20MCA132 – OBJECT ORIENTED PROGRAMMING LAB

*Lab Report Submitted By*

**PRANAV P K**

# Reg. No.:AJC22MCA-2072

*In Partial fulfilment for the Award of the Degree Of*

**MASTER OF COMPUTER APPLICATIONS (2 Year) (MCA)**

**APJ ABDUL KALAM TECHNOLOGICALUNIVERSITY**



# AMAL JYOTHI COLLEGE OF ENGINEERING KANJIRAPPALLY

[Affiliated to APJ Abdul Kalam Technological University, Kerala. Approved by AICTE, Accredited by NAAC with ‘A’ grade. Koovapally, Kanjirappally, Kottayam, Kerala – 686518]

**2022-2023**

# DEPARTMENT OF COMPUTER APPLICATIONS

**AMAL JYOTHI COLLEGE OF ENGINEERING KANJIRAPPALLY**



**CERTIFICATE**

This is to certify that the lab report, “**20MCA132 OBJECT ORIENTED PROGRAMMING LAB”** is the bonafide work of **PRANAV P K (AJC22MCA-2072)** in partial fulfilment of the requirements for the award of the Degree of Master of Computer Applications under APJ Abdul Kalam Technological University during the year **2022-23.**

Mrs. Lisha Varghese Rev. Fr. Dr. Rubin Thottupurathu Jose

**Lab In- Charge Head of the Department**

**Internal Examiner External Examiner**



|  |  |  |  |
| --- | --- | --- | --- |
| **Course Code** | **Course Name** | **Syllabus Year** | **L-T-P-C** |
| 20MCA132 | Object Oriented Programming Lab | 2020 | 0-1-3-2 |

## VISION

To promote an academic and research environment conducive for innovation centric technical education.

## MISSION

MS1 - Provide foundations and advanced technical education in both theoretical and applied Computer Applications in-line with Industry demands.

MS2 - Create highly skilled computer professionals capable of designing and innovating real life solutions.

MS3 - Sustain an academic environment conducive to research and teaching focused to generate up- skilledprofessionals with ethical values.

MS4 - Promote entrepreneurial initiatives and innovations capable of bridging and contributing with sustainable, socially relevant technology solutions.

## COURSE OUTCOME

|  |  |  |
| --- | --- | --- |
| **CO** | **Outcome** | **Target** |
| CO1 | Understand object-oriented concepts and design classes and objects to solve problems. | 60 |
| CO2 | Familiarization and understanding of arrays and strings. | 60 |
| CO3 | Understand and implement object-oriented concepts like inheritance, overloading and interfaces. | 60 |
| CO4 | Implement packages, exception handling, multithreading and generic programming by using the java.util package and Collection framework. | 60 |
| CO5 | Develop applications to handle events using applets | 60 |
| CO6 | Design applications using files and networking concepts. | 60 |

**COURSE END SURVEY**

|  |  |  |
| --- | --- | --- |
| **CO** | **Survey Question** | **Answer Format** |
| CO1 | To what extend you are able to understand object-oriented concepts and design classes and objects to solve problems? | Excellent/Very Good/Good  /Fair/Poor |
| CO2 | To what extend you are able to implement arrays and strings? | Excellent/Very Good/Good  /Fair/Poor |
| CO3 | To what extend you are able to implement object-oriented concepts like inheritance, overloading and interfaces? | Excellent/Very Good/Good  /Fair/Poor |
| CO4 | To what extend you are able to implement packages, exception  handling , multithreading and generic programming. Use java.util package and Collection framework? | Excellent/Very Good/Good  /Fair/Poor |
| CO5 | To what extent you are able to develop applications to handle events using applets? | Excellent/Very Good/Good  /Fair/Poor |
| CO6 | To what extend you are able to develop applications using files and networking concepts? | Excellent/Very Good/Good  /Fair/Poor |



