SOURCE CODE:

```
class welcome {
    public static void main(String[] args){
     System.out.print("WELCOME JAVA");
        }
}
```

OUTPUT:

```
WELCOME JAVA
```

Experiment No:1 Lab Cycle 1 Date: 26-02-2024

WELCOME JAVA

AIM:

Write a program to display the message “WELCOME JAVA”.

ALGORITHM:

1. Start
2. Define a class named welcome.
3. Inside the class, define a main method, which is the entry point of the program.
4. Display "WELCOME JAVA"
5. Stop

RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

```
class rectarea{
        double l,b;
        void setData(double length, double breadth){
                l = length;
                b = breadth;
        }
    double getArea(){
    return l*b;
    }
    public static void main(String[] args){
        rectarea r = new rectarea();
        r.setData(12.38,13);
        System.out.println("Area of rectangle = "+r.getArea());
    }
}
```

OUTPUT:

```
Area of rectangle = 160.94
```

Experiment No: 2 Lab Cycle 1 Date: 26-02-2024

RECTANGLE AREA

AIM:

Create a class Rectangle with instance variable length and breadth. Define a method setData for setting values of instance variables and a method getArea to return the area of Rectangle using the class. Find the area of the rectangle using the values length = 12.48 and breadth = 13.

ALGORITHM:

1. Define a class **rectarea** with instance variables **l** and **b**.
2. Create a method **setData** to set the length and breadth of the rectangle.
3. Define a method **getArea** to calculate and return the area of the rectangle.
4. In the **main** method:
    - Create an object **r** of class **rectarea**.
    - Set the length to **12.38** and breadth to **13** using **setData**.
    - Print "Area of rectangle = " followed by the result of **getArea** method.

RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

```
import java.util.Scanner;
class oddeven{
	public static void main(String[] args){
		int num;
		Scanner scanner = new Scanner(System.in);
		System.out.println("Enter number: ");
		num = scanner.nextInt();
		if(num%2==0){
			System.out.println(num+" is even!!");
		}
		else{
			System.out.println(num+" is odd!!");
		}
		scanner.close();
	}
}
```

OUTPUT:

```
Enter number:
5
5 is odd!!
```

Experiment No: 3 Lab Cycle 1 Date: 26-02-2024

ODD OR EVEN

AIM:

Write a program to read integer from keyboard and check whether the number is even or odd.

ALGORITHM:

1. Start the program.
2. Display prompt: "Read a number from the user: "
3. Read inputNumber from user
4. If inputNumber modulo 2 equals 0:

   Display: inputNumber + " is even!!"

   Else:

   Display: inputNumber + " is odd!!"
5. End the program.

RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

```
class product{
	String pcode,pname;
	float price;
	public product(String pcode,String pname,float price){
		this.pcode=pcode;
		this.pname=pname;
		this.price=price;
	}
}
public class productprice{
	public static void main(String[] args){
		product p1 = new product("01","IceCream",100);
		product p2 = new product("02","Biscuit",150);
		product p3 = new product("03","Cake",200);
		product lowp = p1;
		if(p2.price < lowp.price){
			lowp = p2;
		}
		if(p3.price < lowp.price){
			lowp = p3;
		}
		System.out.println("The Product with the lowest price:");
		System.out.println("Product Code: "+lowp.pcode);
		System.out.println("Product Name: "+lowp.pname);
		System.out.println("Product Price: "+lowp.price);
	}
}
```

OUTPUT:

```
The Product with the lowest price:
Product Code: 01
Product Name: IceCream
Product Price: 100.0
```

Experiment No: 4 Lab Cycle 1 Date: 04-03-2024

LOWEST PRICE PRODUCT

AIM:

Define a class 'product' with data members pcode , pname and price. Create three objects of the class and find the product having the lowest price.

ALGORITHM:

1. Define a class named 'product' with attributes: pcode (Product Code), pname (Product Name), and price (Price).
2. Define a constructor in the 'product' class to initialize the attributes.
3. Define a class named 'productprice' as the main class.
4. In the 'main' method of the 'productprice' class:
    - Create three instances of the 'product' class: p1, p2, and p3, with their respective details.
    - Assign 'p1' as the initial candidate for the product with the lowest price.
5. Check if the price of 'p2' is lower than the price of the current lowest priced product ('lowp'):
    - If true, assign 'p2' as the new 'lowp'.
6. Check if the price of 'p3' is lower than the price of the current lowest priced product ('lowp'):
    - If true, assign 'p3' as the new 'lowp'.
7. Display the product with the lowest price:
    - Output: "The Product with the lowest price:"
    - Output: "Product Code: " + lowp.pcode
    - Output: "Product Name: " + lowp.pname
    - Output: "Product Price: " + lowp.price

RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

```
import java.util.Scanner;
class matrixadd{
	public static void main(String[] args){
		Scanner scanner = new Scanner(System.in);
		System.out.println("Enter rows and column of the martices: ");
		int row = scanner.nextInt();
		int col = scanner.nextInt();
		int[][] matrix1 = new int[row][col];
		int[][] matrix2 = new int[row][col];
		System.out.println("Enter the elements of the first matrix: ");
		for(int i = 0; i<row; i++){
			for(int j = 0; j<col; j++){
				matrix1[i][j] = scanner.nextInt();
				}
		}
		System.out.println("Enter the elements of the second matrix: ");
		for(int i = 0; i<row; i++){
			for(int j = 0; j<col; j++){
				matrix2[i][j] = scanner.nextInt();
				}
		}
		int[][] summatrix = new int[row][col];
		for(int i = 0; i<row; i++){
			for(int j = 0; j<col; j++){
				summatrix[i][j] = matrix1[i][j] + matrix2[i][j];
				}
		}
		System.out.println("Sum of Matrices: ");
		for(int i = 0; i<row; i++){
			for(int j = 0; j<col; j++){
				System.out.print(summatrix[i][j]+" ");
				}
				System.out.print("\n");
		}
	}
}
```

OUTPUT:

```
Enter rows and column of the martices:   3    2
Enter the elements of the first matrix:      1  2  3  4  5  6
Enter the elements of the second matrix: 2  4  6  8  13  15
Sum of Matrices:
 3      6
 9     12
18     21
```

Experiment No: 5 Lab Cycle 1 Date: 04-03-2024

MATRIX ADDITION

AIM:

Read two matrices from the console and perform matrix addition.

ALGORITHM:

1. Display "Enter rows and column of the matrices: ".
2. Read row and col from the user.
3. Create matrix1[row][col] and matrix2[row][col].
4. Read the values to both matrices.
5. Loop through each index of the each matrix and read the elements one by one
8. Create sumMatrix[row][col].
9. Loop through each index of the each matrix and perform:

   sumMatrix[i][j] = matrix1[i][j] + matrix2[i][j].
10. Display "Sum of Matrices: ".
11. Loop through each index of the summatrix and print the elements one by one

RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

```
import java.util.Scanner;
class ComplexNumber{
	double real;
	double imaginary;
	public ComplexNumber(double real,double imaginary){
		this.real = real;
		this.imaginary = imaginary;
	}
	public ComplexNumber add(ComplexNumber other){
		double NewReal = this.real + other.real;
		double NewImaginary = this.imaginary + other.imaginary;
		return new ComplexNumber(NewReal, NewImaginary);
	}
	public String toString(){
		if(imaginary < 0){
			return real + " - " + (-imaginary) + "i";
		}
		else{
			return real + " + " + (imaginary) + "i";
		}
	}
}
public class complexadd{
	public static void main(String[] args){
		Scanner scanner = new Scanner(System.in);
		System.out.println("Enter the real and imaginary part of the first complex number:");
		double real1 = scanner.nextDouble();
		double imaginary1 = scanner.nextDouble();
		System.out.println("Enter the real and imaginary part of the second complex number:");
		double real2 = scanner.nextDouble();
		double imaginary2 = scanner.nextDouble();
		ComplexNumber num1 = new ComplexNumber(real1, imaginary1);
		ComplexNumber num2 = new ComplexNumber(real2, imaginary2);
		ComplexNumber sum = num1.add(num2);
		System.out.println("The sum of the two complex numbers is: " + sum);
	}
}
```

OUTPUT:

```
Enter the real and imaginary part of the first complex number:  4  5
Enter the real and imaginary part of the second complex number:  5  4
The sum of the two complex numbers is: 9.0 + 9.0i
```

Experiment No: 6 Lab Cycle 1 Date: 04-03-2024

COMPLEX NUMBER ADDITION

AIM:

Write a program to perform complex number addition.

ALGORITHM:

1. Request real and imaginary components of the first complex number from the user
2. Store the entered real part as real1
3. Store the entered imaginary part as imaginary1
4. Request real and imaginary components of the second complex number from the user
5. Store the entered real part as real2
6. Store the entered imaginary part as imaginary2
7. Compute the sum of the real parts:

   real_sum = real1 + real2
8. Compute the sum of the imaginary parts:

   imaginary_sum = imaginary1 + imaginary2
9. Display the sum of the two complex numbers as:

   real_sum + imaginary_sum*i

RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

```
import java.util.Scanner;
public class SymmetricMatrix {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter number of rows: ");
        int rows = scanner.nextInt();
        System.out.print("Enter number of columns: ");
        int cols = scanner.nextInt();
        int[][] matrix = new int[rows][cols];
        System.out.println("Enter the elements of the matrix:");
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                matrix[i][j] = scanner.nextInt();
            }
        }
        boolean symmetric = true;
        if (rows != cols) {
            symmetric = false;
        } else {
            for (int i = 0; i < rows; i++) {
                for (int j = 0; j < cols; j++) {
                    if (matrix[i][j] != matrix[j][i]) {
                        symmetric = false;
                        break;
                    }
                }
                if (!symmetric) {
                    break;
                }
            }
        }
        if (symmetric) {
            System.out.println("The matrix is symmetric.");
        } else {
            System.out.println("The matrix is not symmetric.");
        }
        scanner.close();
    }
}
```

OUTPUT:

Enter number of rows: 3

Enter number of columns: 3

Enter the elements of the matrix: 1 7 3 7 4 5 3 5 2

The matrix is symmetric.

