());
        } catch (ArithmeticException e) {
            System.out.println(e.getMessage());
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println(e.getMessage());
        } catch (FileNotFoundException e) {
            System.out.println(e.getMessage());
        }

    }

}
```

Nested Try

```
import java.io.*;

public class NestedTry1 {

    public int dinoy() {
        return 1;
    }

    public static void main(String args[]) {
        try {
            char c = args[1].charAt(3);
            try {

                NestedTry1 nt = new NestedTry1();
                nt = null;
```

```
                System.out.println(nt.dinoy());

            } catch (NullPointerException e) {

                System.out.println("inner Exception : " + e.getMessage());

            }
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("outer Exception : " + e.getMessage());
        }
    }

}
```

Throw Example

```
public class ThrowEx {

    public static void main(String[] args) {
        if (args.length != 1) {
            throw new RuntimeException("Input One Number Through Command
Line");
        }

        try {

            int d = Integer.parseInt(args[0]);

        } catch (NumberFormatException e) {
```

```
            System.out.println("Exception :" + e.getMessage());
        }
    }
}
```

Throws Example

```
import java.io.*;

public class Throws {

    public static void main(String args[]) throws FileNotFoundException,
ArithmeticException {

        FileInputStream st = new FileInputStream("input.txt");
        int a = 1 / 0;

    }

}
```

Output

1. Try Finally - 1

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-5\TryFinally>javac TryFinallyEx.java

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-5\TryFinally>java TryFinallyEx

Enter The Value Of A And B :

1

0

Exception Found : java.lang.ArithmeticException: / by zero

This One Will Excecute

2. Try Finally - 2

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-5\TryFinally>javac TryFinallyEx2.java

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-5\TryFinally>java TryFinallyEx2

Enter The 4 Numbers :

1

2

3

4

Enter Index To Acess :

7

Index 6 out of bounds for length 4

[Ljava.lang.StackTraceElement;@2f4d3709

This One Will Excecute

1

2

3

4

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-5\TryFinally>

3.Multiple Catch

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-5\MultipleCatch>javac MultiCatch1.java

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-5\MultipleCatch>java MultiCatch1

Index 0 out of bounds for length 0

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-5\MultipleCatch>javac MultiCatch1.java

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-5\MultipleCatch>java MultiCatch1 w

For input string: "w"

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-5\MultipleCatch>java MultiCatch1 1 2 0

/ by zero

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-5\MultipleCatch>java MultiCatch1 1 2 5 df

Exception in thread "main" java.lang.StringIndexOutOfBoundsException: String index out of range: 10

    at java.base/java.lang.StringLatin1.charAt(StringLatin1.java:48)

    at java.base/java.lang.String.charAt(String.java:712)

    at MultiCatch1.main(MultiCatch1.java:12)

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-5\MultipleCatch>java MultiCatch1 1 2 5 dffgfgfgfgfgfgfgffgfgfgfgf

input.txt (The system cannot find the file specified)

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-5\MultipleCatch>

4. Nested Try

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-5\NestedTry>javac NestedTry1.java

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-5\NestedTry>java NestedTry1

outer Exception : Index 1 out of bounds for length 0

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-5\NestedTry>java NestedTry1 adadassd sdadada sdsds

inner Exception : Cannot invoke "NestedTry1.dinoy()" because "<local2>" is null

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-5\NestedTry>

5. Throw Example

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-5\Throw>javac ThrowEx.java

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-5\Throw>java ThrowEx

Exception in thread "main" java.lang.RuntimeException: Input One Number Through Command Line

    at ThrowEx.main(ThrowEx.java:6)

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-5\Throw>java ThrowEx 1

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-5\Throw>java ThrowEx d

Exception :For input string: "d"

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-5\Throw

5. Throws Example

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-5\Throws>javac Throws.java

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-5\Throws>java Throws

Exception in thread "main" java.io.FileNotFoundException: input.txt (The system cannot find the file specified)

    at java.base/java.io.FileInputStream.open0(Native Method)

    at java.base/java.io.FileInputStream.open(FileInputStream.java:211)

```
        at java.base/java.io.FileInputStream.<init>(FileInputStream.java:153)
        at java.base/java.io.FileInputStream.<init>(FileInputStream.java:108)
        at Throws.main(Throws.java:8)

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-5\Throws>
```

Result

Exception handling is carried out in a Java program.

Lab -5

Experiment 10

Aim : Write a Java program that read from a file and write to file by handling all file related exceptions.

Concepts Used : File Input/Output classes,java.io package

Algorithm

Steps :

Start

1. import java.io classes
2.throws IOException
3.Initialise FileInputStream object fin on the input file
4.Initialise FileOutputStream object fout on the output file
5.try
6.while((i = fin.read()) != -1)
7.fout.write(i)
8.endwhile
9.catch
10.Print IOException object
11.endtry

Stop

CODE