*MCA 2022-2024*

**CONTENT**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl.**  **No.** | **Experiment** | **Date** | **CO** | **Page No.** |
| **1** | Define a class ‘product’ with data members pcode, pname and price. Create 3 objects of the class and find the product having the lowest price. | 07-03-2023 | CO1 | 1 |
| **2** | Read 2 matrices from the console and perform matrix addition. | 09-03-2023 | CO1 | 4 |
| **3** | Add complex numbers | 09-03-2023 | CO1 | 7 |
| **4** | Read a matrix from the console and check whether it is symmetric or not. . | 14-03-2023 | CO1 | 9 |
| **5** | Program to Sort strings | 16-03-2023 | CO2 | 11 |
| **6** | Search an element in an array. | 16-03-2023 | CO2 | 13 |
| **7** | Perform string manipulations | 16-03-2023 | CO2 | 15 |
| **8** | Program to create a class for Employee having attributes eNo, eName eSalary. Read n employ information and Search for an employee given eNo, using the concept of Array of Objects. | 16-03-2023 | CO2 | 17 |
| **9** | Area of different shapes using overloaded functions | 21-03-2023 | CO3 | 20 |
| **10** | 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. | 21-03-2023 | CO3 | 22 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl.**  **No.** | **Experiment** | **Date** | **CO** | **Page No.** |
| **11** | 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, Teacherid and also contain constructors and methods to  display the data members. Use array of objects to  display details of N teachers. | 23-03-2023 | CO3 | 26 |
| **12** | Write a program has class Publisher, Book, Literature and Fiction. Read the information and print the details of books from either the category, using inheritance. | 23-03-2023 | CO3 | 31 |
| **13** | Create classes Student and Sports. Create another class Result inherited from Student and Sports.  Display the academic and sports score of a student. | 28-03-2023 | CO3 | 35 |
| **14** | 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. | 28-03-2023 | CO3 | 38 |
| **15** | Prepare bill with the given format using calculate method from interface.  Order No.  Date :  ProductId Name Quantity unitprice Total 101 A 2 25 50  102 B 1 100 100  Net.Amount 150 | 28-03-2023 | CO3 | 42 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl.**  **No.** | **Experiment** | **Date** | **CO** | **Page No.** |
| **16** | Create a Graphics package that has classes and interfaces for figures Rectangle, Triangle, Square and Circle. Test the package by finding the area of these figures. | 04-04-2023 | CO4 | 46 |
| **17** | 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. | 04-04-2023 | CO4 | 50 |
| **18** | Write a user defined exception class to authenticate the user name and password. | 11-04-2023 | CO4 | 53 |
| **19** | Find the average of N positive integers, raising a user defined exception for each negative input. | 11-04-2023 | CO4 | 56 |
| **20** | Define 2 classes; one for generating multiplication table of 5 and other for displaying first N prime numbers. Implement using threads. (Thread class). | 11-04-2023 | CO4 | 58 |
| **21** | Define 2 classes; one for generating Fibonacci numbers and other for displaying even numbers in a given range. Implement using threads. (Runnable Interface) . | 13-04-2023 | CO4 | 61 |
| **22** | Program to create a generic stack and do the Push and Pop operations. | 13-04-2023 | CO4 | 64 |
| **23** | Using generic method perform Bubble sort. | 18-04-2023 | CO4 | 66 |
| **24** | Maintain a list of Strings using ArrayList from collection framework, perform built-in operations. | 18-04-2023 | CO4 | 68 |
| **25** | Program to remove all the elements from a linked list. | 18-04-2023 | CO4 | 70 |
| **26** | Program to remove an object from the Stack when the position is passed as parameter. | 27-04-2023 | CO4 | 72 |
| **27** | Program to demonstrate the creation of queue object using the PriorityQueue class. | 01-06-2023 | CO4 | 74 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl.**  **No.** | **Experiment** | **Date** | **CO** | **Page No.** |
| **28** | Program to demonstrate the addition and deletion of elements in deque. | 01-06-2023 | CO4 | 76 |
| **29** | Program to demonstrate the creation of Set object using the LinkedHashset class. | 08-06-2023 | CO4 | 78 |
| **30** | Write a Java program to compare two hash set. | 08-06-2023 | CO4 | 80 |
| **31** | Program to demonstrate the working of Map interface by adding, changing and removing elements. | 13-06-2023 | CO4 | 83 |
| **32** | Program to Convert HashMap to TreeMap . | 13-06-2023 | CO4 | 85 |
| **33** | Program to draw Circle, Rectangle, Line in Applet. | 20-06-2023 | CO5 | 87 |
| **34** | Program to find maximum of three numbers using AWT. | 20-06-2023 | CO5 | 89 |
| **35** | Find the percentage of marks obtained by a student in 5 subjects. Display a happy face if he secures above 50% or a sad face if otherwise. | 22-06-2023 | CO5 | 91 |
| **36** | Using 2D graphics commands in an Applet, construct a house. On mouse click event, change the color of the door from blue to red. | 22-06-2023 | CO5 | 95 |
| **37** | Develop a program that has a Choice component which contains the names of shapes such as rectangle, triangle, square and circle. Draw the corresponding shapes for given parameters as per user’s choice. | 27-06-2023 | CO5 | 99 |
| **38** | Develop a program to handle all mouse events and window events. | 04-07-2023 | CO5 | 103 |
| **39** | Write a program to write to a file, then read from the file and display the contents on the console. | 20-07-2023 | CO6 | 106 |
| **40** | Write a program to copy one file to another. | 20-07-2023 | CO6 | 108 |
| **41** | Write a program that reads from a file having integers. Copy even numbers and odd numbers to separate files. | 22-07-2023 | CO6 | 110 |
| **42** | Client Server communication using DatagramSocket - UDP | 27-07-2023 | CO6 | 112 |