Experiment No: 7 Lab Cycle 1 Date: 11-03-2024

SYMMETRIC MATRIX

AIM:

Read a matrix from the console and check whether it is symmetric/not

ALGORITHM:

1. Store rows and columns as rows and cols from the user
2. Create a 2D array called matrix with dimensions rows x cols
3. Display a message asking the user to enter the elements of the matrix
4. Iterate over each row from 0 to rows-1

   Iterate over each column from 0 to cols-1

   Read the integer input from the user and store it in the corresponding position in the matrix
5. Initialize a boolean variable symmetric to true
6. If the number of rows is not equal to the number of columns

   Set symmetric to false
7. Else, if the number of rows is equal to the number of columns

   Iterate over each row from 0 to rows-1

   a. Iterate over each column from 0 to cols-1

   If the element at matrix[i][j] is not equal to the element at matrix[j][i]

   - Set symmetric to false
   - Break out of the inner loop

   b. If symmetric is false

   Break out of the outer loop
8. If symmetric is true

   Display "The matrix is symmetric."
9. Else

   Display "The matrix is not symmetric."

RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

```
import java.util.Scanner;
public class leapyearprinter{
   public static void main (String[] args) {
      Scanner scanner = new Scanner(System.in);
      System.out.println("Enter Start Year:");
      int startYear = scanner.nextInt();
      System.out.println("Enter End Year:");
      int endYear = scanner.nextInt();
      System.out.println("Leap Years:");
      for(int year = startYear; year<=endYear; year++){
         if((year%4==0&&year%100!=0)||(year%400==0)){
            System.out.println(year);
         }
      }
   }
}
```

OUTPUT:

Enter Start Year:

2020

Enter End Year:

2030

Leap Years:

2020

2024

2028

Experiment No: 8 Lab Cycle 1 Date: 11-03-2024

LEAP YEARS

AIM:

Write a program to print the leap years within the given range

ALGORITHM:

1. Request the user to enter the starting year and store it as startYear
2. Request the user to enter the ending year and store it as endYear
3. Display "Leap Years:"
4. Iterate over each year from startYear to endYear (inclusive):
   If the current year is divisible by 4 and not divisible by 100, or if it is divisible by 400:
   Display the current year as a leap year

RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

```
import java.util.Scanner;
public class cpu {
   int price = 1000;
   public class processor {
      int cores = 6;
      String manufacturer = "Pentium";
   }
   void print() {
      processor pc = new processor();
      System.out.println("Processor:\nCores=" + pc.cores + "\nManufacturer = " + pc.manufacturer);
   }
   static class ram {
      static int memory = 64;
      static String manufacturer = "Intel";
   }
   public static void main(String[] args) {
      cpu cp = new cpu();
      cpu.ram rm = new cpu.ram();
      System.out.println("CPU: \nPrice: " + cp.price);
      cp.print();
      System.out.println("RAM: \n Memory=" + rm.memory + " Manufacturer=" + rm.manufacturer);
   }
}
```

OUTPUT:

CPU:

Price: 1000

Processor:

Cores=6

Manufacturer = Pentium

RAM:

Memory=64 Manufacturer=Intel

Experiment No: 9 Lab Cycle 1 Date: 11-03-2024

CPU

AIM:

Create CPU with attribute price. Create an inner class processor (no of cores, manufacturer) and static nested class RAM (memory, manufacturer). Create an object of CPU and print information of processor and RAM.

ALGORITHM:

1. Define a class named "cpu" with an instance variable "price" set to 1000.
2. Define an inner class named "processor" within the "cpu" class, with instance variables "cores" set to 6 and "manufacturer" set to "Pentium".
3. Define a method named "print" within the "cpu" class to print the details of the processor.
    a. Instantiate a "processor" object.
    b. Print the number of cores and the manufacturer of the processor.
4. Define a static inner class named "ram" within the "cpu" class, with a static variable "memory" set to 64 and a static variable "manufacturer" set to "Intel".
5. In the main method:
    a. Instantiate a "cpu" object.
    b. Instantiate a "ram" object.
    c. Print the price of the CPU.
    d. Call the "print" method of the "cpu" object to print the details of the processor.
    e. Print the memory and manufacturer of the RAM.

RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

```
import java.util.Scanner;
class calc_mark{
	public static void main(String[] args){
		Scanner scanner = new Scanner(System.in);
		System.out.println("Enter the number of subjects: ");

		int total_subjects = scanner.nextInt();
		int[] marks = new int[total_subjects];
		int total_marks = 0;
		float percentage;

		System.out.println("Enter the marks one by one : ");
		for(int i = 0; i<total_subjects; i++){
				marks[i] = scanner.nextInt();
		}

		for(int i = 0; i<total_subjects; i++){
				total_marks = total_marks + marks[i];
		}
		percentage = (total_marks / (float) (total_subjects * 100)) * 100;
		System.out.println("Total marks: " + total_marks);
		System.out.println("Percentage: " + percentage);

	}
}
```

OUTPUT:

Enter the number of subjects:

4

Enter the marks one by one:

99

98

91

90

Total marks: 378

Percentage: 94.5

Experiment No: 10 Lab Cycle 2 Date: 18-03-2024

STUDENTS MARK ARRAY

AIM:

Write a program which accepts the marks of a student into a 1D array from the keyboard. Calculate and display total marks & percentage obtained by the student.

ALGORITHM:

1. Define a class named "calc_mark".
2. Create a Scanner object named "scanner" to read user input.
3. Prompt the user to enter the number of subjects.
4. Read the total number of subjects entered by the user and store it as "total_subjects".
5. Create an integer array named "marks" with size equal to "total_subjects" to store the marks of each subject.
6. Declare integer variable "total_marks" and float variable "percentage".
7. Prompt the user to enter the marks for each subject one by one.
8. Iterate from 0 to "total_subjects - 1":

   Read the marks for each subject and store them in the "marks" array.

9. Calculate the total marks by iterating over the "marks" array and summing up all the marks.
10. Calculate the percentage by dividing the total marks by (total_subjects * 100), ensuring one operand is float for a floating-point result, then multiply by 100.
11. Print the total marks and percentage.

RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

```
import java.util.Arrays;
import java.util.Scanner;
public class sort_string {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter the number of Strings: ");
        int nos = scanner.nextInt();
        scanner.nextLine();

        String[] strings = new String[nos];
        System.out.println("Enter the Strings: ");

        for (int i = 0; i < nos; i++) {
            strings[i] = scanner.nextLine();
        }
        Arrays.sort(strings);
        System.out.println("\nSorted Strings: ");
        for (String str : strings) {
            System.out.println(str);
        }
        scanner.close();
    }
}
```

OUTPUT:

```
Enter the number of Strings:
5
Enter the Strings:
Hanna
Rishi
Alan
Bhagya
Arun
Sorted Strings:
Alan
Arun
Bhagya
Hanna
Rishi
```

Experiment No: 11 Lab Cycle 2 Date: 18-03-2024

SORT STRINGS

AIM:

Program to sort strings.

ALGORITHM:

1. Read the number of strings to be sorted from user and store it as "nos".
2. Create a string array "strings" of size "nos".
3. Request the user to enter each string and store them in the "strings" array.
    a. Repeat the process "nos" times:
        i. Read a string from the user and store it in the current index of the "strings" array.
4. Sort the strings in the "strings" array using the Arrays.sort() method.
5. Print a message indicating the sorted strings.
6. Iterate through the sorted "strings" array and print each string on a new line.

RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

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

public class sort_char {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      System.out.println("Enter the String: ");
         String str = scanner.nextLine();
      str = str.toLowerCase();
      char[] chars = str.toCharArray();

      Arrays.sort(chars);
      String sorted = new String(chars);

      System.out.println("\nSorted characters of the string: "+ sorted);

      scanner.close();
   }
}
```

OUTPUT:

```
Enter the String:
Thiruvananthapuram

Sorted characters of the string: aaaahhimnnprrttuuv
```

Experiment No: 12 Lab Cycle 2 Date: 18-03-2024

SORT CHARACTER

AIM:

Program to sort characters from a string.

ALGORITHM:

1. Read the number of strings to be sorted from user and store it as "nos".
2. Create a string array "strings" of size "nos".
3. Request the user to enter each string and store them in the "strings" array.
    a. Repeat the process "nos" times:
        i. Read a string from the user and store it in the current index of the "strings" array.
4. Sort the strings in the "strings" array using the Arrays.sort() method.
5. Print a message indicating the sorted strings.
6. Iterate through the sorted "strings" array and print each string on a new line.

RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

```
import java.util.Scanner;
public class search{
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      System.out.println("Enter the number of elements: ");
         int n = scanner.nextInt();
         int[] ar = new int[n];
         int flag =0;
         System.out.println("Enter the elements: ");
         for(int i = 0; i<n; i++){
                  ar[i] = scanner.nextInt();
         }
         System.out.println("Enter the element to be searched: ");
      int e = scanner.nextInt();
      for(int i=0;i<n;i++){
                  if(ar[i]==e){
                           flag++;
                           break;
                  }
         }
         if(flag!=0){
                  System.out.println(e + " is found!!");
         }
         else
         {
                  System.out.println(e + " is NOT found!!");
         }
      scanner.close();
   }
}
```

OUTPUT:

Enter the number of elements:

5

Enter the elements: 6 9 4 3 8

Enter the element to be searched:

4

4 is found!!

Experiment No: 13 Lab Cycle 2 Date: 01-04-2024

ARRAY SEARCH

AIM:

Search an element in an array.

ALGORITHM:

1. Prompt the user to enter the number of elements
2. Read the entered value and store it as n
3. Create an array called ar of size n to store the elements
4. Initialize a variable named flag to 0
5. Prompt the user to enter the elements of the array
6. Read each element entered by the user and store them in the array ar
7. Prompt the user to enter the element to be searched and store it as e
8. Iterate over each element in the array ar
    a. If the current element is equal to e:
        i. Increment the value of flag by 1
        ii. Break out of the loop
9. If flag is not equal to 0:
    a. Display "e is found!!"
10. Else:
    a. Display "e is NOT found!!"

RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