```
import java.io.*;

public class FileHandling {

    public static void main(String args[]) throws IOException {

        try {
            FileInputStream fi = new FileInputStream("input.txt");
            FileOutputStream ft = new FileOutputStream("output.txt");

            int i = 0;

            while ((i = fi.read()) != -1) {
                ft.write(i);
            }
        } catch (IOException e) {
            System.out.println("Exception " + e);
        }

    }

}
```

Output

//output.txt at same folder

hello dinoy

Result

File handling is carried out in a Java program

Lab -5

Experiment 11

Aim : Write a Java program that reads a line of integers, and then displays each integer, and the sum of all the integers (Use String Tokenizer class of java.util)

Concepts Used : Scanner class, StringTokenizer clas

Algorithm

Steps :

Start

Algorithm StringTokens

1.import java.util.Scanner
2.sum = 0
3.Initialise Scanner object in on System.in
4.s = in.nextLine() //read the numbers separated by a space
5.Initialise StringTokenizer object str on s
6.while(str.hasMoreTokens())
7.num = Integer.parseInt(str.nextToken())
8.Print num
9.sum += num
10.endwhile
11.Print sum

Stop

CODE

```
import java.util.*;

public class StringToToken {

    public static void main(String[] args) {
        System.out.println("Enter The Line Of Integer :");
        Scanner sc = new Scanner(System.in);
        String s = sc.nextLine();

        StringTokenizer st = new StringTokenizer(s);

        int num, sum = 0;
        System.out.println("Digits : ");
        while (st.hasMoreTokens()) {
            num = Integer.parseInt(st.nextToken());
            System.out.println(num);
            sum += num;

        }
        System.out.println("Sum :" + sum);
    }

}
```

Output

```
C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-6\StringTokenizer>javac
StringToToken.java

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-6\StringTokenizer>java StringToToken
Enter The Line Of Integer :
12 33 4 6 76 8 87 6 65 5
Digits :
12
33
4
6
76
8
87
6
65
5
Sum :302
```

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-6\StringTokenizer>

Result

A line of integers are read from the user and they are printed along with their sum.

Lab - 7

Experiment 12

Aim : Write a Java program for the following: 1) Create a doubly linked list of elements. 2) Delete a given element from the above list. 3) Display the contents of the list after deletion

Concepts Used : Collections framework

Algorithm

Steps :

Start

1.import java.util classes
2.Initialise LinkedList<String> object list
3.Add members using list.add() method
4.Print list
5.Remove members using list.remove() method
6.Initialise Iterator object itr = list.iterator()
7.while(itr.hasNext()) //Displaying the list
8.Print itr.next()
9.endwhile

Stop

CODE

```
import java.util.*;

public class Dlinkedlist {

    public static void main(String args[]) {

        LinkedList<String> ll = new LinkedList<String>();

        /*
         * System.out.println("Menu \n1.Insert\n2.delete\n3.display");
         *
         * Scanner sc = new Scanner(System.in);
         *
         * switch (sc.nextInt()) {
         *
         * case 1: System.out.println("Enter The Element : ");
ll.add(sc.nextLine());
         *
         * case 2: System.out.println("Enter The Element To Delete : ");
         * ll.remove(sc.nextLine());
         *
         * }
         */
```

```
        ll.add("dinoy");
        ll.add("dhanya");
        ll.add("kuttan");
        ll.add("deepu");
        ll.add("sangi");

        System.out.println("List After Insertion : ");

        Iterator<String> itr = ll.iterator();

        while (itr.hasNext()) {
            System.out.println(itr.next());
        }

        ll.remove("dinoy");
        ll.remove(2);

        System.out.println("List After Removing : ");

        Iterator<String> rti = ll.iterator();

        while (rti.hasNext()) {
            System.out.println(rti.next());
        }

    }
}
```

Output

```
C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-7\DoublyLinkedList>javac Dlinkedlist.java

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-7\DoublyLinkedList>java Dlinkedlist
List After Insertion :
dinoy
dhanya
kuttan
deepu
sangi
List After Removing :
dhanya
kuttan
sangi

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-7\DoublyLinkedList>
```

Result

A doubly linked list is implemented using Collections framework

Lab - 7

Experiment 13

Aim : Write a Java program that implements the binary search algorithm.

Concepts Used : Scanner, Binary Search algorithm.

Algorithm

Steps :

Start

Algorithm main

1.import java.util.Scanner
2.Intilialise Scanner object in on System.in
3.Read array size, n = in.nextInt()
4.Initialise array arr of sizen
5.for i=0 till n //Read the elements
6.arr[i] = in.nextInt()
7.endfor
8.Read search element, key = in.nextInt()
9.binarySearch()

Algorithm binarySearch

1.first = 0, last = arr.length −1
2.while (first <= last)
3.mid = (first + last) / 2
4.if(key < arr[mid])
5.last = mid −1
6.else if (key > arr[mid])
7.first = mid + 1
8.else
9.Print “Element found”
10.return
11.endif
12.endwhile
13.Print “Element not found

Stop

CODE