**Experiment No.: 1**

**Aim:** Define a class ‘product’ with data members pcode, pname and price. Create 3 objects of the class and find the product having the lowest price.

**CO1**: Understand object-oriented concepts and design classes and objects to solve problems.

**Procedure**:

import java.util.\*;

public class Product{

int pcode;

String pname;

int price;

public void get(){

Scanner sc =new Scanner(System.in);

System.out.println("Enter the pcode:");

pcode=sc.nextInt();

System.out.println("Enter the product name:");

pname=sc.next();

System.out.println("Enter the price:");

price=sc.nextInt();

}

public void put()

{

System.out.println("The Product details are:");

System.out.println("The Product code:"+pcode);

System.out.println("The Product name:"+pname);

System.out.println("The Product price:"+price);

}

public static void main(String args[])

{

Product p1=new Product();

Product p2=new Product();

Product p3=new Product();

p1.get();

p2.get();

p3.get();

p1.put();

p2.put();

p3.put();

if(p1.price<p2.price && p1.price<p3.price)

{

System.out.println("The lowest price is:");

p1.put();

}

else if(p2.price<p1.price && p2.price<p3.price)

{

System.out.println("The lowest price is:");

p2.put();

}

else

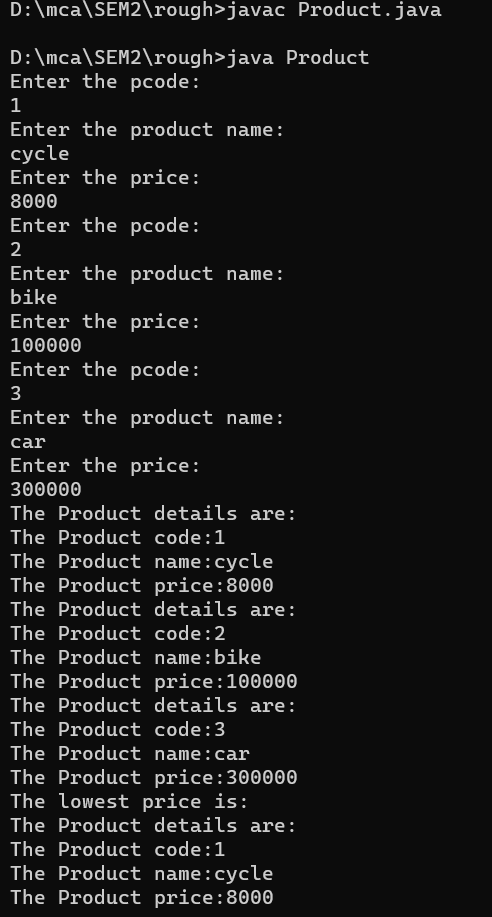
{ System.out.println("The lowest price is:");

p3.put();

}

}}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

**Experiment No.: 2**

**Aim:** Read 2 matrices from the console and perform matrix addition.

**CO1**: Understand object-oriented concepts and design classes and objects to solve problems.

**Procedure**:

import java.util.\*;

public class Matrix{

public static void main(String args[]){

Scanner obj =new Scanner(System.in);