```
public class StringManipulation {
    public static void main(String[] args) {
        String str = "Hello, World!";

        int length = str.length();
        System.out.println("Length of the string: " + length);

        String upperCaseStr = str.toUpperCase();
        System.out.println("Uppercase string: " + upperCaseStr);

        boolean containsWorld = str.contains("World");
        System.out.println("Does the string contain 'World'?: " + containsWorld);

        String[] splitStr = str.split(",");
        System.out.println("Split string: ");
        for (String s : splitStr) {
            System.out.println(s.trim()); // trim to remove leading/trailing spaces
        }

        StringBuffer stringBuffer = new StringBuffer();

        stringBuffer.append("Hello");
        stringBuffer.append(" ");
        stringBuffer.append("World");
        System.out.println("StringBuffer after appending: " + stringBuffer);

        stringBuffer.insert(5, ", ");
        System.out.println("StringBuffer after insertion: " + stringBuffer);

        stringBuffer.reverse();
        System.out.println("Reversed StringBuffer: " + stringBuffer);

        stringBuffer.delete(5, 8);
        System.out.println("StringBuffer after deletion: " + stringBuffer);
    }
}
```

OUTPUT:

Length of the string: 13

Uppercase string: HELLO, WORLD!

Does the string contain 'World'?: true

Split string:

Hello

World!

StringBuffer after appending: Hello World

StringBuffer after insertion: Hello, World

Reversed StringBuffer: dlroW ,olleH

StringBuffer after deletion: dlroWolleH

Experiment No: 14 Lab Cycle 2 Date: 01-04-2024

STRING MANIPULATION

AIM:

Perform string manipulation (using Built-in methods of String Class and StringBuffer Class)

ALGORITHM:

a. Initialize a string variable "str" with the value "Hello, World!".
b. Calculate the length of the string and store it in the variable "length".
c. Print the length of the string.
d. Convert the string to uppercase and store it in "upperCaseStr".
e. Print the uppercase string.
f. Check if the string contains the substring "World" and store the result in "containsWorld".
g. Print whether the string contains "World" or not.
h. Split the string by commas and store the substrings in the array "splitStr".
i. Print each substring after removing leading and trailing spaces.
j. Create a StringBuffer object named "stringBuffer".
k. Append "Hello" to the StringBuffer.
l. Append a space to the StringBuffer.
m. Append "World" to the StringBuffer.
n. Print the StringBuffer after appending.
o. Insert ", " at index 5 in the StringBuffer.
p. Print the StringBuffer after insertion.
q. Reverse the StringBuffer.
r. Print the reversed StringBuffer.
s. Delete characters from index 5 to 8 in the StringBuffer.
t. Print the StringBuffer after deletion.

RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

```
import java.util.Scanner;

class Employee {
    int eNo;
    String eName;
    double salary;

    public Employee(int eNo, String eName, double salary) {
        this.eNo = eNo;
        this.eName = eName;
        this.salary = salary;
    }
}

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

        System.out.print("Enter the number of employees: ");
        int n = scanner.nextInt();

        Employee[] employees = new Employee[n];

        for (int i = 0; i < n; i++) {
            System.out.println("\nEnter details for employee " + (i + 1) + ":");
            System.out.print("Employee Number: ");
            int eNo = scanner.nextInt();
            System.out.print("Employee Name: ");
            String eName = scanner.next();
            System.out.print("Employee Salary: ");
            double salary = scanner.nextDouble();

            employees[i] = new Employee(eNo, eName, salary);
        }

        System.out.print("\nEnter the employee number to search: ");
        int searchEno = scanner.nextInt();
        int flag = 0;
        for (Employee emp : employees) {
            if (emp.eNo == searchEno) {
                System.out.println("Employee found:");
                System.out.println("Employee Number: " + emp.eNo);
                System.out.println("Employee Name: " + emp.eName);
```

```
            System.out.println("Employee Salary: " + emp.salary);
            flag = 1;
            break;
        }
    }
    if (flag == 0) {
        System.out.println("Employee with Employee Number " + searchEno + " not found.");
    }

    scanner.close();
  }
}
```

OUTPUT:

Enter the number of employees: 2

Enter details for employee 1:

Employee Number: 221

Employee Name: Arjun

Employee Salary: 50000

Enter details for employee 2:

Employee Number: 222

Employee Name: Kamal

Employee Salary: 75000

Enter the employee number to search: 222

Employee found:

Employee Number: 222

Employee Name: Kamal

Employee Salary: 75000.0

Experiment No: 15 Lab Cycle 2 Date: 01-04-2024

EMPLOYEE SEARCH

AIM:

Program to create a class for Employee having attributes eNo, eName,Salary. Read n employee information and search for an employee given eNo using the concept of array of objects.

ALGORITHM:

1. Define a class named "Employee" with instance variables eNo, eName, and salary.
2. Define a constructor in the "Employee" class to initialize the instance variables.
3. Define a class named "EmployeeSearch" with a main method.
4. Inside the main method:
    a. Create a Scanner object to read user input.
    b. Prompt the user to enter the number of employees and store it as 'n'.
    c. Create an array of Employee objects named 'employees' with size 'n'.
    d. Iterate 'n' times:

i. Prompt the user to enter details for each employee, including employee number, name, and salary.

1. ii. Create a new Employee object with the entered details and store it in the 'employees' array.
    e. Prompt the user to enter the employee number to search and store it as 'searchEno'.
    f. Initialize a flag variable to 0.
    g. Iterate over each employee in the 'employees' array:
    h. If the employee number matches the 'searchEno':
        A. Print the details of the employee.
        B. Set the flag to 1.
        C. Break out of the loop.
    i. If the flag is still 0:
    j. Print a message indicating that the employee with the specified number was not found.
    i. Close the Scanner object.

RESULT : Program is executed successfully and output is obtained.

SOURCE CODE:

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

    public double calculateArea(double length, double width) {
        return length * width;
    }

    public double calculateArea(double radius) {
        return Math.PI * radius * radius;
    }

    public double calculateArea(int side) {
        return side * side;
    }

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

        System.out.println("Enter the length and width of the rectangle:");
        double length = scanner.nextDouble();
        double width = scanner.nextDouble();
        double rectangleArea = calculator.calculateArea(length, width);
        System.out.println("Area of Rectangle: " + rectangleArea);
        System.out.println("Enter the radius of the circle:");
        double radius = scanner.nextDouble();
        double circleArea = calculator.calculateArea(radius);
        System.out.println("Area of Circle: " + circleArea);
        System.out.println("Enter the side length of the square:");
        int side = scanner.nextInt();
```

Experiment No: 16 Lab Cycle 3 Date: 08-04-2024

METHOD OVERLOADING

AIM:

Program to find the area of different shapes rectangle, circle and square using the concept of method overloading.

ALGORITHM:

1. Define the AreaCalculator Class:

   Overload the calculateArea Method:

   a. calculateArea(double length, double width): Calculates area of a rectangle.

   b. calculateArea(double radius): Calculates area of a circle.

   c. calculateArea(int side): Calculates area of a square.

2. Inside the main method:

   a. Create a Scanner object to read user input.

   b. Create an AreaCalculator object

   c. Prompt user for dimensions and calculate areas using overloaded methods:

   d. Read length and width for a rectangle, call calculateArea(double, double), and print the result.

   e. Read radius for a circle, call calculateArea(double), and print the result.

   f. Read side length for a square, call calculateArea(int), and print the result.

```
        double squareArea = calculator.calculateArea(side);
        System.out.println("Area of Square: " + squareArea);
        scanner.close();
    }
}
```

OUTPUT:

```
Enter the length and width of the rectangle:
1
2
Area of Rectangle: 2.0
Enter the radius of the circle:
5
Area of Circle: 78.53981633974483
Enter the side length of the square:
6
Area Of Square: 36.0
```

RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

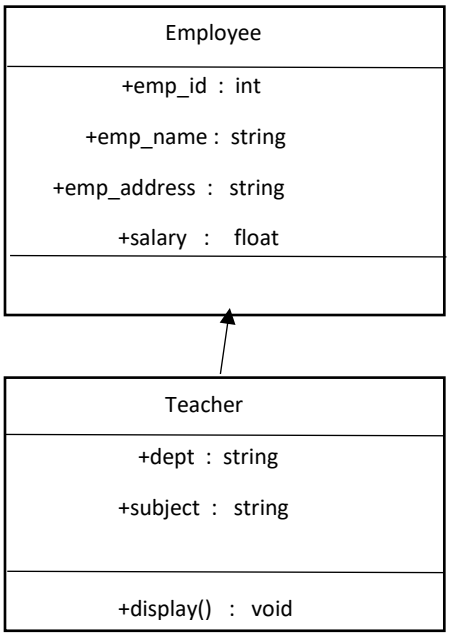
```
import java.util.Scanner;

class Employee {

    String empID, eName, address;

    double salary;

    public Employee(String empID, String eName, double salary, String address) {

        this.empID = empID;

        this.eName = eName;

        this.salary = salary;

        this.address = address;

    }

}

class Teacher extends Employee {

    String department;

    String[] subjectsTaught;

    public Teacher(String empID, String eName, double salary, String address, String department, String[]
subjectsTaught) {

        super(empID, eName, salary, address);

        this.department = department;

        this.subjectsTaught = subjectsTaught;

    }

    void display() {

        System.out.println("Employee ID: " + empID);

        System.out.println("Employee Name: " + eName);

        System.out.println("Salary: $" + salary);

        System.out.println("Address: " + address);

        System.out.println("Department: " + department);

        System.out.println("Subjects Taught:");

        for (String subject : subjectsTaught) {

            System.out.println(" - " + subject);

        }

        System.out.println();

    }
```

Experiment No: 17 Lab Cycle 3 Date: 08-04-2024

SIMPLE INHERITANCE

AIM:

Create a class 'Employee' with data members Empid, Name, Salary, Address and constructors to initialize the data members. Create another class 'Teacher' that inherit the properties of class employee and contain its own data members department, Subjects taught and constructors to initialize these data members and also include display function to display all the data members. Use array of objects to display details of N teachers.