```
import java.util.*;

public class BinarySearch {

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

```
        int num, i = 0;

        System.out.println("Enter The Number Of Digits: ");
        Scanner sc = new Scanner(System.in);

        num = sc.nextInt();

        System.out.println("Enter The Digits (Ascending order): ");

        int arr[] = new int[num];

        while (i < num) {

            arr[i] = sc.nextInt();
            i++;

        }
        System.out.println("Enter The Digits To Search: ");

        int sr = sc.nextInt();

        int l = 0, r = num - 1;
        int mid = 0;

        while (l <= r) {

            mid = (l + r) / 2;

            if (arr[mid] == sr) {
                break;
            } else if (arr[mid] > sr) {

                r = mid - 1;
                continue;

            } else {
                l = mid + 1;
```

```
                continue;
            }
        }

        if (l > r) {
            System.out.println("Elements Not Found ! ");
        } else if (arr[mid] == sr) {
            System.out.println(arr[mid] + " Found At Position " + mid);
        }

    }

}
```

Output

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-7\BinarySort>javac BinarySort.java

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-7\BinarySort>java BinarySort
Enter The Number Of Digits:
5
Enter The Digits (Ascending order):
1
2
3
4
5

```
Enter The Digits To Search:
4
4 Found At Position 3

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-7\BinarySort>java BinarySort
Enter The Number Of Digits:
5
Enter The Digits (Ascending order):
1
2
3
4
5
Enter The Digits To Search:
7
Elements Not Found !

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-7\BinarySort>
```

Result

Binary Search is performed on an array of integers

Lab - 8

Experiment 14

Aim : Write a Java program that implements a multi-threaded program which has three threads. First thread generates a random integer every 1 second. If the value is even, the second thread computes the square of the number and prints. If the value is odd the third thread will print the value of the cube of the number. .

Concepts Used : Multithreading, Random generator, Math.pow() method

Algorithm

Steps :

Start

Class X implements Class Runnable

1.importjava.util.Random

2.Initialise Random object rand and static field random

3.Override public void run() method

4.random = rand.nextInt(25)

5.Print random

6.End run()

Class Y implements Class Runnable

1.Override public void run() method

2.if(X.random % 2 == 0)
3.Print Math.pow(X.random, 2)
4.endif
5.End run()

Class Z implements Class Runnable

6.Override public void run() method
7.if(X.random % 2 != 0)
8.Print Math.pow(X.random, 3)
9.endif
10.End run()

Class Main
1.for i=0 till 10
2.Thread x = new Thread(new X())
3.Thread y = new Thread(new Y())
4.Thread z = new Thread(new Z())
5.x.sleep(1000) //Inside try-catch block
6.x.start()
7.y.start()
8.z.start()
9.endfo

Stop

CODE

```
import java.util.Random;

class X implements Runnable {
    static int random;
    Random rand = new Random();

    public void run() {
        random = rand.nextInt(25);
        System.out.println(random);
    }
}

class Y implements Runnable {
    public void run() {
        if (X.random % 2 == 0)
            System.out.println((int) Math.pow(X.random, 2) + "\n");
    }
}

class Z implements Runnable {
    public void run() {
        if (X.random % 2 != 0)
            System.out.println((int) Math.pow(X.random, 3) + "\n");
    }
}

class Multhread {
    public static void main(String args[]) {
        for (int i = 0; i < 10; i++) {
            Thread objX = new Thread(new X());
            Thread objY = new Thread(new Y());
            Thread objZ = new Thread(new Z());
            try {
                Thread.sleep(1000);
            } catch (Exception e) {
            }
```

```
            objX.start();
            objY.start();
            objZ.start();
        }
    }
}
```

Output

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-8\Thread1>javac Multithread.java

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-8\Thread1>java Multithread
16
256

13
2197

7
343

7
343

18
324

```
10
100

5
125

10
100

6
36

2
4


C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-8\Thread1>
```

Result

MultiThreading Program Implimented In Java

Lab - 8

Experiment 15

Aim : Write a Java program that shows thread synchronization.

Concepts Used : Multithreading, Thread synchronization.

Algorithm

Steps :

Start

Class Demo
1.synchronized display(n, t)
1.for i = n to t2.Print i * 5
3.Thread.sleep(400)
4.enfor
5.End display()

Class X inherits Class Thread

1.Declare field obj (object of class Demo) and define a constructor to initialise obj
2.public void run()
3.obj.display(1, 5)
4.End run()

Class Y inherits Class Thread

5.Declare field obj (object of class Demo) and define a constructor to initialise obj
6.public void run()
7.obj.display(6, 10)
8.End run()

Class Main

1.Initialise Demo obj
2.X x = new X(obj)
3.Y y = new Y(obj)
4.x.start()
5.y.start( )

Stop

CODE

```
class Demo {
    public synchronized void display(int n, int t) {
        for (int i = n; i <= t; i++) {
            System.out.println(i * 5);
            try {
                Thread.sleep(400);
            } catch (Exception e) {
```