System.out.println("enter the size of matrix:");

int n=obj.nextInt();

int arr1[][] = new int [n][n];

int arr2[][] = new int [n][n];

int arr3[][] = new int [n][n];

System.out.println("enter the elements in 1st matrix:");

for(int i=0;i<n;i++)

{

for(int j=0;j<n;j++)

{

arr1[i][j] = obj.nextInt(); }

}

System.out.println("enter the elements in 2nd matrix:");

for(int i=0;i<n;i++)

{

for(int j=0;j<n;j++)

{

arr2[i][j] = obj.nextInt();

}}

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

for(int i=0;i<n;i++)

{for(int j=0;j<n;j++)

{

System.out.print(arr1[i][j]+" "); }

System.out.println();

}

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

for(int i=0;i<n;i++)

{

for(int j=0;j<n;j++)

{

System.out.print(arr2[i][j]+" ");

}

System.out.println();

}

for(int i=0;i<n;i++)

{

for(int j=0;j<n;j++)

{

arr3[i][j]=arr1[i][j]+arr2[i][j];

}}

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

for(int i=0;i<n;i++)

{

for(int j=0;j<n;j++)

{

System.out.print(arr3[i][j]+" ");

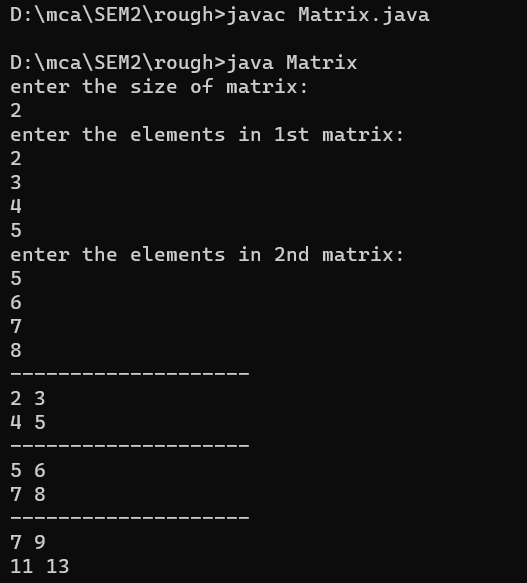
}

System.out.println();

}

}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

**Experiment No.: 3**

**Aim:** Add complex numbers

**CO1**: Understand object-oriented concepts and design classes and objects to solve problems.

**Procedure**:

import java.util.\*;

public class Addcomplex{

public static void main(String args[])

{

Scanner obj =new Scanner(System.in);

System.out.println("Enter complex number 1:");

int a=obj.nextInt();

int b=obj.nextInt();

System.out.println("Enter complex number 2:");

int c=obj.nextInt();

int d=obj.nextInt();

int real = a+b;

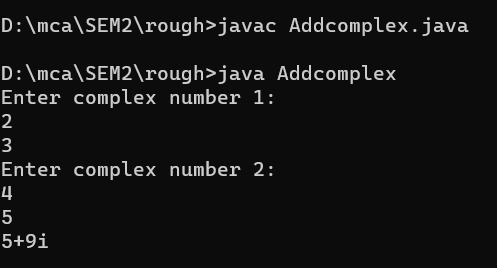
int imag = c+d;

System.out.println(real+"+"+imag+"i");

}

}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

**Experiment No.: 4**

**Aim:** Read a matrix from the console and check whether it is symmetric or not. .

**CO1**: Understand object-oriented concepts and design classes and objects to solve problems.

**Procedure**:

import java.util.Scanner;

public class Symmetric {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Enter the number of rows of the matrix: ");

int rows = sc.nextInt();

System.out.print("Enter the number of columns of the matrix: ");

int cols = sc.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] = sc.nextInt();

}

}boolean isSymmetric = true;

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

for (int j = 0; j < cols; j++) {

if (matrix[i][j] != matrix[j][i]) {

isSymmetric = false;

break;

}

} if (!isSymmetric) {

break;

}

} if (isSymmetric) {

System.out.println("The matrix is symmetric");