ALGORITHM:

1. Define the Employee Class:
    a. Create the variables: empID, eName, salary, address.
    b. Then create a constructor to initialize the variable.
2. Inside the main method:
    a. Create a Scanner object to read user input.
    b. Prompt the user to enter the number of teachers.
    c. Create an array of Teacher objects with the size based on user input.
    d. Read empID, eName, salary, address, department and also the number of subjects taught of the Teacher.
    e. Create an object ‘Teacher’ and store the details of teachers.
    f. Repeat the steps d to e for each teacher.

```
}
public class Program17 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);


        System.out.print("Enter the number of teachers: ");
        int numTeachers = scanner.nextInt();
        scanner.nextLine(); // consume newline
        Teacher[] teachers = new Teacher[numTeachers];
        for (int i = 0; i < numTeachers; i++) {
            System.out.println("\nEnter the details of Teacher " + (i + 1) + ":");
            System.out.print("Employee ID: ");
            String empID = scanner.nextLine();
            System.out.print("Employee Name: ");
            String eName = scanner.nextLine();
            System.out.print("Salary: ");
            double salary = scanner.nextDouble();
            scanner.nextLine(); // consume newline
            System.out.print("Address: ");
            String address = scanner.nextLine();
            System.out.print("Department: ");
            String department = scanner.nextLine();
            System.out.print("Number of Subjects taught: ");
            int numSubjects = scanner.nextInt();
            scanner.nextLine(); // consume newline
            String[] subjectsTaught = new String[numSubjects];
            for (int j = 0; j < numSubjects; j++) {
                System.out.print("Subject " + (j + 1) + ": ");
                subjectsTaught[j] = scanner.nextLine();
            }
            teachers[i] = new Teacher(empID, eName, salary, address, department, subjectsTaught);
        }
```

g. Now display all details of all teachers using display().

h. Close the Scanner.

```
        System.out.println("\nDetails of all teachers:");
        for (Teacher teacher : teachers) {
            teacher.display();
        }
        scanner.close();
    }
}
```

OUTPUT:

```
Enter the number of teachers: 3

Enter the details of Teacher 1:
Employee ID: 45E
Employee Name: Aswany Santhosh
Salary: 100000
Address: Kalathoor House, Kurianoor P.O, Thiruvalla
Department: Computer Science
Number of Subjects taught: 2
Subject 1: DBMS
Subject 2: ACN

Enter the details of Teacher 2:
Employee ID: 46E
Employee Name: Hanna Kunjumon
Salary: 150000
Address: Kinaruvila Thekkathil, Venga P.O, Sasthamcotta
Department: Computer Science
Number of Subjects taught: 2
Subject 1: Advanced Data Structures
Subject 2: Mathematical Computing

Enter the details of Teacher 3:
Employee ID: 50F
Employee Name: Anju P Thomas
```

Salary: 20000

Address: Poovelil, Ponmala P.O, Thadiyoor

Department: Chemical

Number of Subjects taught: 1

Subject 1: Oil and Gas

Details of all teachers:

Employee ID: 45E

Employee Name: Aswany Santhosh

Salary: $100000.0

Address: Kalathoor House, Kurianoor P.O, Thiruvalla

Department: Computer Science

Subjects Taught:

- DBMS
- ACN

Employee ID: 46E

Employee Name: Hanna Kunjumon

Salary: $150000.0

Address: Kinaruvila Thekkathil, Venga P.O, Sasthamcotta

Department: Computer Science

Subjects Taught:

- Advanced Data Structures
- Mathematical Computing

Employee ID: 50F

Employee Name: Anju P Thomas

Salary: $20000.0

Address: Poovelil, Ponmala P.O, Thadiyoor

Department: Chemical

Subjects Taught:

- Oil and Gas

RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

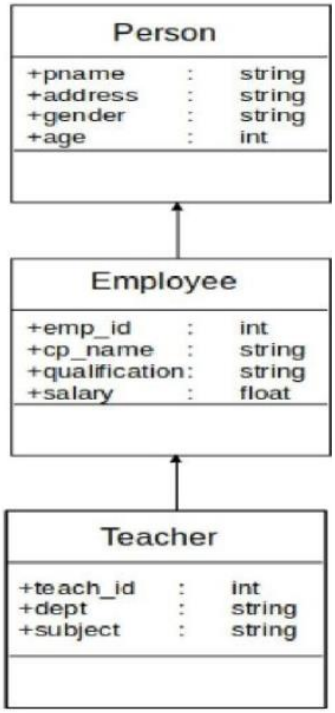
```
import java.util.Scanner;
class Person {
    String pname, address, gender;
    int age;
    Person(String pname, String address, String gender, int age) {
        this.pname = pname;
        this.address = address;
        this.gender = gender;
        this.age = age;
    }
}
class Employee extends Person {
    String emp_id, cp_name, qualification;
    float salary;
    Employee(String pname, String address, String gender, int age, String emp_id, String cp_name,
String qualification, float salary) {
        super(pname, address, gender, age);
        this.emp_id = emp_id;
        this.cp_name = cp_name;
        this.qualification = qualification;
        this.salary = salary;
    }
}
class Teacher extends Employee {
    int teach_id;
    String department, subject;
    Teacher(int teach_id, String department, String subject, String emp_id, String cp_name,
 String qualification, float salary, String pname, String address, String gender, int age) {
        super(pname, address, gender, age, emp_id, cp_name, qualification, salary);
        this.teach_id = teach_id;
        this.department = department;
```

MULTILEVEL INHERITANCE

AIM:

Create a class 'Person' with data members Name, Gender, Address, Age and a constructor to initialize the data members and another class 'Employee' that inherits the properties of class Person and also contains its own data members like Empid, Company name, Qualification, Salary and its own constructor.

Create another class 'Teacher' that inherits the properties of class Employee and contains its own data members like Subject, Department, Teacher id and also contain constructors and methods to display the data members. Use array of objects to display details of N teachers



```
        this.subject = subject;
    }
    public void display() {
        System.out.println("Name: " + pname);
        System.out.println("Gender: " + gender);
        System.out.println("Address: " + address);
        System.out.println("Age: " + age);
        System.out.println("Employee ID: " + emp_id);
        System.out.println("Company Name: " + cp_name);
        System.out.println("Salary: " + salary);
        System.out.println("Qualification: " + qualification);
        System.out.println("Teacher ID: " + teach_id);
        System.out.println("Department: " + department);
        System.out.println("Subject Taught: " + subject);
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter number of employees:");
        int n = sc.nextInt();
        sc.nextLine(); // Consume newline left-over
        Teacher[] empobj = new Teacher[n];
        for (int i = 0; i < n; i++) {
            System.out.println("Enter name of person:");
            String pname = sc.nextLine();
            System.out.println("Enter gender:");
            String gender = sc.nextLine();
            System.out.println("Enter address:");
            String address = sc.nextLine();
            System.out.println("Enter age:");
            int age = sc.nextInt();
            sc.nextLine(); // Consume newline left-over
            System.out.println("Enter employee ID:");
```

ALGORITHM:

1. Define the Person Class:
    a. Create the variables: pname, address, gender, age.
    b. Then create a constructor to initialize the variables.
2. Define the Employee Class (inherits from Person):
    a. Create the variables: empid, cp_name, qualification, salary.
    b. Then create a constructor to initialize the variables.
    c. Call the constructor of the Person class using **super()** to initialize inherited attributes.
3. Define the Teacher Class (inherits from Employee):
    a. Create the variables: teach_id, dept, subject.
    b. Then create a constructor to initialize the variables.
    c. Call the constructor of the Employee class using **super()** to initialize inherited attributes.
4. Inside the main method:
    a. Create a Scanner object to read user input.
    b. Prompt the user to enter the number of Employees.
    c. Read and store the values for pname, gender, address, age, emp_id, cp_name, salary, qualification, teach_id, department, and subject.
    d. Create an instance of the Teacher class using the collected data and store it in an array.
    e. Repeat the steps c to d for each teacher.
    f. Now display all details of all teachers using display().
    g. Close the Scanner.

```
            String emp_id = sc.nextLine();  // Changed to String
            System.out.println("Enter company name:");
            String cp_name = sc.nextLine();
            System.out.println("Enter salary:");
            float salary = sc.nextFloat();
            sc.nextLine(); // Consume newline left-over
            System.out.println("Enter qualification:");
            String qualification = sc.nextLine();
            System.out.println("Enter teacher ID:");
            int teach_id = sc.nextInt();
            sc.nextLine(); // Consume newline left-over
            System.out.println("Enter department:");
            String department = sc.nextLine();
            System.out.println("Enter subject taught:");
            String subject = sc.nextLine();
            empobj[i] = new Teacher(teach_id, department, subject, emp_id, cp_name, qualification, salary,
             pname, address, gender, age);
        }
        System.out.println("The details are:");
        for (int i = 0; i < n; i++) {
            empobj[i].display();
        }
    }
}
```

OUTPUT:

```
Enter number of employees: 3
Enter name of person: Arun
Enter gender: Male
Enter address: Sivajyothi Kovoor Arinalloor P.O Kollam
Enter age: 22
Enter employee ID: 714AD
Enter company name: TechCorp
```