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

class X extends Thread {
    Demo obj;

    X(Demo obj) {
        this.obj = obj;
    }

    public void run() {
        obj.display(1, 5);
    }
}

class Y extends Thread {
    Demo obj;

    Y(Demo obj) {
        this.obj = obj;
    }

    public void run() {
        obj.display(6, 10);
    }
}

class MultithreadSyn {
    public static void main(String args[]) {
        Demo obj = new Demo();
        X objX = new X(obj);
        Y objY = new Y(obj);
        objX.start();
        objY.start();
```

```
    }
}
```

Output

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-8\Thread2>javac MultithreadSyn.java

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-8\Thread2>java MultithreadSyn
5
10
15
20
25
30
35
40
45
50

C:\Users\dinoy\OneDrive\Desktop\javalabfinal\Lab-8\Thread2

Result

Thread synchronization is carried out in a Java program

Lab - 9

Experiment 16

Aim : Write a Java program that works as a simple calculator. Arrange Buttons for digits and the + - * % operations properly. Add a text field to display the result. Handle any possible exceptions like divide by zero. Use Java Swing.

Concepts Used : Java Gui Creation And Event Handling

Algorithm

Steps :

Start

Algorithm Calculator

1.Import java.awt, java.awt.event, java.swing and java.util classes

2.Initialise JFrame frame with title “Simple Calculator”

3.Initialise JPanel panel1, panel2, inputPanel

4.Set layout managers GridLayout(4,3), GridLayout(6,1), GridBagLayout(), GridBagLayout() for panel, panel1, inputPanel and frame respectively

5.frame.setBounds(825,350,400)

6.frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE)//COMPONENTS

7.JTextField result = new JTextField()

8.JButton one = new JButton("1")

9.JButton two = new JButton("2")

10.JButton three = new JButton("3")

11.JButton four = new JButton("4")

12.JButton five = new JButton("5")

13.JButton six = new JButton("6")

14.JButton seven = newJButton("7")

15.JButton eight = new JButton("8")

16.JButton zero = new JButton("0")

17.JButton nine = new JButton("9")

18.JButton AC = new JButton("AC")

19.JButton dot = new JButton(".")

20.JButton plus = new JButton("+")

21.JButton minus = new JButton("-")

22.JButton into = new JButton("*")

23.JButton by = new JButton("/")

24.JButton mod = new JButton("%")

25.JButton equal = new JButton("=")

26.Add buttons from one to dotto panel

27.Add buttons from plus to equal to panel1

28.result.setHorizontalAlignment(JTextField.CENTER)

29.Modify UI using GridBagConstraints object c

30.frame.add(result, c)

31.Modify UI using c

32.inputPanel.add(panel, c)

33.Modify UI using c

34.inputPanel.add(panel1, c)

35.Modify UI using c

36.frame.add(inputPanel,c)

37.frame.setVisible(true)

38.panel.setVisible(true)

39.panel1.setVisible(true)

40.inputPanel.setVisible(true)//EVENT HANDLING//For the input buttons, the event handler is

41.result.setText(result.getText() + "1") //Button one//For All Cancel button, the event handler is,

42.result.setText("")//For Equals button, the event handler is,

43.exp = result.getText()

44.i = 1

45.res = 0

46.z = exp.charAt(1)

47.try

48.while(z != '+' && z != '-' && z != '*' && z != '/' && z != '%')

49.i++

50.z = exp.charAt(i)

51.endwhile

52.x = Float.parseFloat(exp.substring(0,i))

53.y = Float.parseFloat(exp.substring(i+1,exp.length()))

54.switch(z)

55.case + :

56.res = x+y

57.break

58.case -:

59.res = x-y

60.break

61.case * :

62.res = x*y

63.break

64.case / :

65.if(y == 0)

66.break

67.res = x/y

68.break

69.case % :

70.res = x%y

71.break

72.endcase

73.endswitch

74.if(y == 0)

75.result.setText("Not defined!")

76.else

77.result.setText(res+"")

78.endif

79.catch

80.result.setText("Enter two operands!")

81.endtry

82.STOP

Stop

CODE

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class calculator {

  calculator() {

    JFrame jrm = new JFrame();
    jrm.setSize(500, 500);

    JLabel jbl = new JLabel("Calculator");
    jbl.setBounds(227, 10, 400, 30);
    jrm.add(jbl);

    JTextField jtf = new JTextField();
    jtf.setEditable(false);
    jtf.setBounds(50, 50, 400, 30);
    jrm.add(jtf);

    JButton b1, b2, b3, b4, b5, b6, b7, b8, b9, b0, bdiv, bmul, bsub,
badd, bdec, beq, bdel, bclr, clear, dot;

    b1 = new JButton("1");
    b2 = new JButton("2");
    b3 = new JButton("3");
    b4 = new JButton("4");
    b5 = new JButton("5");
    b6 = new JButton("6");
    b7 = new JButton("7");
    b8 = new JButton("8");
    b9 = new JButton("9");
```

```
    b0 = new JButton("0");
    bdiv = new JButton("/");
    bmul = new JButton("*");
    bsub = new JButton("-");
    badd = new JButton("+");
    bdec = new JButton(".");
    beq = new JButton("=");
    clear = new JButton("Clear");
    dot = new JButton(".");

    b7.setBounds(90, 100, 50, 40);
    b8.setBounds(180, 100, 50, 40);
    b9.setBounds(270, 100, 50, 40);
