INDEX

Experiment Number: 1a **Date: 30th November 2021**

Basic Programs in Java

PALINDROME

Aim:

To write a java program that checks whether a given string is palindrome or not.

Algorithm:

```
1. Start
2. Read String S from user
3. Initialize int i with 0
4. Initialize int length with S.length()
5. While i < length/2, do
    a. If S.charAt[i] is NOT S.charAt[length – i -1]
        i. Print “The String is not a palindrome!”
        ii. Exit program
    b. Else
        i. Increment value of i
6. End loop
7. Print “The String IS a palindrome!”
8. Stop
```

Code

```
/*
Name          :  ASHIS SOLOMON
Roll Number :  MDL20CS035
Class          :  CS3B

* Program to check whether a string is palindrome or not
*/
import java.util.Scanner;
public class palindrome {
  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter a string: ");
    String S = sc.nextLine();
    int length = S.length();
    int i = 0;
    while (i < length / 2) {
      if (S.charAt(i) != S.charAt(length - i - 1)) {
        System.out.println("The string is not a palindrome");
        return;}
      i++;}
    System.out.println("The string is a palindrome! ");}}
```

Output

```
javac palindrome.java
java palindrome
Enter a string: malayalam
The string is a palindrome!

java palindrome
Enter a string: hello
The string is not a palindrome
```

Experiment Number: 1b **Date: 30th November 2021**

Basic Programs in Java

FREQUENCY OF A GIVEN CHARACTER

Aim:

To write a java program to find the frequency of a given character in a string.

Algorithm:

```
1. Start
2. Read String str from user
3. Read char ch from user
4. Initialize count and i to 0
5. Loop (i<str.length())
        a. If str.charAt(i) == ch
                i. count++
                ii. End If
6. i++
7. End loop
8. Print "The character '"+ch+"' appeared "+count+" times in "+str+" !!!"
9. Stop
```

Code

```
/*
Name          :  ASHIS SOLOMON
Roll Number :  MDL20CS035
Class           :  CS3B

* Program to find the frequency of a given character in a string
*/
import java.util.Scanner;
class frequency{
  public static void main(String[] args){
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter the string : ");
    String str = sc.nextLine();
    System.out.println("\nEnter the character to be checked : ");
    char ch = sc.nextLine().charAt(0);
    int count = 0;
    for(int i=0;i<str.length();i++){
      if(str.charAt(i)==ch)
        count++;
}
    System.out.println("\nThe character '"+ch+"' appeared "+count+" times in "+str+" !!!");
}
}
```

Output

I

```
Enter the string :
elective

Enter the character to be checked :
e

The character 'e' appeared 3 times in elective !!!
```

II

```
Enter the string :
apple

Enter the character to be checked :
f

The character 'f' appeared 0 times in apple !!!
```

Experiment Number: 1c **Date: 30th November 2021**

Basic Programs in Java

MATRIX MULTIPLICATION

Aim:

To write a java program to multiply two given matrices.

Algorithm:

```
1. Start
2. Accept the number of rows and columns of matrix A from user and store it in rowA &
   colA.
3. Accept the number of rows and columns of matrix B from user and store it in rowB &
   colB.
4. if(colA! = rowB)
     a. Print “Matrix cannot be multiplied !!”
     b. return ;
5. initialize matrix A with rowA and colA
6. initialize matrix A with rowB and colB
7. initialize matrix A with rowC and colC
8. Accept the elements of matrix A from user.
9. Accept the elements of matrix B from user.
10. Initialize sum to 0
11. for every i from 0 to rowA-1 {
      a. for every j from 0 to colB-1 {
           i. sum = 0;
           ii. for every k from 0 to rowB-1
                   sum += A[i][k]*B[k][j];
           iii. C[i][j] = sum;
12. Display the resultant matrix C.
13. Stop
```

Code:

```
/*
Name          :  ASHIS SOLOMON
Roll Number :  MDL20CS035
Class           :  CS3B

*Program to implement Matrix Multiplication in Java.
*/
import java.io.*;
import java.util.*;
class matrmul {
  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter the number of rows of matrix A : ");
    int rowA = sc.nextInt();
    System.out.println("Enter the number of columns of matrix A : ");
    int colA = sc.nextInt();
    System.out.println("Enter the number of rows of matrix B : ");
    int rowB = sc.nextInt();
    System.out.println("Enter the number of columns of matrix B : ");
    int colB = sc.nextInt();
    if (colA != rowB) {
      System.out.println("Matrix cannot be multiplied !!");
      return;
    }
    int A[][] = new int[rowA][colA];
    int B[][] = new int[rowB][colB];
    int C[][] = new int[colA][rowB];
    System.out.println("Enter the elements of matrix A :\n");
    for (int i = 0; i < rowA; i++)
      for (int j = 0; j < colA; j++)
        A[i][j] = sc.nextInt();
    System.out.println("Enter the elements of matrix B :\n");
    for (int i = 0; i < rowB; i++)
      for (int j = 0; j < colB; j++)
        B[i][j] = sc.nextInt();
    System.out.println("Matrix A :\n");
    for (int i = 0; i < rowA; i++) {
      for (int j = 0; j < colA; j++)
        System.out.print(A[i][j] + "\t");
      System.out.println("\n");
    }
    System.out.println("Matrix B :\n");
    for (int i = 0; i < rowB; i++) {
      for (int j = 0; j < colB; j++)
        System.out.print(B[i][j] + "\t");
      System.out.println("\n");
    }
    int sum = 0;
    for (int i = 0; i < rowA; i++) {
      for (int j = 0; j < colB; j++) {
        sum = 0;
        for (int k = 0; k < rowB; k++) {
          sum += A[i][k] * B[k][j];
        }
        C[i][j] = sum;
      }
    }
    System.out.println("Matrix C :\n");
    for (int i = 0; i < colA; i++) {
      for (int j = 0; j < rowB; j++)
        System.out.print(C[i][j] + "\t");
      System.out.println("\n");
    }
  }
}
```

Output:

```
javac matr_mul.java
java matr_mul
Enter the number of rows of matrix A :
2
Enter the number of columns of matrix A :
2
Enter the number of rows of matrix B :
2
Enter the number of columns of matrix B :
2
Enter the elements of matrix A :

1 2 3 4
Enter the elements of matrix B :

1 2 3 4
Matrix A :

1     2
3     4


Matrix B :

1     2
3     4
Matrix C :

7     10
15     22
```

Experiment Number: 2a **Date: 7th December 2021**

OOPS Concepts

INHERITANCE

Aim:

To write a java program that implements inheritance in java.