} else {

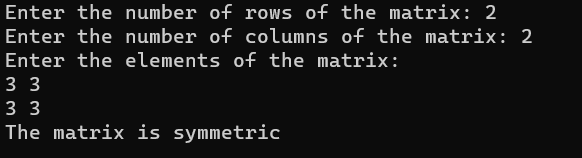
System.out.println("The matrix is not symmetric");

}

}

}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

**Experiment No.: 5**

**Aim:** Program to Sort strings

**CO2**: Familiarization and understanding of arrays and strings.

**Procedure**:

import java.util.Scanner;

public class co2q1 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Enter the size of the Array :");

int l = sc.nextInt();

String str[] = new String[l];

int i;

System.out.println("Enter the Elements of the Array");

for (i = 0; i < str.length; i++) {

str[i] = sc.next();

}

String temp;

int j;

for (i = 0; i < str.length; i++) {

for (j = i + 1; j < str.length; j++) {

if (str[i].compareTo(str[j]) > 0) {

temp = str[i];

str[i] = str[j];

str[j] = temp;

}

}

}

System.out.println("The Sorted Array : ");

for (i = 0; i < str.length; i++) {

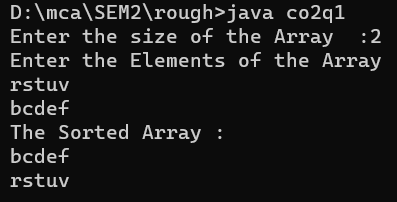
System.out.println(str[i]);

}

}

}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO2 was obtained.

**Experiment No.: 6**

**Aim:** Search an element in an array.

**CO2**: Familiarization and understanding of arrays and strings.

**Procedure**:

import java.util.Scanner;

public class Arrays

{

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

System.out.println("Enter the size of the array:");

int l=sc.nextInt();

int arr[]=new int[l];

int i;

System.out.println("Enter the elements into the array:");

for(i=0;i<arr.length;i++)

{

arr[i]=sc.nextInt();

}

System.out.println("The array elements are:");

for(i=0;i<arr.length;i++)

{

System.out.println(arr[i]);

}

System.out.println("Enter the array element to be searched:");

int val=sc.nextInt();

for(i=0;i<arr.length;i++)

{

if(arr[i]==val)

{

System.out.println("The element is found at:"+i+" position");

break;

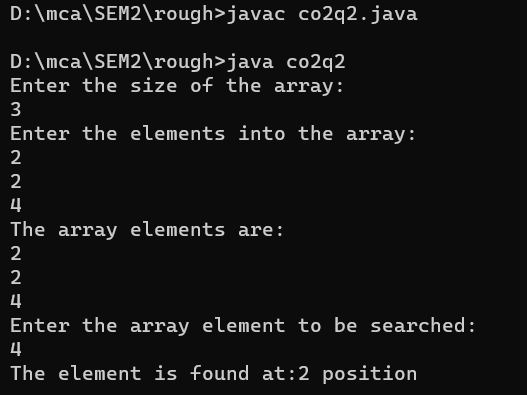
}

}

}

}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO2 was obtained.

**Experiment No.: 7**

**Aim:** Perform string manipulations

**CO2**: Familiarization and understanding of arrays and strings.

**Procedure**:

import java.util.\*;

public class Mystring

{

public static void main(String[] args)

{

Scanner obj= new Scanner(System.in);

System.out.print("Enter 1st string: ");

String str1= obj.nextLine();

System.out.print("Enter 2nd string: ");

String str2= obj.nextLine();

System.out.println(str1.toUpperCase());

System.out.println(str1.toLowerCase());

System.out.println("After Concating :"+str1.concat(str2));

System.out.println(str1.equals(str2));

System.out.println("Length of the string 1 :"+str1.length());

System.out.println("Length of the string 2 :"+str2.length());

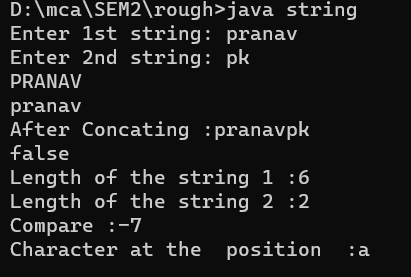
System.out.println("Compare :"+str2.compareTo(str1));

System.out.println("Character at the position :"+str1.charAt(4));

}

}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO2 was obtained.