    bdiv.setBounds(360, 100, 50, 40);

    b4.setBounds(90, 170, 50, 40);
    b5.setBounds(180, 170, 50, 40);
    b6.setBounds(270, 170, 50, 40);
    bmul.setBounds(360, 170, 50, 40);

    b1.setBounds(90, 240, 50, 40);
    b2.setBounds(180, 240, 50, 40);
    b3.setBounds(270, 240, 50, 40);
    bsub.setBounds(360, 240, 50, 40);

    bdec.setBounds(90, 310, 50, 40);
    b0.setBounds(180, 310, 50, 40);
    beq.setBounds(270, 310, 50, 40);
    badd.setBounds(360, 310, 50, 40);
    clear.setBounds(200, 400, 90, 40);
    dot.setBounds(250, 470, 50, 40);

    jrm.add(b7);
    jrm.add(b8);
    jrm.add(b9);
    jrm.add(bdiv);
    jrm.add(b4);
```

```
    jrm.add(b5);
    jrm.add(b6);
    jrm.add(bmul);
    jrm.add(b1);
    jrm.add(b2);
    jrm.add(b3);
    jrm.add(bsub);
    jrm.add(bdec);
    jrm.add(b0);
    jrm.add(beq);
    jrm.add(badd);
    jrm.add(clear);
    jrm.add(dot);

    jrm.setLayout(null);
    jrm.setResizable(false);
    jrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    jrm.setVisible(true);

    b1.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        if (jtf.getText().equals("Enter two operands!") ||
jtf.getText().equals("Not defined!"))
          jtf.setText("");
        jtf.setText(jtf.getText() + "1");
      }
    });
    b2.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        if (jtf.getText().equals("Enter two operands!") ||
jtf.getText().equals("Not defined!"))
          jtf.setText("");
        jtf.setText(jtf.getText() + "2");
      }
    });
    b3.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
```

```
        if (jtf.getText().equals("Enter two operands!") ||
jtf.getText().equals("Not defined!"))
          jtf.setText("");
        jtf.setText(jtf.getText() + "3");
      }
    });
    b4.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        if (jtf.getText().equals("Enter two operands!") ||
jtf.getText().equals("Not defined!"))
          jtf.setText("");
        jtf.setText(jtf.getText() + "4");
      }
    });
    b5.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        if (jtf.getText().equals("Enter two operands!") ||
jtf.getText().equals("Not defined!"))
          jtf.setText("");
        jtf.setText(jtf.getText() + "5");
      }
    });
    b6.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        if (jtf.getText().equals("Enter two operands!") ||
jtf.getText().equals("Not defined!"))
          jtf.setText("");
        jtf.setText(jtf.getText() + "6");
      }
    });
    b7.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        if (jtf.getText().equals("Enter two operands!") ||
jtf.getText().equals("Not defined!"))
          jtf.setText("");
        jtf.setText(jtf.getText() + "7");
      }
```

```
    });
    b8.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        if (jtf.getText().equals("Enter two operands!") ||
jtf.getText().equals("Not defined!"))
          jtf.setText("");
        jtf.setText(jtf.getText() + "8");
      }
    });
    b9.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        if (jtf.getText().equals("Enter two operands!") ||
jtf.getText().equals("Not defined!"))
          jtf.setText("");
        jtf.setText(jtf.getText() + "9");
      }
    });
    b0.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        if (jtf.getText().equals("Enter two operands!") ||
jtf.getText().equals("Not defined!"))
          jtf.setText("");
        jtf.setText(jtf.getText() + "0");
      }
    });
    clear.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        if (jtf.getText().equals("Enter two operands!") ||
jtf.getText().equals("Not defined!"))
          jtf.setText("");
        jtf.setText("");
      }
    });
    dot.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        if (jtf.getText().equals("Enter two operands!") ||
jtf.getText().equals("Not defined!"))
```

```
          jtf.setText("");
        jtf.setText(jtf.getText() + ".");
      }
    });
    badd.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        if (jtf.getText().equals("Enter two operands!") ||
jtf.getText().equals("Not defined!"))
          jtf.setText("");
        jtf.setText(jtf.getText() + "+");
      }
    });
    bsub.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        if (jtf.getText().equals("Enter two operands!") ||
jtf.getText().equals("Not defined!"))
          jtf.setText("");
        jtf.setText(jtf.getText() + "-");
      }
    });
    bmul.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        if (jtf.getText().equals("Enter two operands!") ||
jtf.getText().equals("Not defined!"))
          jtf.setText("");
        jtf.setText(jtf.getText() + "X");
      }
    });
    bdiv.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        if (jtf.getText().equals("Enter two operands!") ||
jtf.getText().equals("Not defined!"))
          jtf.setText("");
        jtf.setText(jtf.getText() + "/");
      }
    });
```