Enter salary: 55000.5

Enter qualification: MSc

Enter teacher ID: 1001

Enter department: Physics

Enter subject taught: Quantum Mechanics

Enter name of person: Beena

Enter gender: Female

Enter address: MG Road, Ernakulam

Enter age: 29

Enter employee ID: 715BE

Enter company name: EduWorld

Enter salary: 60000.75

Enter qualification: PhD

Enter teacher ID: 1002

Enter department: Chemistry

Enter subject taught: Organic Chemistry

Enter name of person: Charles

Enter gender: Male

Enter address: Brigade Road, Bangalore

Enter age: 35

Enter employee ID: 716CH

Enter company name: LearnTech

Enter salary: 75000.0

Enter qualification: MPhil

Enter teacher ID: 1003

Enter department: Mathematics

Enter subject taught: Algebra

The details are:

Name: Arun

Gender: Male

Address: Sivajyothi Kovoor Arinalloor P.O Kollam

Age: 22

Employee ID: 714AD

Company Name: TechCorp

Salary: 55000.5

Qualification: MSc

Teacher ID: 1001

Department: Physics

Subject Taught: Quantum Mechanics

Name: Beena

Gender: Female

Address: MG Road, Ernakulam

Age: 29

Employee ID: 715BE

Company Name: EduWorld

Salary: 60000.75

Qualification: PhD

Teacher ID: 1002

Department: Chemistry

Subject Taught: Organic Chemistry

Name: Charles

Gender: Male

Address: Brigade Road, Bangalore

Age: 35

Employee ID: 716CH

Company Name: LearnTech

Salary: 75000.0

Qualification: MPhil

Teacher ID: 1003

Department: Mathematics

Subject Taught: Algebra

RESULT: Program is executed successfully and output is obtained.

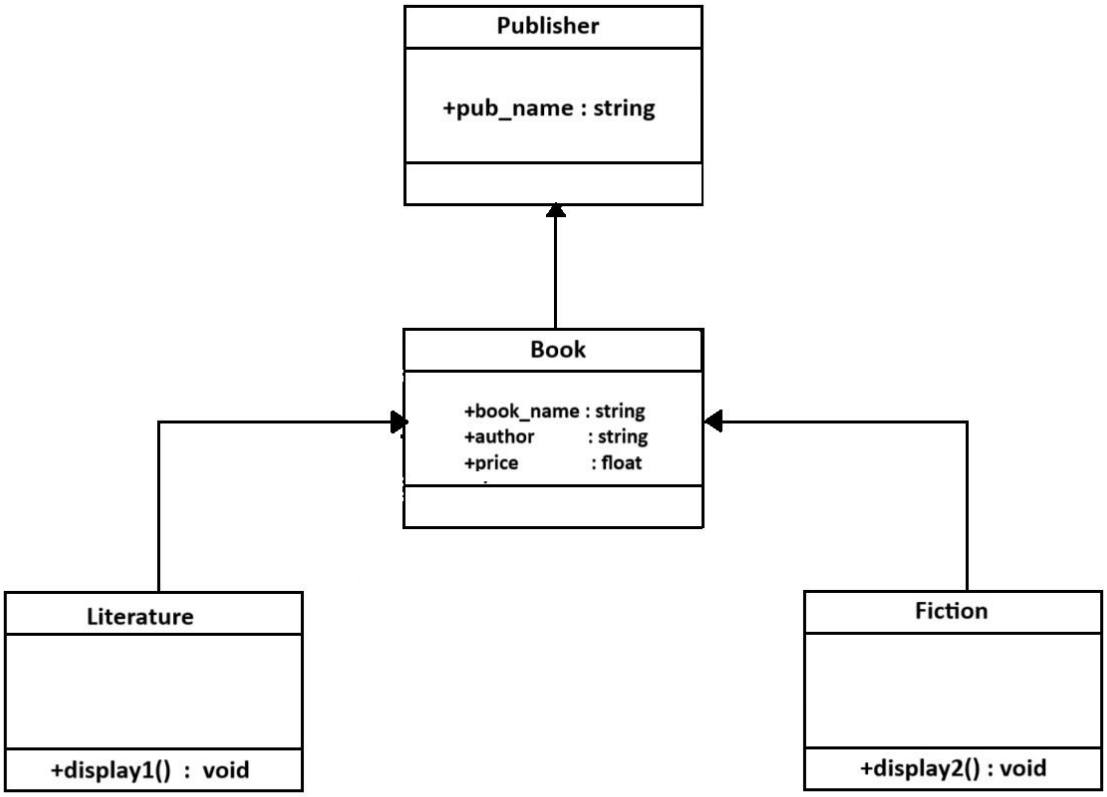
SOURCE CODE:

```
import java.util.Scanner;
class Publisher {
    protected String pub_name;
    public Publisher(String pub_name) {
        this.pub_name = pub_name;
    }
}
class Book extends Publisher {
    protected String book_name;
    protected String author;
    protected float price;
    public Book(String pub_name, String book_name, String author, float price) {
        super(pub_name);
        this.book_name = book_name;
        this.author = author;
        this.price = price;
    }
}
class Literature extends Book {
    public Literature(String pub_name, String book_name, String author, float price) {
        super(pub_name, book_name, author, price);
    }

    public void display1() {
        System.out.println("Literature Book Details:");
        System.out.println("Publisher: " + pub_name);
        System.out.println("Book Name: " + book_name);
        System.out.println("Author: " + author);
        System.out.println("Price: Rs" + price);
    }
}
```

HIERARCHIAL INHERITANCE

AIM:

Write a program having classes: Publisher, Book, Literature and Fiction. Read the information and print the details of books from either the category, using inheritance.



ALGORITHM:

1. Define the Publisher Class:
   a. Create the variable: pub_name.
   b. Then create a constructor to initialize the variable.
2. Define the Book class, inheriting from Publisher, with additional attributes book_name, author, and price.
3. Define the Literature class, inheriting from Book, with a method display1() to display literature book details.
4. Define the Fiction class, inheriting from Book, with a method display2() to display fiction book details.
5. Inside the main method:
   a. Create a Scanner object to read user input.

```
class Fiction extends Book {
    public Fiction(String pub_name, String book_name, String author, float price) {
        super(pub_name, book_name, author, price);
    }

    public void display2() {
        System.out.println("Fiction Book Details:");
        System.out.println("Publisher: " + pub_name);
        System.out.println("Book Name: " + book_name);
        System.out.println("Author: " + author);
        System.out.println("Price: Rs" + price);
    }
}
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter details for a Literature book:");
        System.out.print("Publisher: ");
        String litPubName = scanner.nextLine();
        System.out.print("Book Name: ");
        String litBookName = scanner.nextLine();
        System.out.print("Author: ");
        String litAuthor = scanner.nextLine();
        System.out.print("Price: ");
        float litPrice = scanner.nextFloat();
        scanner.nextLine();  // consume the newline
        Literature litBook = new Literature(litPubName, litBookName, litAuthor, litPrice);

        System.out.println("\nEnter details for a Fiction book:");
        System.out.print("Publisher: ");
```

b. Read details for a literature book.

c. Create a Literature object with the provided details.

d. Read details for a fiction book.

e. Create a Fiction object with the provided details.

f. Now display all details of Literature Book using display1().

g. Then display all the details of Fiction Book using display2().

h. Close the Scanner.

```
        String ficPubName = scanner.nextLine();
        System.out.print("Book Name: ");
        String ficBookName = scanner.nextLine();
        System.out.print("Author: ");
        String ficAuthor = scanner.nextLine();
        System.out.print("Price: ");
        float ficPrice = scanner.nextFloat();

        Fiction ficBook = new Fiction(ficPubName, ficBookName, ficAuthor, ficPrice);
        System.out.println();
        litBook.display1();

        System.out.println();
        ficBook.display2();
        scanner.close();
    }
}
```

OUTPUT:

```
Enter details for a Literature book:
Publisher: Penguin
Book Name: To Kill a Mockingbird
Author: Harper Lee
Price: 180          .99

Enter details for a Fiction book:
Publisher: Random House
Book Name: The Da Vinci Code
Author: Dan Brown
Price: 150.99

Literature Book Details:
Publisher: Penguin
Book Name: To Kill a Mockingbird
```

Author: Harper Lee

Price: Rs 180.99

Fiction Book Details:

Publisher: Random House

Book Name: The Da Vinci Code

Author: Dan Brown

Price: Rs 150.99

RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

```
import java.util.Scanner;
interface Shape {
    double area();
    double perimeter();
}
class Circle implements Shape {
    private double radius;
    public Circle(double radius) {
        this.radius = radius;
    }
    @Override
    public double area() {
        return Math.PI * radius * radius;
    }
    @Override
    public double perimeter() {
        return 2 * Math.PI * radius;
    }
}
class Rectangle implements Shape {
    private double length;
    private double width;
    public Rectangle(double length, double width) {
        this.length = length;
        this.width = width;
    }
    @Override
    public double area() {
        return length * width;
    }
    @Override
```

Experiment No: 20 Lab Cycle 3 Date: 15-04-2024

INTERFACE

AIM:

Create an interface having prototypes of functions area() and perimeter(). Create two classes Circle and Rectangle which implements the above interface. Create a menu driven program to find area and perimeter of objects.

```
    public double perimeter() {
        return 2 * (length + width);
    }
}
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        while (true) {
            System.out.println("Menu:");
            System.out.println("1. Circle");
            System.out.println("2. Rectangle");
            System.out.println("3. Exit");
            System.out.print("Choose an option: ");
            int choice = scanner.nextInt();
            switch (choice) {
                case 1:
                    handleCircle(scanner);
                    break;
                case 2:
                    handleRectangle(scanner);
                    break;
                case 3:
                    System.out.println("Exiting...");
                    scanner.close();
                    System.exit(0);
                default:
                    System.out.println("Invalid option. Please try again.");
            }
        }
    }
    private static void handleCircle(Scanner scanner) {
        System.out.print("Enter radius of the circle: ");
```