**Experiment No.: 8**

**Aim:** Program to create a class for Employee having attributes eNo, eName eSalary. Read n employ information and Search for an employee given eNo, using the concept of Array of Objects.

**CO2**: Familiarization and understanding of arrays and strings.

**Procedure**:

import java.util.\*;

public class Employee

{

int eno;

String ename;

int esalary;

public void get(){

Scanner sc=new Scanner(System.in);

System.out.println("Enter the employee number:");

eno=sc.nextInt();

System.out.println("Enter the employee name:");

ename=sc.next();

System.out.println("Emter the salary of the employee:");

esalary=sc.nextInt();

}

public void put()

{

System.out.println("The employee details:");

System.out.println(eno);

System.out.println(ename);

System.out.println(esalary);

}

public static void main(String args[]){

int i,val;

Scanner sc=new Scanner(System.in);

System.out.println("Enter the number of employees:");

int n=sc.nextInt();

Employee e[]=new Employee[n];

for(i=0;i<n;i++)

{

e[i]=new Employee();

e[i].get();

}

for(i=0;i<n;i++){

e[i].put();

}

System.out.println("Enter the employee number to search for an employee:");

val=sc.nextInt();

for(i=0;i<n;i++)

{

if(e[i].eno==val){

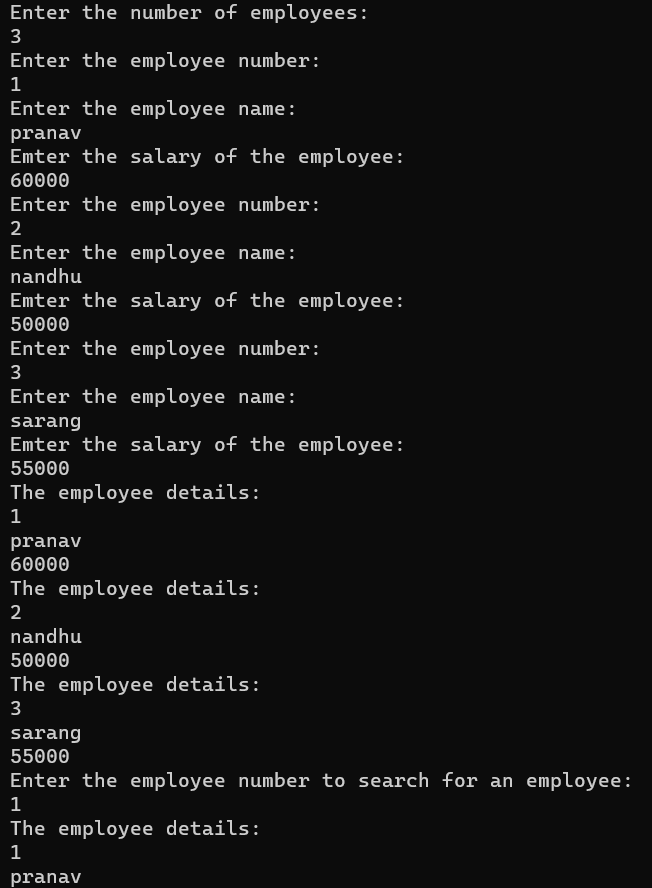
e[i].put();

}

}

}}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO2 was obtained.

**Experiment No.: 9**

**Aim:** Area of different shapes using overloaded functions

**CO3**: Understand and implement object-oriented concepts like inheritance, overloading and interfaces.

**Procedure**:

import java.util.\*;

class shape{

public double area(double radius){

return 3.14 \* radius \* radius;

}

public double area(double base,double height){

return 0.5 \* base \* height;

}

public double area(float side){

return side \* side;

}

public double area(double length,float breadth){

return length \* breadth;

}}

public class co3q1{

public static void main(String[]args){

shape obj=new shape();

Scanner sc=new Scanner(System.in);

System.out.print("Enter the radius of circle:");

int rs=sc.nextInt();

System.out.print("Enter the base of triangle::");

int bs=sc.nextInt();

System.out.print("Enter the height of triangle:");

int ht=sc.nextInt();

System.out.print("Enter the side of square:");

int sd=sc.nextInt();

System.out.print("Enter the length of rectangle:");

int lg=sc.nextInt();