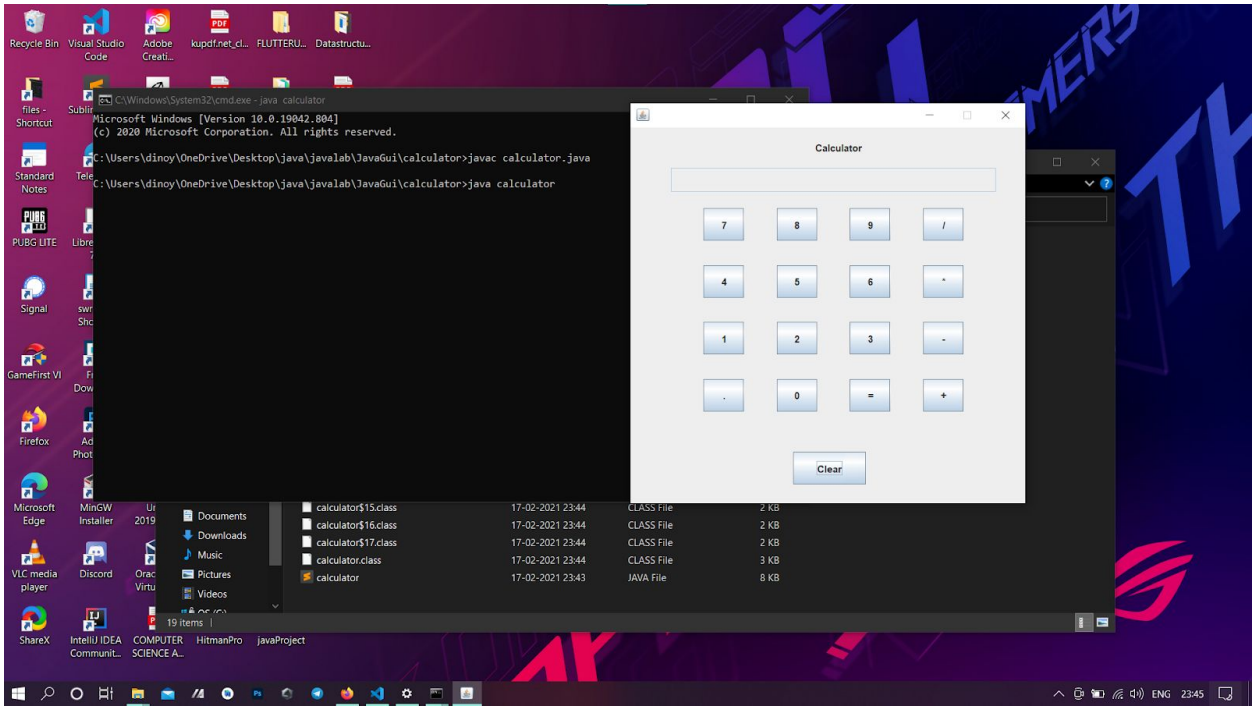
```
    beq.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        try {
          String S = jtf.getText();
          int i = 0;
          char b = S.charAt(i);
          while (b != '+' && b != '-' && b != 'X' && b != '/' && b != '%')
{
            i++;
            b = S.charAt(i);
          }
          Float x, y, ans = 0f;
          x = Float.parseFloat(S.substring(0, i));
          y = Float.parseFloat(S.substring(i + 1, S.length()));
          if (b == '+') {
            ans = x + y;
          } else if (b == '-') {
            ans = x - y;
          } else if (b == 'X') {
            ans = x * y;
          } else if (b == '/') {
            ans = x / y;
          }
          if (b == '/' && y == 0) {
            jtf.setText("Not defined!");
          } else {
            jtf.setText(ans + "");
          }
        } catch (Exception ex) {
          jtf.setText("Enter two operands!");
        }
      }
    });
  }

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