ALGORITHM:

1. Define an interface named Shape with methods area() and perimeter().
2. Create a class named Circle that implements the Shape interface together with the area() and perimeter() methods.
3. Similarly, define a class for Rectangle and implement the Shape interface and its methods.
4. Inside the main method:
    a. Create a Scanner object to read user input.
    b. Read the choice of the user by using a menu driven switch-based approach for reading the details of circle or rectangle.
    c. If it is Circle, create the object and find the area and perimeter of circle
    d. If it is Rectangle, create the object and find the area and perimeter of rectangle
    e. Repeat it until the user chooses to exit
    f. Close the Scanner.

```
        double radius = scanner.nextDouble();
        Circle circle = new Circle(radius);
        System.out.println("Area: " + circle.area());
        System.out.println("Perimeter: " + circle.perimeter());
    }
    private static void handleRectangle(Scanner scanner) {
        System.out.print("Enter length of the rectangle: ");
        double length = scanner.nextDouble();
        System.out.print("Enter width of the rectangle: ");
        double width = scanner.nextDouble();
        Rectangle rectangle = new Rectangle(length, width);
        System.out.println("Area: " + rectangle.area());
        System.out.println("Perimeter: " + rectangle.perimeter());
    }
}
```

OUTPUT:

```
Menu:
1. Circle
2. Rectangle
3. Exit
Choose an option: 1
Enter radius of the circle: 5
Area: 78.53981633974483
Perimeter: 31.41592653589793
Menu:
1. Circle
2. Rectangle
3. Exit
Choose an option: 2
Enter length of the rectangle: 6
Enter width of the rectangle: 3
Area: 18.0
Perimeter: 18.0
```

RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

```
import java.util.Scanner;
class Shapes {
   float a, b;
   Shapes(){}
   Shapes(float value) {
      a = value;
   }

   Shapes(float val1, float val2) {
      a = val1;
         b = val2;
   }

   double area() {
         System.out.println("Area of different Shapes:");
      return 0;
   }
}

class Rectangle extends Shapes{
         Rectangle(float a, float b){
                  super(a,b);
         }
         double area(){
                  return a*b;
         }
}

class Circle extends Shapes {
   Circle(float a) {
      super(a);
```

Experiment No: 21 Lab Cycle 3 Date: 15-04-2024

METHOD OVERRIDING

AIM:

Using the concept of method overriding, find the area of shapes Rectangle, Circle and Square.

ALGORITHM:

1. Define a base class Shapes with:
    a. Shapes(): Default constructor.
    b. Shapes(float value): Constructor with one parameter to initialize a.
    c. Shapes(float val1, float val2): Constructor with two parameters to initialize a and b.
    d. • Define method double area(): Prints "Area of different Shapes:" and returns 0.
2. Create a Rectangle class that extends Shapes with:
    a. constructor Rectangle(float a, float b).
    b. Call superclass constructor to initialize a and b
    c. Override method double area(): Returns the area of the rectangle as a * b.
3. Create a Circle class that extends Shapes with:
    a. constructor Circle(float a).
    b. Call superclass constructor to initialize a
    c. Override method double area(): Returns the area of the rectangle as Math.PI * a * a.
4. Create a Circle class that extends Shapes with:
    a. constructor Circle(float a).
    b. Call superclass constructor to initialize a
    c. Override method double area(): Returns the area of the square as a * a
5. Inside the main method:
    a. Create a Scanner object to read user input.
    b. Instantiate Shapes object and call area() method to display default message.
    c. Read the circle`s radius, and instantiate Circle object, calculate and display area.

```
    }

    double area() {
        return Math.PI * a * a;
    }
}

class Square extends Shapes {
    Square(float a) {
        super(a);
    }

    double area() {
        return a * a;
    }
}

public class FindArea {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Shapes obj = new Shapes();
            obj.area();
        System.out.print("Enter the radius of the circle: ");
        float r = sc.nextFloat();
        Circle obj1 = new Circle(r);
        System.out.println("Area of circle is: " + obj1.area());

        System.out.print("Enter the length of the rectangle: ");
        float l = sc.nextFloat();
        System.out.print("Enter the breadth of the rectangle: ");
        float b = sc.nextFloat();
        Rectangle obj2 = new Rectangle(l, b);
```

d. Read the rectangle`s length and breadth, and instantiate Rectangle object, calculate and display area.

e. Read the square`s side, and instantiate Square object, calculate and display area.

f. Close Scanner object.

```
        System.out.println("Area of rectangle is: " + obj2.area());


        System.out.print("Enter the side length of the square: ");
        float a = sc.nextFloat();
        Square obj3 = new Square(a);
        System.out.println("Area of square is: " + obj3.area());


        sc.close();
    }
}
```

OUTPUT:

```
Area of different Shapes:
Enter the radius of the circle: 5
Area of circle is: 78.53981633974483
Enter the length of the rectangle: 5
Enter the breadth of the rectangle: 6
Area of rectangle is: 30.0
Enter the side length of the square: 8
Area of square is: 64.0
```

RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

```
import java.util.Scanner;


abstract class Shape {
    public abstract void area();
}


class Rectangle extends Shape {
    @Override
    public void area() {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter length and breadth of rectangle:");
        int length = scanner.nextInt();
        int breadth = scanner.nextInt();
        int area = length * breadth;
        System.out.println("Area of the rectangle is: " + area);
    }
}


class Circle extends Shape {
    @Override
    public void area() {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter radius of circle:");
        double radius = scanner.nextDouble();
        double area = Math.PI * radius * radius;
        System.out.println("Area of the circle is: " + area);
    }
}


class Square extends Shape {
    @Override
```

Experiment No: 22 Lab Cycle 3 Date: 15-04-2024

ABSTRACT CLASS

AIM:

Create an Abstract Class 'Shape' with an abstract method find Area to find the area of different shapes. Create subclasses Rectangle, Circle and Square from Shape. Calculate and display area of Rectangle, Circle and Square.

ALGORITHM:

1. Define an abstract class Shape with an abstract method area().
2. Define concrete classes Rectangle, Circle, and Square, each extending the Shape class.
3. In Rectangle class, read the length and breadth of the rectangle from the user and calculate area of the rectangle as length*breadth and print the area.
4. In Circle class, read the radius of the circle from the user and calculate area of the rectangle as π * radius * radius (using Math.PI for π) and print the area.
5. In Square class, read the side of the square from the user and calculate area of the square as side*side and print the area.
6. Inside the main method:
    a. Create a Scanner object to read user input.
    b. Create instances of Rectangle, Circle, and Square classes.
    c. Call the area() method for each shape instance to calculate and display its area.

```
    public void area() {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter side of square:");
        int side = scanner.nextInt();
        int area = side * side;
        System.out.println("Area of the square is: " + area);
    }
}

public class AbstractClass {
    public static void main(String[] args) {
        Rectangle rect = new Rectangle();
        Circle cir = new Circle();
        Square sq = new Square();
        rect.area();
        cir.area();
        sq.area();
    }
}
```

OUTPUT:

```
Enter length and breadth of rectangle: 5 6
Area of the rectangle is: 30
Enter radius of circle: 3
Area of the circle is: 28.274333882308138
Enter side of square: 3
Area of the square is: 9
```

RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

Prgrm.java

```
import java.util.Scanner;
import Arithmetic.*;
public class Prgrm {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        int i = 1, ch;
        System.out.println("_____Program to demonstrate the basic arithmetic operations_____");
        System.out.println("Enter two numbers: ");
        float n1 = sc.nextFloat();
        float n2 = sc.nextFloat();

        while (i != 0) {
            System.out.println("1. Addition \t2. Subtraction \t3. Multiplication \t4. Division");
            System.out.print("Choose an option: ");
            ch = sc.nextInt();
            switch (ch) {
                case 1:
                    Addition obj1 = new Addition();
                    obj1.calculate(n1, n2);
                    break;
                case 2:
                    Subtraction obj2 = new Subtraction();
                    obj2.calculate(n1, n2);
                    break;
                case 3:
                    Multiplication obj3 = new Multiplication();
                    obj3.calculate(n1, n2);
                    break;
                case 4:
                    Division obj4 = new Division();
```

Experiment No: 23 Lab Cycle 4 Date: 22-04-2024

PACKAGES IN JAVA

AIM:

Create an Arithmetic package that has classes and interfaces for the 4 basic arithmetic operations. Test the package by implementing all operations on two given numbers.