Algorithm:

1. Start
2. Create a base class named employee
3. Declare data members inside it as age, name, address, phone and salary
4. Also declare a method called printSalary and display the details
5. Then declare a derived class named Officer with data member ‘specialization’
6. Declare another derived class Manager with data member ‘department’
7. In main method, read the details from user
8. Print details of Officer and Manager
9. Stop

Code:

```
/*
Name          :  ASHIS SOLOMON
Roll Number :  MDL20CS035
Class          :  CS3B

*Program to implement inheritance in Java.
*/
import java.util.Scanner;

class employee {
    String name, address;
    int age, phone_number;
    float salary;

    void salary() {
        System.out.println("The salary of the employee is: " + salary);
    }
}

class Officer extends employee {
    String specialization;
}

class Manager extends employee {
    String department;
}

class inheritance {
    public static void main(String[] args) {
        Officer o = new Officer();
        Manager m = new Manager();
        Scanner sc = new Scanner(System.in);
        System.out.println("OFFICER");
        System.out.print("Enter name: ");
        o.name = sc.nextLine();
        System.out.print("Enter age: ");
        o.age = sc.nextInt();
        System.out.print("Enter address: ");
        sc.nextLine();
        o.address = sc.nextLine();
        System.out.print("Enter phone number: ");
        o.phone_number = sc.nextInt();
        System.out.print("Enter salary: ");
        o.salary = sc.nextFloat();
        System.out.print("Enter specialization: ");
        sc.nextLine();
        o.specialization = sc.nextLine();
        System.out.println("\nMANAGER");
        System.out.print("Enter name: ");
        m.name = sc.nextLine();
        System.out.print("Enter age: ");
        m.age = sc.nextInt();
        System.out.print("Enter address: ");
        sc.nextLine();
        m.address = sc.nextLine();
        System.out.print("Enter phone number: ");
        m.phone_number = sc.nextInt();
        System.out.print("Enter salary: ");
        m.salary = sc.nextFloat();
        System.out.print("Enter Department: ");
        sc.nextLine();
        m.department = sc.nextLine();

        System.out.println("\n\nOFFICER DETAIL");
        System.out.println("Name: " + o.name);
        System.out.println("Age: " + o.age);
        System.out.println("Address: " + o.address);
```

```
        System.out.println("Phone number: " + o.phone_number);
        o.salary();
        System.out.println("Specialization: " + o.specialization);

        System.out.println("\nMANAGER DETAIL");
        System.out.println("Name: " + m.name);
        System.out.println("Age: " + m.age);
        System.out.println("Address: " + m.address);
        System.out.println("Phone number: " + m.phone_number);
        m.salary();
        System.out.println("Department: " + m.department);
    }
}
```

Output:

```
java inheritance
OFFICER
Enter name: Bob
Enter age: 23
Enter address: 41st Street, Michigan
Enter phone number: 123456789
Enter salary: 22000
Enter specialization: IOT

MANAGER
Enter name: Ross
Enter age: 32
Enter address: 11th Street, Colorado
Enter phone number: 987654321
Enter salary: 69420
Enter Department: Game Dev
```

```
OFFICER DETAIL
Name: Bob
Age: 23
Address: 41st Street, Michigan
Phone number: 123456789
The salary of the employee is: 22000.0
Specialization: IOT

MANAGER DETAIL
Name: Ross
Age: 32
Address: 11th Street, Colorado
Phone number: 987654321
The salary of the employee is: 69420.0
Department: Game Dev
```

Experiment Number: 2b **Date: 14th December 2021**

OOPS Concepts

ABSTRACT CLASS

Aim:

To write a java program that implements abstract class.

Algorithm:

1. Start
2. Create a base abstract class named shape and declare an abstract method NoOfSides()
3. Then declare a derived class called rectangle which extends shape and define the method NoOfSides.() inside rectangle class
4. Then declare another derived class triangle which extends shape and define the method NoOfSides.() inside triangle class
5. Declare another derived class hexagon which extends shape and define the method NoOfSides.() inside hexagon class
6. Then inside main class create object for each derived class.
7. Then call NoOfSides methods in each class
8. Print
9. Stop

Code:

```
/*
Name          :  ASHIS SOLOMON
Roll Number :  MDL20CS035
Class           :  CS3B

*Program to implement abstract class in Java.
*/
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:

```
java polygon
Triangle : 3
Rectangle : 4
Hexagon : 6
```

Experiment Number: 2c **Date: 14th December 2021**

OOPS Concepts

GARBAGE COLLECTION

Aim:

To write a java program that demonstrates Garbage Collection

Algorithm:

1. Start
2. Create a class memorydemo
3. Inside main create an object r for the class Runtime
4. Declare long variable mem1
5. Set mem1 = r.freeMemory()
6. Declare a large matrix of type long
7. Declare long variable mem2
8. Set mem2 = r.freeMemory()
9. Nullify the long matrix
10. Call garbage collector
11. Declare long variable mem3
12. Set mem3 = r.freeMemory()
13. Print mem1, mem2, mem3 and mem3-mem2
14. Stop

Code:

```
/*
Name            :  ASHIS SOLOMON
Roll Number :  MDL20CS035
Class            :  CS3B

* Program to demonstrate Garbage collection in Java
*/

public class MemoryDemo {
  public static void main(String[] args) {
    System.gc();
    long mem1, mem2, mem3;
    mem1 = Runtime.getRuntime().freeMemory();
    long[][] l = new long[100][100];
    mem2 = Runtime.getRuntime().freeMemory();
    l = null;
    System.gc();
    mem3 = Runtime.getRuntime().freeMemory();
    System.out.println("Initial memory: " + mem1);
    System.out.println("After matrix allocation: " + mem2);
    System.out.println("After nullification and garbage collection: " + mem3);
    System.out.println("Memory saved: " + (mem3 - mem2));
  }
}
```

Output:

```
java MemoryDemo
Initial memory: 7826312
After matrix allocation: 7783560
After nullification and garbage collection: 7868048
Memory saved: 84488
```

Experiment Number: 3a **Date: 21st December 2021**

File handling & IO management

FILE HANDLING WITH READER/WRITER

Aim:

To write a java program that demonstrates the use of Reader/Writer class in file handling

Algorithm:

1. Start
2. Create an object ‘file’ for file class and pass the path of the file
3. Create a main method with throws IO exception
4. Create an object ‘r’ of Reader class and pass in the ‘file’ object
5. Reach each individual character in the file using ‘read()’ method in Reader class and print them
6. Close object ‘r’
7. Stop

Code:

```
/*
Name          :  ASHIS SOLOMON
Roll Number :  MDL20CS035
Class           :  CS3B

* Program to implement File Handling with Reader/Writer in java.
*/


import java.io.*;

class readingFile {
   public static void main(String args[]) throws FileNotFoundException, IOException {
      File file = new File("C:/Users /Desktop/OOPS/ FileHandle.txt");
      Reader r = new FileReader(file);
      int c = r.read();
      while (c != -1) {
         System.out.print((char) c);
         c = r.read();
      }
      r.close();
   }
}
```

Output:

```
java readingFile
Hello, my name is Bob Ross
```

Experiment Number: 3b **Date: 21st December 2021**

File handling & IO management

FILE EXCEPTION HANDLING

Aim:

To write a java program that handles exceptions that are thrown while working with files

Algorithm:

1. Start
2. Begin a try block
3. Create an object f1 for FileOutputStream and pass in the path to one text file
4. Create an object f2 for FileInputStream and pass in the path to another text file
5. Read the contents of f2 and write them to f1
6. Print “Written Successfully”
7. End try block
8. Create one catch block to catch ‘FileNotFoundException’
9. Create one catch block to catch ‘IOException’
10. Stop

Code:

```
/*
Name          :  ASHIS SOLOMON
Roll Number :  MDL20CS035
Class           :  CS3B