System.out.print("Enter the breadth of rectangle:");

int bt=sc.nextInt();

System.out.println("area of circle:"+obj.area(rs));

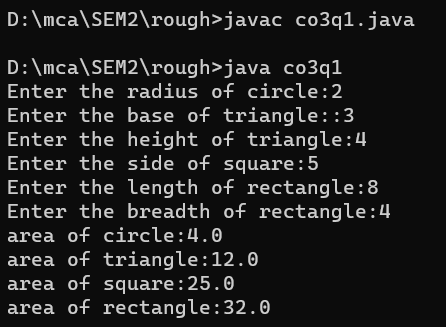
System.out.println("area of triangle:"+obj.area(bs,ht));

System.out.println("area of square:"+obj.area(sd));

System.out.println("area of rectangle:"+obj.area(lg,bt));

}}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO3 was obtained.

**Experiment No.: 10**

**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.

**CO3**: Understand and implement object-oriented concepts like inheritance, overloading and interfaces.

**Procedure**:

import java.util.\*;

class Employee

{

String empid;

String name;

int salary;

String address;

Employee(String id,String ename,int sal,String addr)

{

empid =id;

name=ename;

salary=sal;

address=addr;

}

Employee()

{

Scanner sc=new Scanner(System.in);

System.out.println("Enter the employee id:");

empid=sc.next();

System.out.println("Enter the employee name:");

name=sc.next();

System.out.println("Enter the employee salary:");

salary=sc.nextInt();

System.out.println("Enter the employee address:");

address=sc.next();

}

}

class Teacher extends Employee

{

String department;

String subject;

Teacher(String id,String ename,int sal,String addr,String deptment,String sub){

super(id,ename,sal,addr);

department=deptment;

subject=sub;

}

Teacher()

{

Scanner sc=new Scanner(System.in);

System.out.println("Enter the employee department:");

department=sc.next();

System.out.println("Enter the employee subject:");

subject=sc.next();

}

void display()

{

System.out.println("Employee id:"+empid);

System.out.println("Employee name:"+name);

System.out.println("Employee slary:"+salary);

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

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

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

}}public class Multi{

public static void main(String args[])

{

int i;

Scanner sc=new Scanner(System.in);

System.out.println("Enter the number of employess:");

int n=sc.nextInt();

Teacher e[]=new Teacher[n];

for(i=0;i<n;i++)

{e[i]=new Teacher();

}

for(i=0;i<n;i++){

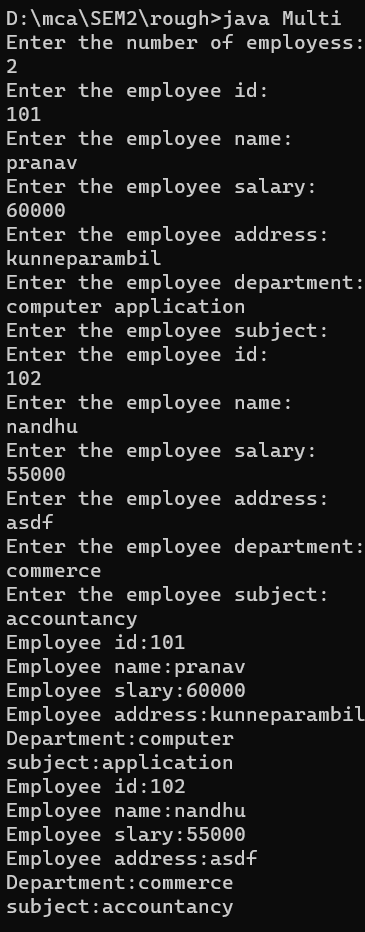
e[i].display();

}

}

}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO3 was obtained.

**Experiment No.: 11**

**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, Teacherid and also contain constructors and methods to

display the data members. Use array of objects to

display details of N teachers.