ALGORITHM:

1. Create four files: Addition, Subtraction, Multiplication and Division.
2. Define an abstract class Shape with an abstract method area().
3. Define concrete classes Rectangle, Circle, and Square, each extending the Shape class.
4. Make sure each concrete class is in a separate file.
5. Also make sure all the files are in a common folder called ‘Arithmetic’
6. In Rectangle class, read the length and breadth of the rectangle from the user
7. Then calculate area of the rectangle as length*breadth and print the area.
8. In Circle class, read the radius of the circle from the user.
9. Calculate area of the rectangle as π * radius * radius (using Math.PI for π) and print the area.
10. In Square class, read the side of the square from the user
11. Calculate area of the square as side*side and print the area.
12. Create a file outside(Prgrm.java) outside of the Arithmetic folder as this file will have the main function and it will access the packages.
13. Inside the main method:
    a. Create a Scanner object to read user input.
    b. Use a menu driven setup for reading the values.
    c. Create instances of Rectangle, Circle, and Square classes.

```
                obj4.calculate(n1, n2);
                break;
            }
            System.out.print("Do you want to continue? (0: no, 1: yes) ");
            i = sc.nextInt();
        }
    }
}
```

Addition.java

```
package Arithmetic;
public class Addition {
    public void calculate(float a, float b) {
        System.out.println("Sum of the given numbers is: " + (a + b));
    }
}
```

Subtraction.java

```
package Arithmetic;
public class Subtraction {
    public void calculate(float a, float b) {
        System.out.println("Difference of the given numbers is: " + (a - b));
    }
}
```

Multiplication.java

```
package Arithmetic;
public class Multiplication {
    public void calculate(float a, float b) {
        System.out.println("Product of the given numbers is: " + (a * b));
    }
}
```

Division.java

```
package Arithmetic;
public class Division {
```

d. Call the area() method for each shape instance to calculate its area.

e. Print the area of each shape.

```
    public void calculate(float a, float b) {
        if (b != 0) {
            System.out.println("Quotient of the given numbers is: " + (a / b));
        } else {
            System.out.println("Cannot divide by zero.");
        }
    }
}
```

OUTPUT:

```
_____Program to demonstrate the basic arithmetic operations_____
Enter two numbers: 1  3
1. Addition     2. Subtraction  3. Multiplication       4. Division
Choose an option: 1
Sum of the given numbers is: 4.0
Do you want to continue? (0: no, 1: yes) 1
1. Addition     2. Subtraction  3. Multiplication       4. Division
Choose an option: 2
Difference of the given numbers is: -2.0
Do you want to continue? (0: no, 1: yes) 1
1. Addition     2. Subtraction  3. Multiplication       4. Division
Choose an option: 3
Product of the given numbers is: 3.0
Do you want to continue? (0: no, 1: yes) 1
1. Addition     2. Subtraction  3. Multiplication       4. Division
Choose an option: 4
Quotient of the given numbers is: 0.33333334
Do you want to continue? (0: no, 1: yes) 0
```

RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

LoginException.java

```
import java.util.Scanner;
class UsernameException extends Exception {
    public UsernameException(String msg) {
        super(msg);
    }
}

class PasswordException extends Exception {
    public PasswordException(String msg) {
        super(msg);
    }
}

public class LoginException {
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        String username, password;

        System.out.print("Enter username: ");
        username = s.nextLine();

        System.out.print("Enter password: ");
        password = s.nextLine();

        int length = username.length();

        try {
            if (length < 6) {
                throw new UsernameException("Username must be greater than 6 characters");
            } else if (!password.equals("abc@123")) {
```

Experiment No: 24 Lab Cycle 4 Date: 22-04-2024

EXCEPTION CLASS

AIM:

Write a user defined exception class to authenticate the username and password.

ALGORITHM:

1. Create two classes: UsernameException and PasswordException both inheriting the Exception class.
2. In the main method:
    a. Read username and password from the user.
    b. Get the length of the username.
    c. Use a try and catch to:
        - Check whether is less than 6 characters, if it's the throw a 'UsernameException'.
        - Check whether the password is not equal to 'abc@123', then thow a 'PasswordException'.
        - In the else part, display that the Authentication is successful.

```
            throw new PasswordException("Incorrect password\nType correct password");
        } else {
            System.out.println("Login Successful !!!");
        }
    } catch (UsernameException e) {
        System.out.println(e);
    } catch (PasswordException p) {
        System.out.println(p);
    }
  }
}
```

OUTPUT:

```
Enter username: arunad714
Enter password: 12345678
PasswordException: Incorrect password
Type correct password
Enter username: 12345678
Enter password: abc@123
Login Successful !!!
```

RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

AverageException.java

```
import java.util.Scanner;
class InputException extends Exception {
    public InputException(String msg) {
        super(msg);
    }
}
public class AverageException {
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        int sum = 0;
        System.out.print("Enter no of integers: ");
        int n = s.nextInt();
        int arr[] = new int[n];
        System.out.print("Enter the integers: ");
        try {
            for (int i = 0; i < n; i++) {
                int in = s.nextInt();
                if (in < 0) {
                    throw new InputException("Number is not positive\nType a positive number !!!");
                }
                arr[i] = in;
                sum += arr[i];
            }
            float avg = (float) sum / n; // Cast one operand to float to ensure floating point division
            System.out.println("Average is: " + avg);
        } catch (InputException e) {
            System.out.println(e);
        }
    }
}
```

Experiment No: 25 Lab Cycle 4 Date: 22-04-2024

AVERAGE EXCEPTION

AIM:

Find the average of N positive integers, raising a user defined exception for each negative input.

ALGORITHM:

1. Prompt the user to input the number of integers (`n`) they want to enter.
2. Read the value of `n` from the user.
3. Create an integer array `arr` of size `n` to store the integers entered by the user.
4. Initialize a variable `sum` to keep track of the sum of the entered integers.
5. Start a loop from 0 to `n-1` to read `n` integers from the user.
    - Read the integer input (`in`) from the user.
    - Check if the input integer is negative.
    - If the input is negative, throw an `InputException` with an appropriate error message
    - Otherwise, store the integer in the `arr` array and add it to the `sum`.
6. After the loop, calculate the average of the entered integers by dividing the sum by `n`. Ensure to cast one of the operands to `float` to perform floating-point division.
7. Print the calculated average.
8. End.

OUTPUT:

Enter no of integers: 5

Enter the integers: 3 6 9 -4

InputException: Number is not positive

Type a positive number !!!

Enter no of integers: 5

Enter the integers: 4 6 9 3 4

Average is: 5.2

RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

```
import java.util.Scanner;
class Multiplication extends Thread {
    @Override
    public void run() {
        System.out.println("Multiplication table of 5:");
        for (int i = 1; i <= 10; i++) {
            System.out.println(i + " X 5 = " + i * 5);
        }
        System.out.println("Exiting from Thread Multiplication...");
    }
}
class PrimeNumbers extends Thread {
    @Override
    public void run() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the value of n: ");
        int n = sc.nextInt();
        prime_N(n);
        sc.close();
    }
    static void prime_N(int N) {
        int x, y, flag;
        System.out.println("All the Prime numbers within 1 and " + N + " are:");
        for (x = 1; x <= N; x++) {
            if (x == 1 || x == 0) {
                continue;
            }
            flag = 1;
            for (y = 2; y <= x / 2; ++y) {
                if (x % y == 0) {
                    flag = 0;
```

```
                break;
            }
        }
        if (flag == 1) {
            System.out.print(x + "\t");
        }
    }
    System.out.println();
  }
}
public class ThreadClass {
  public static void main(String args[]) {
    Multiplication a = new Multiplication();
    PrimeNumbers b = new PrimeNumbers();
    a.start();
    b.start();
  }
}
```

OUTPUT:

```
Multiplication table of 5:
1 X 5 = 5
2 X 5 = 10
3 X 5 = 15
4 X 5 = 20
5 X 5 = 25
6 X 5 = 30
7 X 5 = 35
8 X 5 = 40
9 X 5 = 45
10 X 5 = 50
Exiting from Thread Multiplication...
Enter the value of n: 5
All the Prime numbers within 1 and 5 are:  2    3    5
```

Experiment No: 26 Lab Cycle 4 Date: 22-04-2024

THREADS IN JAVA

AIM:

Define 2 classes; one for generating multiplication table of 5 and other for displaying first N prime numbers. Implement using threads. (Thread class).

ALGORITHM:

1. Define the Multiplication Thread:
    a. In Start Method: The thread starts executing the run method.
    b. In Run Method:
    c. Print "Multiplication table of 5:".
    d. Loop through numbers from 1 to 10.
    e. For each number i, print i * 5.
    f. Print "Exiting from Thread Multiplication...".
2. Define the PrimeNumbers Thread:
    a. In Start Method: The thread starts executing the run method.
    b. In Run Method:
    c. Create a Scanner object to read input from the user.
    d. Read the integer value n.
    e. Call the prime_N method with n as the argument.
    f. Close the Scanner object.
    g. In prime_N Method:
    h. Print "All the Prime numbers within 1 and N are:".
    i. Use a flag variable for Prime checking
    j. Loop through numbers from 1 to N and inside it loop it from 2 to x / 2.
    k. If x is divisible by any of these numbers, set the flag to 0 and break the loop.
    l. If the flag is still 1, print x as it is a prime number.
    m. Print a newline for formatting.
3. Main Method in ThreadClass:

a. Create an instance of the Multiplication class.

b. Create an instance of the PrimeNumbers class.

c. Start both threads by calling their start methods.

RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

```
import javax.swing.*;
import java.awt.event.*;
import java.awt.*;
import java.awt.Graphics;
class MOUSEWINDOWEX extends JFrame implements MouseListener,WindowListener
{
 MOUSEWINDOWEX()
 {
 //setTitle(" Mouse Listener Java Swing Example");
 setBounds(100,200,500,500);
 setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
 setVisible(true);
 addMouseListener(this);
 addWindowListener(this);
 }
 public void mousePressed(MouseEvent e)
 {
Graphics g= getGraphics();
g.setColor(Color.BLUE);
g.fillOval(e.getX(),e.getY(),50,50);
 System.out.println("Mouse Pressed");
 }
 public void mouseReleased(MouseEvent e)
 {
 Graphics g= getGraphics();
g.setColor(Color.GREEN);
g.fillOval(e.getX(),e.getY(),50,50);
 System.out.println("Mouse Released");
 }
 public void mouseClicked(MouseEvent e)
 {
```

Experiment No: 27 Lab Cycle 5 Date: 29-04-2024

EVENT HANDLING IN JAVA

AIM:

Develop a program to handle all mouse and window events.