* Program to demonstrate File Exception Handling
*/

import java.util.*;

public class FileStream {
   public static void main(String a[]) throws IOException, FileNotFoundException {
      int i;
      try {
         FileOutputStream f1 = new FileOutputStream("xyz.txt");
         FileInputStream f2 = new FileInputStream("abc.txt");
         while ((i = f2.read()) != -1) {
            f1.write((char) i);
         }
         System.out.println("Written Successfully");
         f1.close();
         f2.close();
         FileInputStream f3 = new FileInputStream("xyz.txt");
         }
      } catch (FileNotFoundException e) {
         System.out.println("e");
      } catch (IOException e) {
         System.out.println("e");
      }
   }
}
```

Output:

```
java Fileexceptionhandle
Written Successfully
Bob Ross is a great painter.


java Fileexceptionhandle
java.io.FileNotFoundException: abc.txt (The system cannot find the file specified)
```

Experiment Number: 3c **Date: 22nd February 2022**

File handling & IO management

STRING TOKENIZER

Aim:

To write a java program to show the use of StringTokenizer.

Algorithm:

```
1. Start
2. Define class StringTokenizerEg.
3. Inside main(), declare sum to 0.
4. Get string of integers from user.
5. Declare StringTokenizer object ‘st’ by passing string S and ‘,’ as arguments.
6. while(st.hasMoreTokens())
            int t = Integer.parseInt(st.nextToken())
            Print t.
            sum += t.
7. Print sum
8. Stop
```

Code

```
/*
Name          :  ASHIS SOLOMON
Roll Number :  MDL20CS035
Class          :  CS3B

* Program to demonstrate the usage of String Tokenizer
*/
import java.util.*;
class StringTokenizerEg {
   public static void main(String a[]) {
      Scanner sc = new Scanner(System.in);
      int sum = 0;
      System.out.println("Enter a line of integers separated by commas: ");
      String S = sc.nextLine();
      StringTokenizer st = new StringTokenizer(S, ",");
      System.out.println("Entered integers are: ");
      while (st.hasMoreTokens()) {
         int t = Integer.parseInt(st.nextToken());
         System.out.print(t + "  ");
         sum += t;
      }
      System.out.println();
      System.out.println("Sum = " + sum);
   }
}
```

Output

```
Enter a line of integers separated by commas:
1,4,6,9,3,0
Entered integers are:
1 4 6 9 3 0
Sum = 23
```

Experiment Number: 4a **Date: 25th January 2022**

Exception Handling & Multithreading

USING TRY, CATCH, THROWS AND FINALLY

Aim:

To write a java program to show the usage of try, catch, throws and finally

Algorithm:

```
1. Start
2. Read int age from the user
3. Start Try Block
    a. Call checkAge() with parameter age passed to it
4. End Try Block
5. Catch(ArithmeticException e)
    a. Print the error message
6. End Catch Block
7. Start Finally Block
8. Print "THANK YOU!!!"
9. End Finally Block
10. Stop
```

Method: checkAge()

```
1. Start
2. If age < 18
    a. throw new ArithmeticException("Access denied - You must be at least 18 years
       old.")
3. End If
4. Else
    a. Print "Access granted - You are old enough!"
5. End Else
6. Stop
```

Code

```
/*
Name          :  ASHIS SOLOMON
Roll Number :  MDL20CS035
Class          :  CS3B

* Program to show the usage of try, catch, throws and finally
*/

import java.util.Scanner;
public class errorcheck {
  public static void checkAge(int age) throws ArithmeticException {
    if (age < 18) {
      throw new ArithmeticException("Access denied - You must be at least 18 years old.");}
    else {
      System.out.println("Access granted - You are old enough!");}}
  public static void main(String[] args) {
    int age;
    Scanner in = new Scanner(System.in);
    System.out.println("Enter the age: ");
    age = in.nextInt();
    try{
      checkAge(age);}
    catch(ArithmeticException e){
      System.out.println(e);}
    finally{
      System.out.println ("THANK YOU!!!");}}}
```

Output

```
javac errorcheck.java
java errorcheck
Enter the age:
23
Access granted - You are old enough!
THANK YOU!!!

java errorcheck
Enter the age:
14
java.lang.ArithmeticException: Access denied - You must be at least 18 years old.
THANK YOU!!!
```

Experiment Number: 4b **Date: 15th February 2022**

Exception Handling & Multithreading

MULTITHREADING

Aim:

To write a java program that implements multithreading.

Algorithm:

```
1. Start
2. Create class Even which extends Thread
    a. Create variable int x;
    b. Create parameterized constructor
        Even(int n){
            x = n;
        }
    c. public void run(){
            int square  = x * x;
            Print "Square of Even number " + x + " is : " + square
            }
3. Create class Odd which extends Thread
    a. Create variable int x;
    b. Create parameterized constructor
        Odd(int n){
            x = n;
        }
    c. public void run(){
            int cube  = x * x * x;
            Print "Cube of Odd number " + x + " is : " + cube}
```

```
4. Create class Number which extends Thread
    a. public void run()
        i. initialize int i = -1;
        ii. loop 5 times
            a. int r = random number from 0 to 100
            b. Print "Random Integer : " + r
            c. if(r%2 == 0)
                    Even e = new Even(r);
                    e.start();
            d. else
                    Odd o = new Odd(r);
                    o.start();
            e. try{
                  Thread.sleep(1000);
               }
            f. catch(Exception e){System.out.println(e);}

5. Create class multithread with main method.
6. Number num = new Number();
7. num.start();
8. Stop
```

Code:

```
/*
Name          :  ASHIS SOLOMON
Roll Number :  MDL20CS035
Class          :  CS3B

*Program to implement Multithreading in Java.
*/
import java.util.*;
import java.io.*;

class Even extends Thread {
  int x;
  Even(int n) {
    x = n;
  }
  public void run() {
    int square = x * x;
    System.out.println("Square of Even number " + x + " is : " + square);
    System.out.println();
  }
}

class Odd extends Thread {
  int x;
  Odd(int n) {
    x = n;
  }
  public void run() {
    int cube = x * x * x;
    System.out.println("Cube of Odd number " + x + " is : " + cube);
    System.out.println();
  }
}
```

```
class Number extends Thread {
  public void run() {
    Random random = new Random();
    int i = -1;
    while (++i < 5) {
      int r = random.nextInt(100);
      System.out.println("Random Integer : " + r);
      if (r % 2 == 0) {
        Even e = new Even(r);
        e.start();
      } else {
        Odd o = new Odd(r);
        o.start();
      }
      try {
        Thread.sleep(1000);
      } catch (Exception e) {
        System.out.println(e);
      }
    }
  }
}

class multithread {
  public static void main(String[] args) {
    Number num = new Number();
    num.start();
  }
}
```

Output:

```
javac multithread.java
java multithread

Random Integer : 55
Cube of Odd number 55 is : 166375

Random Integer : 91
Cube of Odd number 91 is : 753571

Random Integer : 6
Square of Even number 6 is : 36

Random Integer : 82
Square of Even number 82 is : 6724

Random Integer : 36
Square of Even number 36 is : 1296
```

Experiment Number: 4c **Date: 15th February 2022**

Exception Handling & Multithreading

THREAD SYNCHRONIZATION

Aim:

To write a java program that implements thread synchronization.

Algorithm:

```
1. Start
2. Create a class Account with instance variables double balance, double wamt along with
   a parameterized constructor and two synchronized methods withdraw() and deposit().
3. Initialize the parameter balance with the value of b.
4. Declare Thread1 and Thread2 which implements Runnable along with a parameterized
   constructor as follows:
   Thread (Account a){
               this.a = a;
               Thread t = new Thread(this);
               t.start();
         }
5. Thread1 and Thread2 calls a.withdraw() and a.deposit() respectively inside public void
   run().
6. Create class threadsync and create the main method.
   a. Create object acc of Account using
      i. Account acc = new Account(10000);
   b. Create new Thread1(acc);
   c. Create new Thread2(acc);
7. Stop
```

Method: **withdraw()**

```
1. Start
2. Print "Withdraw Process Started !!"
3. Print "Enter amount to withdraw : "
4. Accept wamt from user.
5. Print “Balance before withdrawal : ”  + balance
6. if(balance < wamt)
       a. Print “Insufficient Balance waiting for deposit ..."
       b. try{
            wait();
          }
       c. catch(Exception e){ }
7. balance = balance - wamt;
8. Print “Balance after withdrawal : ” + balance
9. return wamt;
10. Stop
```

Method: **deposit()**

```
1. Start
2. Print "Deposit Process started !!"
3. Print "Balance before deposit : " + balance
4. balance = balance + damt;
5. Print "Balance after deposit : " + balance
6. this.notify()
7. Stop
```

Code:

```
/*
Name          :  ASHIS SOLOMON
Roll Number :  MDL20CS035
Class           :  CS3B

*Program to implement Thread Synchronization in Java.
*/
import java.io.*;
import java.util.*;

class Account {
  double balance;
  double wamt;
  Account(double b) {
    balance = b;
  }

  synchronized public double withdraw() {
    System.out.println("Withdraw Process Started !!");
    System.out.println("Enter amount to withdraw : ");
    java.util.Scanner sc = new java.util.Scanner(System.in);
    wamt = sc.nextDouble();
    System.out.println("\nBalance before withdrawal : " + balance);

    if (balance < wamt) {
      System.out.println("\nInsufficient Balance waiting for deposit ...\n");
      try {
        wait();
      } catch (Exception e) {}
    }

    balance = balance - wamt;
    System.out.println("Balance after withdrawal : " + balance);
    return wamt;
  }
```

```
    synchronized public void deposit(double damt) {
    System.out.println("Deposit Process started !!");
    System.out.println("Balance before deposit : " + balance);
    balance = balance + damt;
    System.out.println("Balance after deposit : " + balance);
    this.notify();
  }
}

class Thread1 implements Runnable {
  Account a;
  Thread1(Account a) {
    this.a = a;
    Thread t = new Thread(this);
    t.start();
  }

  public void run() {
    a.withdraw();
  }
}

class Thread2 implements Runnable {
  Account a;
  Thread2(Account a) {
    this.a = a;
    Thread t = new Thread(this);
    t.start();
  }

  public void run() {
    a.deposit(5000);
  }
}
```

```
class threadsync {
  public static void main(String args[]) {
    Account acc = new Account(10000);
    new Thread1(acc);
    new Thread2(acc);
  }
}
```

Output:

```
java threadsync
Withdraw Process Started !!
Enter amount to withdraw :
12000

Balance before withdrawal : 10000.0

Insufficient Balance waiting for deposit ...

Deposit Process started !!
Balance before deposit : 10000.0
Balance after deposit : 15000.0
Balance after withdrawal : 3000.0
```

Experiment Number: 5a **Date: 11th January 2022**

Search & Sort

DOUBLY LINKED LIST

Aim:

To write a Java program that implements doubly linked list and to perform the following:

i) Insert a node at the beginning of the list.
ii) Insert a node after a particular node in the list.
iii) Insert a node at the end of the list.
iv) Delete a node containing a particular item.
v) Display the contents of the list.

Algorithm:

```
1. Start
2. Define class Node inside main class with instance variables int data, Node prev and
   Node next and create a parameterised constructor to initialise instance variable data
3. Initialise Node head = null
                Node tail = null
4. Procedure addToLast() : int data
      a. Set Node newNode = new Node(data)
      b. If head = null
            i. Set head = newNode
                   tail = newNode
                   head.prev = null
                   tail.next = null
      c. Else
            i. Set tail.next = newNode
                   newNode.prev = tail
                   tail = newNode
                   tail.next = null
```

```
        d. End Procedure
5. Procedure addToFront() : int data
        a. Set Node newNode = new Node(data)
        b. If head = null
              i. Set head = newNode
                    tail = newNode
                    head.prev = null
                    tail.next = null
        c. Else
              i.   Set  head.prev = newNode
                        newNode.next = head
                        head = newNode
                        head.prev = null
        d. End Procedure
6. Procedure InsertAfter() : int data, int item
        a. Set Node newNode = new Node(data)
        b. If head = null
              i. Print "The given previous node cannot be null"
             ii. Return to main()
        c. Set Node current = head
        d. While current != null and current.data != data, do
              i.  Set current = current.next
        e. Set Node prev = current
             Node newNode = new Node(item)
             Node next = prev.next
             next.prev = newNode
             newNode.prev = prev
             prev.next = newNode
             newNode.next = next
        f. End Procedure
7. Procedure deleteNode() : int data
        a. If head = null
              i. Print "The list is Empty!!!"
             ii. Return to main()
        b. Set Node current = head
        c. while current != null and current.data != data, do
            i.   Set current = current.next
        d. Set Node ptr = current
        e. If ptr != null
            i.  If ptr.prev != null
                   Set ptr.prev.next = ptr.next
           ii. Else
                   Set head = ptr.next;
          iii. If ptr.next != null
                   Set ptr.next.prev = ptr.prev
           iv. Else
                   Set tail = ptr.prev
        f. End Procedure
8. Procedure showData() :
        a. Set Node current = head
        b. If head = null
            i.   Print "List is empty"
           ii.   Return to main()
        c. Print "Nodes of doubly linked list :"
        d. While current != null, do
            i.   Print current.data
           ii.   Set current = current.next
        e. End Procedure
9. Stop
```

Code

```
/*
Name          :  ASHIS SOLOMON
Roll Number :  MDL20CS035
Class           :  CS3B

* Program to implement Doubly Linked List and to perform  and to perform the
following :
 i)   Insert a node at the beginning of the list .
ii)   Insert a node after a particular node in the list .
iii)  Insert a node at the end of the list .
iv)  Delete a node containing a particular item
v)   Display the contents of the list  .