```
    }


}
```

Output


Calculator
7
8
9
/
4
5
6
*
1
2
3
-
.
0
=
+
Clear
Microsoft Windows [Version 10.0.19042.804]
(c) 2020 Microsoft Corporation. All rights reserved.
C:\Users\dinoy\OneDrive\Desktop\java\javalab\JavaGui\calculator>javac calculator.java
C:\Users\dinoy\OneDrive\Desktop\java\javalab\JavaGui\calculator>java calculator

Result

Working Calculator Is Made With Java Gui And Event Handling

Lab - 9

Experiment 17

15/09/2020

Aim : Write a Java program that simulates a traffic light. The program lets the user select one of three lights: red, yellow, or green. When a radio button is selected, the light is turned on, and only one light can be on at a time. No light is on when the program starts. .

Concepts Used : Java Swing And Event Handling

Algorithm

Steps :

Start

1.Import java.awt, java.awt.event and java.swing classes

2.JFrame frame = new JFrame("Traffic Light")

3.JRadioButton[] b = new JradioButton[3]

4.ButtonGroup bg = new ButtonGroup()

5.for i = 0 till 3

6.b[i]=newJRadioButton("")

7.b[i].setBackground(Color.GRAY)

8.bg.add(b[i])

9.frame.add(b[i])

10.endfor

11.frame.setSize(300,130)

12.frame.setLayout(new GridLayout(1,3))

13.frame.setVisible(true)

14.frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE)

15.//Event handler code for RED button

16.b[0].setBackground(Color.RED)

17.b[1].setBackground(Color.GRAY)

18.b[2].setBackground(Color.GRAY)

19.b[0].setText("STOP")

20.b[1].setText("")

21.b[2].setText("")

22.Follow similar procedure for YELLOW and GREEN buttons

23.STOP

Stop

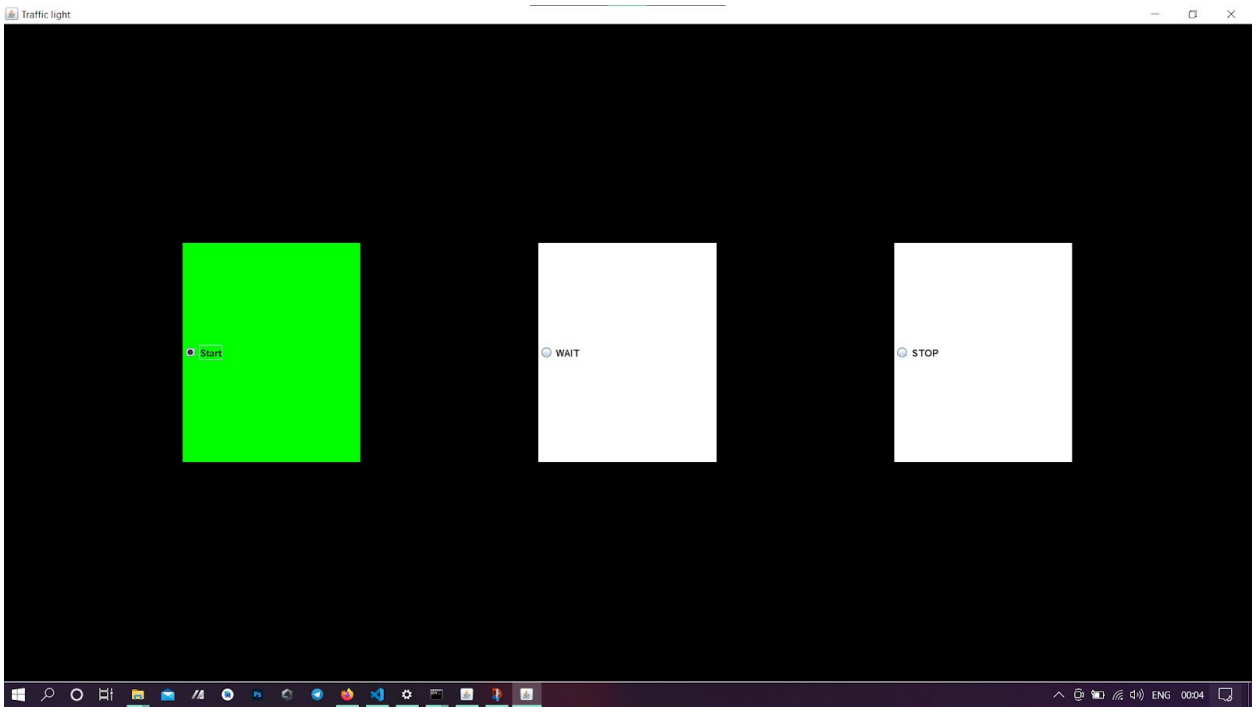
CODE

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
```

```
public class traffic{
   traffic(){
    JFrame T=new JFrame("Traffic light");
    JPanel p1=new JPanel();
    JPanel p2=new JPanel();
    JPanel p3=new JPanel();
    JPanel p4=new JPanel();
    JPanel p5=new JPanel();
    JPanel p6=new JPanel();
    JPanel p7=new JPanel();
    JRadioButton b[]=new JRadioButton[3];
    ButtonGroup bg=new ButtonGroup();
    b[0]=new JRadioButton("Start");
    b[0].setBackground(Color.WHITE);
    b[1]=new JRadioButton("WAIT");
    b[1].setBackground(Color.WHITE);
    b[2]=new JRadioButton("STOP");
    b[2].setBackground(Color.WHITE);
    bg.add(b[0]);
    bg.add(b[1]);
    bg.add(b[2]);
    p1.setBackground(Color.BLACK);
    p2.setBackground(Color.BLACK);
    p3.setBackground(Color.BLACK);
    p4.setBackground(Color.BLACK);
    p5.setBackground(Color.BLACK);
    p6.setBackground(Color.BLACK);
    p7.setBackground(Color.BLACK);
    p2.add(p4);
    p2.add(b[0]);
    p2.add(p5);
    p2.add(b[1]);
    p2.add(p6);
    p2.add(b[2]);
    p2.add(p7);
    p2.setLayout(new GridLayout(1,7));
    T.add(p1);
```

```
    T.add(p2);
    T.add(p3);
    T.setSize(450,300);
    T.setLayout(new GridLayout(3,1));
    T.setVisible(true);
    T.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    b[0].addActionListener(new ActionListener(){
      public void actionPerformed(ActionEvent e){
        b[0].setBackground(Color.GREEN);
        b[1].setBackground(Color.WHITE);
        b[2].setBackground(Color.WHITE);
      }
    });
    b[1].addActionListener(new ActionListener(){
      public void actionPerformed(ActionEvent e){
        b[0].setBackground(Color.WHITE);
        b[1].setBackground(Color.YELLOW);
        b[2].setBackground(Color.WHITE);
      }
    });
    b[2].addActionListener(new ActionListener(){
      public void actionPerformed(ActionEvent e){
        b[0].setBackground(Color.WHITE);
        b[1].setBackground(Color.WHITE);
        b[2].setBackground(Color.RED);
      }
    });
  }
  public static void main(String[] args){
    new traffic();
  }
}
```