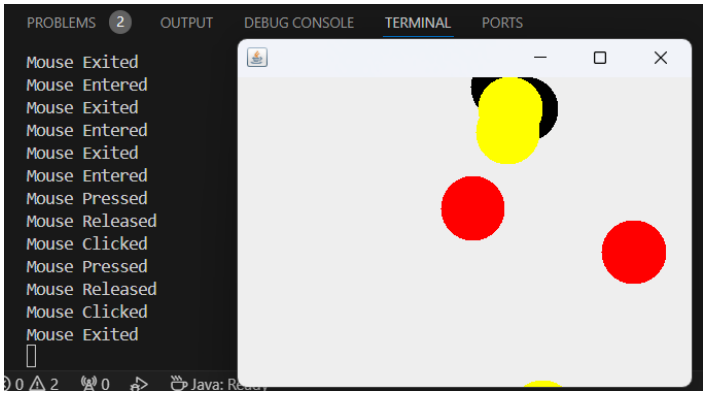
ALGORITHM:

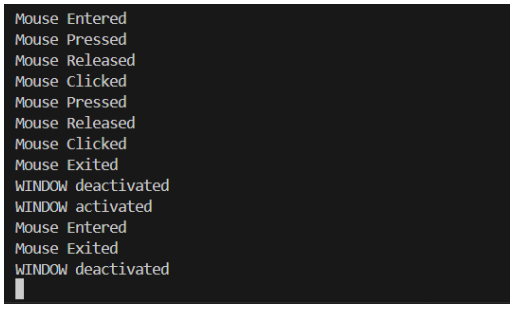
1. Create a JFrame: Instantiate a JFrame object to serve as the main window of your application.
2. Set Window Properties: Set the size, title, and default close operation for the JFrame.
3. Add Mouse Listener: Implement the MouseListener interface to handle mouse events such as press, release, click, enter, and exit. Add the mouse listener to the JFrame.
4. Add Window Listener: Implement the WindowListener interface to handle window events such as open, close, activate, deactivate, etc. Add the window listener to the JFrame.
5. Handle Mouse Events: Define methods to handle mouse events (mousePressed, mouseReleased, mouseClicked, mouseEntered, mouseExited). In each method, update the GUI (e.g., draw a circle of different colors) based on the specific mouse event.
6. Handle Window Events: Define methods to handle window events (windowActivated, windowClosed, windowClosing, windowDeactivated, windowDeiconified, windowIconified, windowOpened). Implement actions to be performed when each window event occurs.
7. Display the JFrame: Make the JFrame visible by calling the setVisible(true) method.

```
 Graphics g= getGraphics();
g.setColor(Color.RED);
g.fillOval(e.getX(),e.getY(),50,50);
 System.out.println("Mouse Clicked");
 }
 public void mouseEntered(MouseEvent e)
 {
Graphics g= getGraphics();
g.setColor(Color.YELLOW);
g.fillOval(e.getX(),e.getY(),50,50);
 System.out.println("Mouse Entered");
 }
 public void mouseExited(MouseEvent e)
 {
Graphics g= getGraphics();
g.setColor(Color.BLACK);
g.fillOval(e.getX(),e.getY(),50,50);
 System.out.println("Mouse Exited");
 }
public void windowActivated (WindowEvent e) {
 System.out.println("WINDOW activated");
}
public void windowClosed (WindowEvent e) {
 System.out.println("WINDOW closed");
}
public void windowClosing (WindowEvent e) {
System.out.println("WINDOW closing");
 dispose();
}
public void windowDeactivated (WindowEvent e) {
 System.out.println("WINDOW deactivated");
}
```

```
public void windowDeiconified (WindowEvent e) {
 System.out.println("WINDOW deiconified");
}

public void windowIconified(WindowEvent e) {
 System.out.println("WINDOW iconified");
}

public void windowOpened(WindowEvent e) {
 System.out.println("WINDOW opened");
}
}
 class MOUSEWINDOW
 {
 public static void main(String[] args)
 {
 MOUSEWINDOWEX frame = new MOUSEWINDOWEX();
 }
 }
```

OUTPUT:





RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

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

public class MaxOfThreeNumbers extends Frame implements ActionListener {
    TextField num1, num2, num3, result;
    Button findMax;

    public MaxOfThreeNumbers() {
        setLayout(new FlowLayout());

        num1 = new TextField(10);
        num2 = new TextField(10);
        num3 = new TextField(10);
        result = new TextField(10);
        result.setEditable(false);

        add(new Label("Number 1: "));
        add(num1);
        add(new Label("Number 2: "));
        add(num2);
        add(new Label("Number 3: "));
        add(num3);

        findMax = new Button("Find Max");
        add(findMax);
        add(new Label("Maximum: "));
        add(result);
        findMax.addActionListener(this);
        setTitle("Find Maximum of Three Numbers");
        setSize(250, 200);
        setVisible(true);
```

Experiment No: 28 Lab Cycle 5 Date: 29-04-2024

ABSTRACT WINDOW TOOLKIT

AIM:

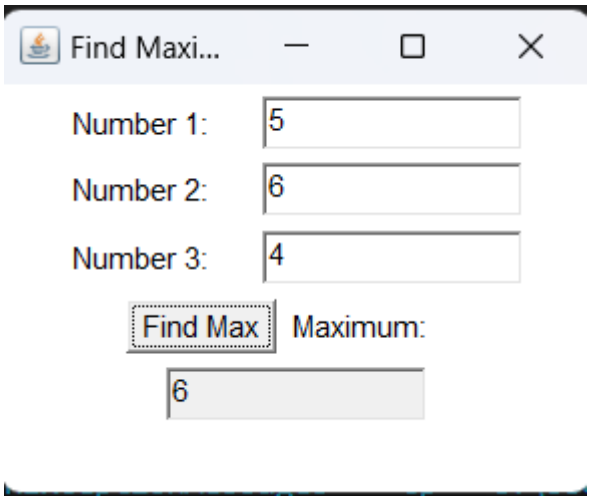
Program to find maximum of three numbers using AWT.

ALGORITHM:

1. Import required AWT classes.
2. Define a class named MaxOfThreeNumbers.
3. Inside the class:
    a. Declare variables: num1, num2, num3, result, findMax.
    b. Define a constructor to initialize the GUI components.
    c. Set layout to FlowLayout.
    d. Create TextFields for num1, num2, num3, and result.
    e. Create a Button for findMax.
    f. Add components to the frame.
    g. Add ActionListener to the findMax button.
    h. Set frame properties: title, size, and visibility
    i. Get input values from num1, num2, and num3 TextFields.
    j. Calculate the maximum of the three numbers using Math.max.
    k. Set the result TextField with the maximum value.
    l. Define main method to create an instance of MaxOfThreeNumbers.

```
    }

    public void actionPerformed(ActionEvent ae) {
        int number1 = Integer.parseInt(num1.getText());
        int number2 = Integer.parseInt(num2.getText());
        int number3 = Integer.parseInt(num3.getText());

        int max = Math.max(number1, Math.max(number2, number3));

        result.setText(String.valueOf(max));
    }

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

OUTPUT:



RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

```
import java.io.*;
class Readfile
{
public static void main(String arg[]) throws IOException
{
FileInputStream f= new FileInputStream("Sample.txt");
FileOutputStream f2= new FileOutputStream("cp.txt");
int c;
while((c=f.read())!=-1)
{
f2.write(c);
System.out.print((char)c);
}
f.close();
f2.close();
}
}
```

OUTPUT:

Sample.txt

this is a sample text

cp.txt

this is a sample text

Experiment No: 29 Lab Cycle 6 Date: 29-04-2024

FILE MANIPULATION IN JAVA

AIM:

Write a program to copy one file to another.

ALGORITHM:

1. Initialize Input and Output Streams:
    a. Create a FileInputStream object f to read from the file "Sample.txt".
    b. Create a FileOutputStream object f2 to write to the file "cp.txt".
2. Read and Write Loop:
    a. Initialize an integer variable c.
    b. Loop until the end of the input file is reached:
    c. Read a byte from the input file and store it in c.
    d. Check if c is not equal to -1 (end of file indicator).
    e. Write the byte stored in c to the output file.
    f. Print the character representation of the byte stored in c to the console.
3. Close Streams:
    a. Close the FileInputStream object f.
    b. Close the FileOutputStream object f2.

RESULT: Program is executed successfully and output is obtained.

SOURCE CODE:

```
class Readfile {
    public static void main(String arg[]) throws IOException {
        FileInputStream f = new FileInputStream("numbers.txt");
        FileOutputStream f2 = new FileOutputStream("odd.txt");
        FileOutputStream f3 = new FileOutputStream("even.txt");
        int c;
        while ((c = f.read()) != -1) {
            // Check if the ASCII value is even or odd
            if (c % 2 == 0) {
                f3.write(c); // Write to even.txt if ASCII value is even
            } else {
                f2.write(c); // Write to odd.txt if ASCII value is odd
            }
            // Print the character to console
            System.out.print((char) c);
        }
        f.close();
        f2.close();
        f3.close();
        System.out.println("\nContents of odd.txt:");
        try (FileInputStream oddFile = new FileInputStream("odd.txt")) {
            while ((c = oddFile.read()) != -1) {
                System.out.print((char) c);
            }
        }
        System.out.println("\nContents of even.txt:");
        try (FileInputStream evenFile = new FileInputStream("even.txt")) {
            while ((c = evenFile.read()) != -1) {
                System.out.print((char) c);
            }
        }
```

Experiment No: 29 Lab Cycle 6 Date: 29-04-2024

FILE MANIPULATION –

READ AND COPY

AIM:

Write a program that reads from a file having integers. Copy even numbers and odd numbers to separate files.

ALGORITHM:

1. Open Files
    a. Open numbers.txt for reading.
    b. Open odd.txt and even.txt for writing.
2. Read and Process Characters
    a. Read each character from numbers.txt until the end of the file:
    b. If the character's ASCII value is even, write it to even.txt.
    c. If the character's ASCII value is odd, write it to odd.txt.
    d. Print the character to the console.
3. Close numbers.txt, odd.txt, and even.txt.
4. Print Output Files

```
    }
}
```

OUTPUT:

1234567890

Contents of odd.txt:

13579

Contents of even.txt:

24680

RESULT: Program is executed successfully and output is obtained.