*/
import java.util.Scanner;
public class DoublyLinkedList {
    class Node {
        int data;
        Node prev;
        Node next;
        public Node(int data) {
            this.data = data;
        }
    }
    Node head = null;
    Node tail = null;
    public void addToLast(int data) {
        Node newNode = new Node(data);
        if (head == null) {
            head = newNode;
            tail = newNode;
            head.prev = null;
            tail.next = null;
    } else {
        tail.next = newNode;
        newNode.prev = tail;
        tail = newNode;
        tail.next = null;
    }
}
public void addToFront(int data) {
    Node newNode = new Node(data);
    if (head == null) {
        head = newNode;
        tail = newNode;
        head.prev = null;
        tail.next = null;
    } else {
        head.prev = newNode;
        newNode.next = head;
        head = newNode;
        head.prev = null;

    }
}
public void InsertAfter(int data, int item) {
    if (head == null) {
        System.out.println("The given previous node cannot be NULL ");
        return;
    }
    Node current = head;
    while (current != null && current.data != data) {
        current = current.next;
    }
    Node prev = current;
    Node newNode = new Node(item);
    Node next = prev.next;
    next.prev = newNode;
    newNode.prev = prev;
    prev.next = newNode;
    newNode.next = next;
}
public void deleteNode(int data) {
    if (head == null) {
        System.out.println("The List is Empty!");
        return;
    }
    Node current = head;
    while (current != null && current.data != data) {
        current = current.next;
    }
    Node ptr = current;
    if (ptr != null) {
        if (ptr.prev != null)
            ptr.prev.next = ptr.next;
        else
            head = ptr.next;
        if (ptr.next != null)
            ptr.next.prev = ptr.prev;
        else
            tail = ptr.prev;
    }
}
public void showData() {
    Node current = head;
    if (head == null) {
        System.out.println("List is empty");
        return;
    }
    System.out.println("Nodes of doubly linked list: ");
    while (current != null) {
            System.out.print(current.data + "  ");
            current = current.next;
        }
        System.out.println("\n");
    }
    public static void main(String[] args) {
        DoublyLinkedList obj = new DoublyLinkedList();
        Scanner sc = new Scanner(System.in);
        int choice, item, data;
        System.out.print("MENU : \n1.Insert at the front \n2.Insert at the back");
        System.out.print("\n3.Insert after a particular node \n4.Delete a node");
        System.out.println("\n5.Display \n6.Exit");
        do {
            System.out.print("\nEnter your choice : ");
            choice = sc.nextInt();
            switch (choice) {
            case 1:
                System.out.print("Enter the data :");
                item = sc.nextInt();
                obj.addToFront(item);
                break;
            case 2:
                System.out.print("Enter the data :");
                item = sc.nextInt();
                obj.addToLast(item);
                break;
            case 3:
                System.out.print("Enter the data of node after which new node is to be added : ");
                item = sc.nextInt();
                System.out.print("Enter the data to be inserted :");
                data = sc.nextInt();
                obj.InsertAfter(item, data);
                break;
            case 4:
```

```
            System.out.print("Enter the data of node to be deleted:");
            item = sc.nextInt();
            obj.deleteNode(item);
            break;
          case 5:
            obj.showData();
            break;
          case 6:
            System.exit(0);
          default:
            System.out.print("Invalid choice!\n");
          }
        } while (choice <= 6 && choice >= 1);
    }
}
```

Output

```
MENU :
1.Insert at the front
2.Insert at the back
3.Insert after a particular node
4.Delete a node
5.Display
6.Exit

Enter your choice : 1
Enter the data :1

Enter your choice : 1
Enter the data :2

Enter your choice : 2
Enter the data :3
Enter your choice : 2
Enter the data :4

Enter your choice : 5
Nodes of doubly linked list:
2  1  3  4


Enter your choice : 3
Enter the data of node after which new node is to be added : 3
Enter the data to be inserted :5

Enter your choice : 5
Nodes of doubly linked list:
2  1  3  5  4


Enter your choice : 4
Enter the data of node to be deleted:5

Enter your choice : 5
Nodes of doubly linked list:
2  1  3  4
```

Experiment Number: 5b **Date: 18th January 2022**

Search & Sort

QUICKSORT

Aim:

To write a java program that implements Quick sort algorithm for sorting a list of names in ascending order.

Algorithm:

```
1. Start
2. Procedure partition() : String arr[], int low, int high
      a. Set String pivot = arr[high]
         int i = low-1
      b. Set int j=low
      c. While j<high, do
      d. If arr[j].compareTo(pivot) <= 0
            i.   Set i = i + 1
                 String temp = arr[i]
                 arr[i] = arr[j]
                 arr[j] = temp
      e. Set j = j + 1
      f. End loop
      g. Set String temp = arr[i+1]
         arr[i+1] = arr[high]
         arr[high] = temp
      h. Return i + 1
      i. End Procedure
3. Procedure sort() : String arr[], int low, int high
      a. If low<high
            i.   Set int pi = partition(arr, low, high)
            ii.  Call procedure sort(arr, low, pi-1)
          iii.   Call procedure sort(arr, pi+1, high)
      b. End Procedure
4. Procedure printArray() : String arr[]
      a. Set int n = arr.length
      b. Set int i = 0
      c. While i<n, do
            i.   Print arr[i]
            ii.   Set i = i +1
      d. End loop
      e. End Procedure
5. Method main() :
      a. Print “Enter the size of list :”
      b. Read the user input and store it in variable n
      c. Declare an array a[] of type String and size n
      d. Print “ Enter the string :  ”
      e. Read n strings and store it in array a[]
      f. Set quicksort obj = new quicksort()
      g. Call procedure sort(a,0,n-1) using object obj
      h. Print “ Sorted array : ”
      i. Call procedure  printArray(a) using object obj
      j. End main()
6. Stop
```

Code

```
/*
Name           :  ASHIS SOLOMON
Roll Number :  MDL20CS035
Class            :  CS3B

* Program to implement Quick sort algorithm for sorting a list of names in ascending
order.
*/
import java.util.*;
class quicksort
{ int partition(String arr[],int low,int high)
  { String pivot = arr[high];
    int i=low-1;
    for(int j=low;j<high;j++)
    { if(arr[j].compareTo(pivot) <= 0)
      { i++;
        String temp = arr[i];
        arr[i] = arr[j];
        arr[j] = temp;
      }
    }
    String temp = arr[i+1];
    arr[i+1] = arr[high];
    arr[high] = temp;
    return i+1;
  }
  void sort(String arr[],int low,int high)
  { if(low<high)
    { int pi = partition(arr,low,high);
      sort(arr,low,pi-1);
      sort(arr,pi+1,high);
    }
  }
  void printArray(String arr[])
  { int n = arr.length;
    for(int i=0;i<n;i++)
    { System.out.print(arr[i]+" ");
    }
    System.out.println(" ");
  }
  public static void main(String args[])
  { Scanner sc=new Scanner(System.in);
    System.out.print("Enter the size of list : ");
    int n = sc.nextInt();
    String a[] = new String[n];
    System.out.println("Enter the string : ");
    sc.nextLine();
    for(int i=0;i<n;i++)
    { a[i] = sc.nextLine();
    }
    quicksort obj = new quicksort();
    obj.sort(a,0,n-1);
    System.out.println("Sorted array: ");
    obj.printArray(a);
  }
}
```

Output

```
Enter the size of list: 5
Enter the string:
allen
mia
grace
jenny
collin
Sorted array:
allen collin grace jenny mia
```

Experiment Number: 5c **Date: 18th January 2022**

Search & Sort

BINARY SEARCH

Aim:

To write a java program that implements the binary search algorithm.

Algorithm:

```
1. Start
2. Procedure binarySearch() : int array[], int x, int low, int high
3. While low <= high, do
      a. Set int mid = (high + low) / 2
      b. If array[mid] = x
            i. Return mid
      c. Else if array[mid] < x
            i. Set low = mid + 1
      d. Else
            i. Set high = mid - 1
4. End loop
5. Return -1
6. End
```

Method main()