Output



Result

Functional Traffic Light Is Made With Java Gui and Event Handling

Lab - 10

Experiment 18

Aim : Write a Java program that implements the binary search algorithm.

Concepts Used : Quick Sort algorithm

Algorithm

Steps :

Start

Algorithm partition(arr, p, r)

1.i = p –1

2.for j = p till r

3.if (arr[j] <= arr[r])

4.if(i++ != j)

5.Swap arr[i] and arr[j]

6.endif

7.endif

8.endfor

9.if(r != i + 1)

10.Swap arr[i+1] and arr[r]

11.endif

12.return i+1

Algorithm quickSort(arr, p, r)

1.if(p < r)

2.q = partition(arr, p, r)

3.quickSort(arr, p, q-1)

4.quickSort(arr, q+1, r)

5.endif

Stop

CODE

```
import java.util.*;
//import java.lang.*;


class quicksort{
```

```
String sortarr[];
int length;


/*quicksort(){

}*/

//used to take input froom the user
void input(){

    int itr=0;
    Scanner sc = new Scanner(System.in);
    System.out.println("\n Total Number Of String ");
    length = sc.nextInt();

    this.sortarr = new String[length];

    while(length>itr){

    System.out.print("\n "+(itr+1)+" th String : \n");
    sortarr[itr] = sc.next();
    itr++;

    }
    sort(0,length-1);


}



void sort(int l,int h){

  int higher = h;
```

```
    int lower =l;

    String pivot = sortarr[lower+(higher-lower)/2];

    while(lower<=higher){

      while(sortarr[lower].compareToIgnoreCase(pivot)<0){
          lower++;
      }

      while(sortarr[higher].compareToIgnoreCase(pivot)>0){
          higher--;
      }

      if(lower<=higher){

          swap(higher,lower);

          lower++;
          higher--;
      }

    }

    if(l<higher){

      sort(l,higher);
    }

    if (lower<h) {

        sort(lower,h);
    }

  }
```

```
    void swap(int higher,int lower){

      String temp =  sortarr[lower];
      sortarr[lower] = sortarr[higher];
      sortarr[higher] = temp;

    }

    void display(){

       quicksort sort1 = new quicksort();

         for(String c:sortarr){

             System.out.println("\n"+c);
         }

    }


    public static void main(String[] args) {

    Scanner s =new Scanner(System.in);
    quicksort st = new quicksort();

    int op =0;
    while(true){

    System.out.println("Menu\n\n1-input\n2-sort and display\n3-EXIT");
    op = s.nextInt();
    switch(op){

        case 1:
            st.input();
            break;
        case 2:
            st.display();
```

```
            break;
        case 3:
            System.exit(0);
        default:
            System.out.println("Error Select Correct Option");

}
}


}
}
```

Output

C:\Users\dinoy\OneDrive\Desktop\java\javalab\sort>javac quicksort.java

C:\Users\dinoy\OneDrive\Desktop\java\javalab\sort>java quicksort
Menu

1-input
2-sort and display
3-EXIT
1

Total Number Of String
4

1 th String :
dinoy

2 th String :
amal

3 th String :
ajith

4 th String :
binoy
Menu

1-input
2-sort and display
3-EXIT
2

ajith

amal

binoy

dinoy
Menu

1-input
2-sort and display
3-EXIT

Result

Quick Sort is performed on a list of names