```
1. Set binarysearch ob = new binarysearch()
2. Print "Enter array size : "
3. Read user input and store it in variable n
4. Create an array a[] of type int and size n
5. Enter n elements and store it in a[]
6. Print " Enter the element to search : "
7. Read user input and store it in variable item
8. Set int result = ob.binarySearch(a, item,0, n-1)
9. int pos = result + 1
```

```
10. If result = -1
        a. Print " Element not found "
11. Else
        a. Print " Element found at index "  result  " and position "  pos
12. End main()
13. Stop
```

Code

```
/*
Name          :  ASHIS SOLOMON
Roll Number :  MDL20CS035
Class          :  CS3B

* Program to implement Binary Search algorithm
*/
import java.util.Scanner;
class binarysearch
 {  int binarySearch(int array[], int x, int low, int high) {
    while(low <= high)
    { int mid = (high + low) / 2;
      if(array[mid] == x)
        return mid;
      if(array[mid] < x)
        low = mid + 1;
      else
        high = mid - 1;
    }
    return -1;
  }
  public static void main(String args[])
  { binarysearch ob = new binarysearch();
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter array size :");
    int n= sc.nextInt();
    int a[] = new int [n];
    System.out.print("Enter array elements :");
    for(int i=0;i<n;i++)
     a[i] = sc.nextInt();
    System.out.print("Enter the element to search :");
```

```
   int item = sc.nextInt();
   int result = ob.binarySearch(a, item,0, n-1);
   int pos = result+1;
   if(result == -1)
     System.out.println("Element not found");
   else
     System.out.println("Element found at index " + result+" and position "+pos);
 }
}
```

<u>Output</u>

```
Enter array size :7
Enter array elements :1 2 3 4 7 8 9
Enter the element to search :3
Element found at index 2 and position 3

Enter array size :7
Enter array elements :1 2 3 4 7 8 9
Enter the element to search :5
Element not found
```

Experiment Number: 6a **Date: 04 March 2022**

Graphics Programming

CALCULATOR

Aim:

To write a java program that works as a GUI calculator using java swing.

Algorithm:

```
1. Start
2. Define class Calculator with members
        1. frame of type JFrame
        2. textfield of type JTextfield
        3. 10 numberButtons of type JButton
        4. 9 functionButtons of type JButton
        5. panel of type JPanel
        6. myFont, font of choice
        7. num1, num2 and result of type double initialized to 0
        8. operator of type char
3. Inside Calculator constructor
        1. Set frame "Calculator" and textfield with its fields
        2. Set functionButtons 0 to 8 with corresponding symbols, listeners, and other
   fields
        3. Set numberButtons 0 to 9 with corresponding symbols, listeners, and other
        fields
        4. Set panel and its other fields. Add buttons into panel
        5. Add panel, textfield and remaining 3 buttons to frame
4. Overrriden function actioPerformed : ActionEvent e
        1. if e is number numberButtons
                Insert number into textbox
        2. if e is decimal button
```

```
            Insert '.' into textbox
        3. if e is any operator buttons
            Assign num2 as current content in textbox
            Assign operator and clear textbox
        4. if e is equal button
            Assign num2 as current content in textbox
            switch (operator) and perform corresponding operation on num1,num2.
            Store into result
            Print result in textbox
            Assign num1 as result
        5. if e is clear button
            Clear textbox
        6. if e is delete button
            Remove last character into textbox
        7. if e is negative button
            Insert negative of current content in textbox
5.  Inside main
        Call constructor Calculator
6.  End
```

Code

```
/*
Name          :  ASHIS SOLOMON
Roll Number :  MDL20CS035
Class           :  CS3B

* Program to create a simple calculator using java swing.
*/
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class Calculator implements ActionListener {

    JFrame frame;
    JTextField textfield;
    JButton[] numberButtons = new JButton[10];
    JButton[] functionButtons = new JButton[9];
    JButton addButton, subButton, mulButton, divButton;
    JButton decButton, equButton, delButton, clrButton, negButton;
    JPanel panel;

    Font myFont = new Font("Calibri", Font.BOLD, 30);

    double num1 = 0, num2 = 0, result = 0;
    char operator;

    Calculator() {

        frame = new JFrame("Calculator");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(420, 550);
        frame.setLayout(null);
```

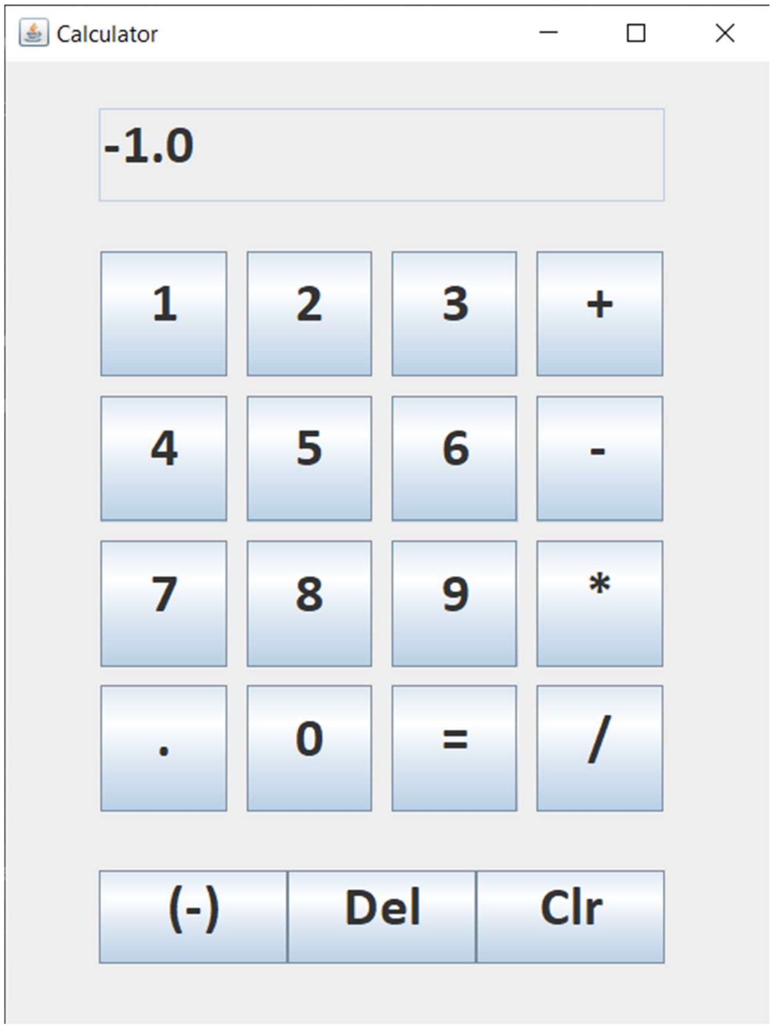
```
textfield = new JTextField();
textfield.setBounds(50, 25, 300, 50);
textfield.setFont(myFont);
textfield.setEditable(false);

functionButtons[0] = addButton = new JButton("+");
functionButtons[1] = subButton = new JButton("-");
functionButtons[2] = mulButton = new JButton("*");
functionButtons[3] = divButton = new JButton("/");
functionButtons[4] = decButton = new JButton(".");
functionButtons[5] = equButton = new JButton("=");
functionButtons[6] = delButton = new JButton("Del");
functionButtons[7] = clrButton = new JButton("Clr");
functionButtons[8] = negButton = new JButton("(-)");

for (int i = 0; i < 9; i++) {
   functionButtons[i].addActionListener(this);
   functionButtons[i].setFont(myFont);
   functionButtons[i].setFocusable(false);
}

for (int i = 0; i < 10; i++) {
   numberButtons[i] = new JButton(String.valueOf(i));
   numberButtons[i].addActionListener(this);
   numberButtons[i].setFont(myFont);
   numberButtons[i].setFocusable(false);
}

negButton.setBounds(50, 430, 100, 50);
delButton.setBounds(150, 430, 100, 50);
clrButton.setBounds(250, 430, 100, 50);

panel = new JPanel();
panel.setBounds(50, 100, 300, 300);
```

```
    panel.setLayout(new GridLayout(4, 4, 10, 10));

    panel.add(numberButtons[1]);
    panel.add(numberButtons[2]);
    panel.add(numberButtons[3]);
    panel.add(addButton);
    panel.add(numberButtons[4]);
    panel.add(numberButtons[5]);
    panel.add(numberButtons[6]);
    panel.add(subButton);
    panel.add(numberButtons[7]);
    panel.add(numberButtons[8]);
    panel.add(numberButtons[9]);
    panel.add(mulButton);
    panel.add(decButton);
    panel.add(numberButtons[0]);
    panel.add(equButton);
    panel.add(divButton);

    frame.add(panel);
    frame.add(negButton);
    frame.add(delButton);
    frame.add(clrButton);
    frame.add(textfield);
    frame.setVisible(true);
}

public static void main(String[] args) {

    new Calculator();
}
```

```
@Override
public void actionPerformed(ActionEvent e) {

    for (int i = 0; i < 10; i++) {
        if (e.getSource() == numberButtons[i]) {
            textfield.setText(textfield.getText().concat(String.valueOf(i)));
        }
    }
    if (e.getSource() == decButton) {
        textfield.setText(textfield.getText().concat("."));
    }
    if (e.getSource() == addButton) {
        num1 = Double.parseDouble(textfield.getText());
        operator = '+';
        textfield.setText("");
    }
    if (e.getSource() == subButton) {
        num1 = Double.parseDouble(textfield.getText());
        operator = '-';
        textfield.setText("");
    }
    if (e.getSource() == mulButton) {
        num1 = Double.parseDouble(textfield.getText());
        operator = '*';
        textfield.setText("");
    }
    if (e.getSource() == divButton) {
        num1 = Double.parseDouble(textfield.getText());
        operator = '/';
        textfield.setText("");
    }
    if (e.getSource() == equButton) {
        num2 = Double.parseDouble(textfield.getText());
```

```
            switch (operator) {
            case '+':
                result = num1 + num2;
                break;
            case '-':
                result = num1 - num2;
                break;
            case '*':
                result = num1 * num2;
                break;
            case '/':
                result = num1 / num2;
                break;
            }
            textfield.setText(String.valueOf(result));
            num1 = result;
        }
        if (e.getSource() == clrButton) {
            textfield.setText("");
        }
        if (e.getSource() == delButton) {
            String string = textfield.getText();
            textfield.setText("");
            for (int i = 0; i < string.length() - 1; i++) {
                textfield.setText(textfield.getText() + string.charAt(i));
            }
        }
        if (e.getSource() == negButton) {
            double temp = Double.parseDouble(textfield.getText());
            temp *= -1;
            textfield.setText(String.valueOf(temp));
        }
    }
}
```

Output



Experiment Number: 6b **Date: 04th March 2022**

Graphics Programming

TRAFFIC LIGHT

Aim:

To write a java program that simulates a traffic light using SWING.

Algorithm:

1. Start
2. Import java.awt, java.awt.event and java.swing packages
3. Create a JFrame named frame
4. Create necessary radio buttons within ButtonGroup and add it to the frame
5. Set the size, layout, and visibility for the frame
6. Draw the traffic light model using 2D graphics
7. Add necessary Listeners and Events for the radio buttons
8. Inside the actionPerformed event handler add the functionality to change the background of the signal light to appropriate color
9. Stop

Code

```
/*
Name          :  ASHIS SOLOMON
Roll Number :  MDL20CS035
Class         :  CS3B

* Program to simulate traffic lights using java swing.
*/

import java.awt.*;

import java.awt.event.*;

import java.util.*;

import javax.swing.*;

public class trafficlight extends JFrame implements ItemListener {

   JRadioButton jr1;

   JRadioButton jr2;

   JRadioButton jr3;

   JTextField j1 = new JTextField(15);

   ButtonGroup b = new ButtonGroup();

   String msg = " ";

   int x = 0, y = 0, z = 0;


   public trafficlight(String msg) {

      super(msg);

      setLayout(new FlowLayout());

      jr1 = new JRadioButton("Red");

      jr2 = new JRadioButton("Yellow");

      jr3 = new JRadioButton("Green");
```

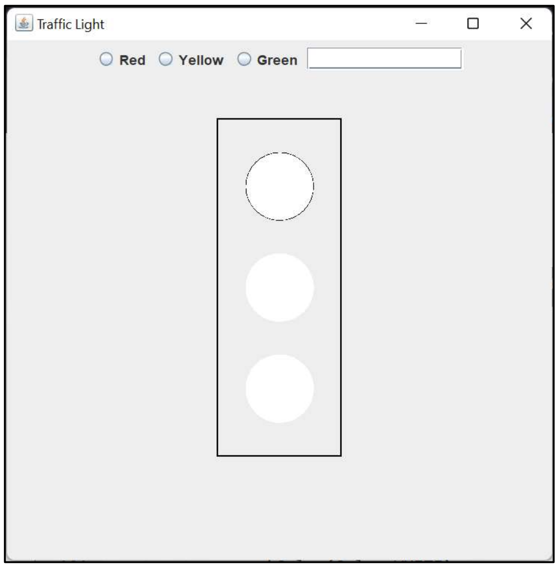
```
        jr1.addItemListener(this);
        jr2.addItemListener(this);
        jr3.addItemListener(this);

        add(jr1);
        add(jr2);
        add(jr3);
        b.add(jr1);
        b.add(jr2);
        b.add(jr3);
        add(j1);
        // addWindowListener(new WindowAdapter() {
        // public void windowClosing(WindowEvent e) {
        // System.exit(0);
        // }
        // });
    }

    public void itemStateChanged(ItemEvent ie) {
        if (ie.getSource() == jr1) {
            if (ie.getStateChange() == 1) {
                msg = "Stop!";
                x = 1;
                repaint();
            } else {
```

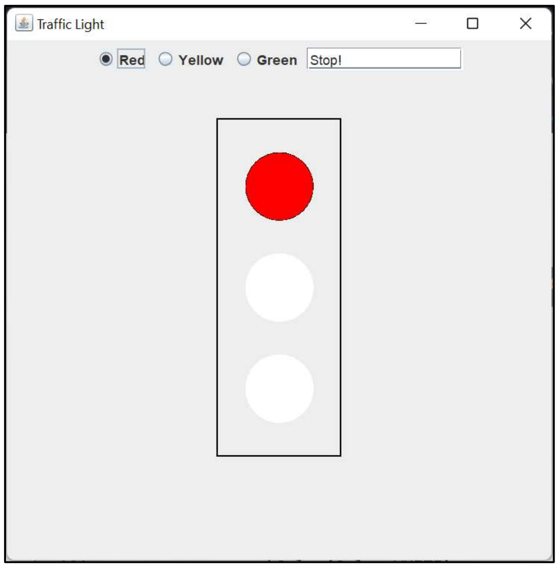
```
            msg = "";
        }
    }
    if (ie.getSource() == jr2) {
        if (ie.getStateChange() == 1) {
            msg = "Get Ready to go!";
            y = 1;
            repaint();
        } else {
            msg = "";
        }
    }
    if (ie.getSource() == jr3) {
        if (ie.getStateChange() == 1) {
            msg = "Go!!";
            z = 1;
            repaint();
        } else {
            msg = "";
        }
    }
    j1.setText(msg);
}
```

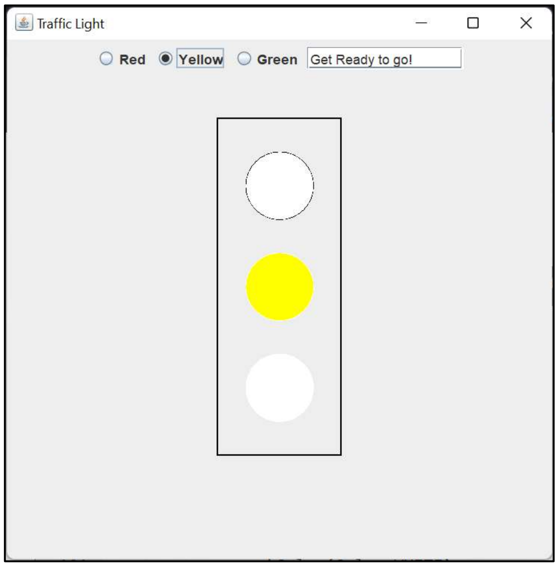
```
public void paint(Graphics g) {
    g.drawRect(195, 100, 110, 300);
    g.drawOval(220, 130, 60, 60);
    g.setColor(Color.WHITE);
    g.fillOval(220, 130, 60, 60);
    g.drawOval(220, 220, 60, 60);
    g.setColor(Color.WHITE);
    g.fillOval(220, 220, 60, 60);
    g.drawOval(220, 310, 60, 60);
    g.setColor(Color.WHITE);
    g.fillOval(220, 310, 60, 60);
    if (x == 1) {
        g.setColor(Color.RED);
        g.fillOval(220, 130, 60, 60);
        g.setColor(Color.WHITE);
        g.fillOval(220, 220, 60, 60);
        g.setColor(Color.WHITE);
        g.fillOval(220, 310, 60, 60);
        x = 0;
    }
    if (y == 1) {
        g.setColor(Color.WHITE);
        g.fillOval(220, 130, 60, 60);
        g.setColor(Color.YELLOW);
        g.fillOval(220, 220, 60, 60);
```

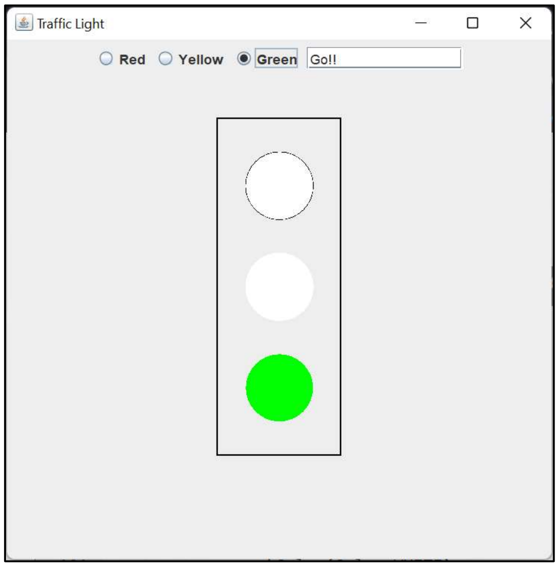
```
            g.setColor(Color.WHITE);
            g.fillOval(220, 310, 60, 60);
            y = 0;
        }
        if (z == 1) {
            g.setColor(Color.WHITE);
            g.fillOval(220, 130, 60, 60);
            g.setColor(Color.WHITE);
            g.fillOval(220, 220, 60, 60);
            g.setColor(Color.GREEN);
            g.fillOval(220, 310, 60, 60);
            z = 0;
        }
    }

    public static void main(String args[]) {
        JFrame jf = new trafficlight("Traffic Light");
        jf.setDefaultCloseOperation(EXIT_ON_CLOSE);
        jf.setSize(500, 500);
        jf.setVisible(true);
    }
}
```

Output


Traffic Light
Red
Yellow
Green


Traffic Light
Red
Yellow
Green
Stop!


Traffic Light
Red
Yellow
Green
Get Ready to go!


Traffic Light
Red
Yellow
Green
Go!!

Experiment Number: 6c **Date: 04th March 2022**

Graphics Programming

JDBC - DICTIONARY

Aim:

To write a java program to search and display the meaning of a word from a database using Java Database Connectivity (JDBC)

Algorithm:

1. Start
2. Import java.sql package
3. Create a class with file name mySqlCon which will have the main method
4. Inside the main method create a try block and enclose the statements for steps a to g in the try block
   a. Print "Enter word "
   b. Read the string from the user and store it in variable "str"
   c. Register a driver
   d. Create a connection for the database dbase already created in your system and create an instance "con" for it
   e. Create a statement using method createStatement and make an instance "stmt" for it.
   f. Create an instance for ResultSet "rs" and call method stmt.excecuteQuery "select from DICTIONARY where word=str " and assign it to rs
   g. Get the meaning using method rs.getString(1) and output it
   h. Close the connection by calling method con.close();
5. Create a catch block to catch the exception that might be thrown.
6. Stop

Code

```
/*
Name         :  ASHIS SOLOMON
Roll Number :  MDL20CS035
Class          :  CS3B

*Program to implement JDBC connectivity in java.
*/
import java.sql.*;
import java.util.Scanner;
class dictionary {
   public static void main(String arg[]) {
      try {
         int flag = 0;
         Scanner sc = new Scanner(System.in);
         System.out.print("\nEnter word : ");
         String str = sc.nextLine();
         Class.forName("com.mysql.cj.jdbc.Driver");
         Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/dbase",
"root", "root");
         Statement stmt = con.createStatement();
         ResultSet rs = stmt.executeQuery("select * from dictionary");
         while (rs.next()) {
            if (rs.getString(1).equals(str)) {
               System.out.println(rs.getString(1) + " - " + rs.getString(2));
               flag = 1;
               break;
            }
         }
         if (flag == 0) {
            System.out.println(str + " was not found");
         }
         con.close();
```

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

Output

```
java dictionary

Enter word : red

red - a primary colour



java dictionary

Enter word : blue

blue was not found
```