**CO3**: Understand and implement object-oriented concepts like inheritance, overloading and interfaces.

**Procedure**:

import java.util.\*;

class Person

{

String name;

String gender;

String address;

int age;

Person()

{

Scanner sc=new Scanner(System.in);

System.out.println("name:");

name=sc.next();

System.out.println("gender:");

gender=sc.next();

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

address=sc.next();

System.out.println("age:");

age=sc.nextInt();

}

}

class Employee extends Person

{

String empid;

String company\_name;

String qualifiaction;

int salary;

Employee()

{

Scanner sc=new Scanner(System.in);

System.out.println("Id:");

empid=sc.next();

System.out.println("Company name:");

company\_name=sc.next();

System.out.println("qualifiaction:");

qualifiaction=sc.next();

System.out.println("Salary:");

salary=sc.nextInt();

}}

class Teacher extends Employee

{

String subject;

String department;

String teacherid;

Teacher()

{

Scanner sc=new Scanner(System.in);

System.out.println("Id:");

teacherid=sc.next();

System.out.println("Subject:");

subject=sc.next();

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

department=sc.next();

}

void display()

{

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

System.out.println("gender:"+gender);

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

System.out.println("age:"+age);

System.out.println("empid:"+empid);

System.out.println("company\_name:"+company\_name);

System.out.println("qualifiaction:"+qualifiaction);

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

System.out.println("teacherid:"+teacherid);

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

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

}

}

public class Multiii{

public static void main(String args[])

{

int i;

Scanner sc=new Scanner(System.in);

System.out.println("Enter the number of teachers:");

int n=sc.nextInt();

Teacher e[]=new Teacher[n];

for(i=0;i<n;i++)

{

e[i]=new Teacher();

}

for(i=0;i<n;i++){

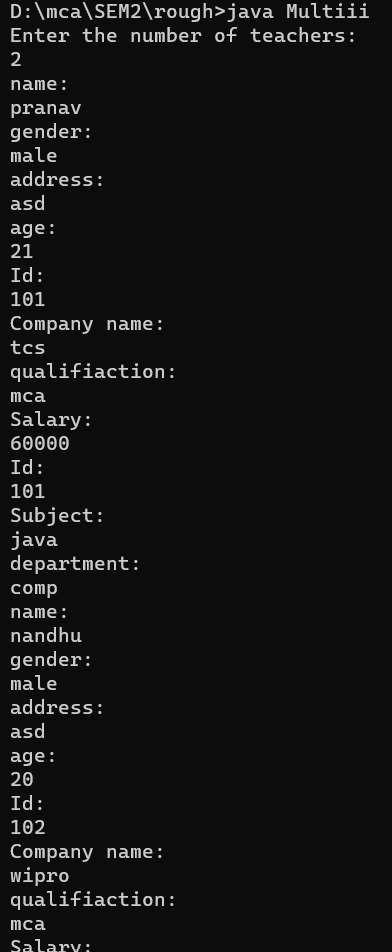
e[i].display();

}

}

}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO3 was obtained.

**Experiment No.: 12**

**Aim:** Write a program has class Publisher, Book, Literature and Fiction. Read the information and print the details of books from either the category, using inheritance.

**CO3**: Understand and implement object-oriented concepts like inheritance, overloading and interfaces.

**Procedure**:

import java.util.\*;

class publisher{

String pub\_name;

publisher()

{

Scanner obj=new Scanner(System.in);

System.out.println("Publisher name :");

pub\_name=obj.next();

}

}

class book extends publisher

{

String book\_name;

book()

{

Scanner obj=new Scanner(System.in);

System.out.println("book name :");

book\_name=obj.next();

}

}

class literature extends book

{

void display()

{

System.out.println("Publisher name is :"+pub\_name);

System.out.println("book name is :"+book\_name);

}

}

class fiction extends book

{

void display()

{

System.out.println("Publisher name is :"+pub\_name);

System.out.println("book name is :"+book\_name);

}

}

public class library

{

public static void main(String[] args)

{

int i=0;

Scanner obj=new Scanner(System.in);

System.out.println("enter the total number :");

int size=obj.nextInt();

literature arr1[]= new literature[size];

fiction arr2[]=new fiction[size];

System.out.println("enter the details of literature books :");

for(i=0;i<size;i++)

{

arr1[i]=new literature();

}

System.out.println("enter the details of fiction books :");

for(i=0;i<size;i++)

{

arr2[i]=new fiction();

}

System.out.println("\n\n\n\n");

System.out.println("details of literature books :");

for(i=0;i<size;i++)

{

arr1[i].display();

}

System.out.println("\n\n\n\n");

System.out.println("details of fiction books :");

for(i=0;i<size;i++)

{

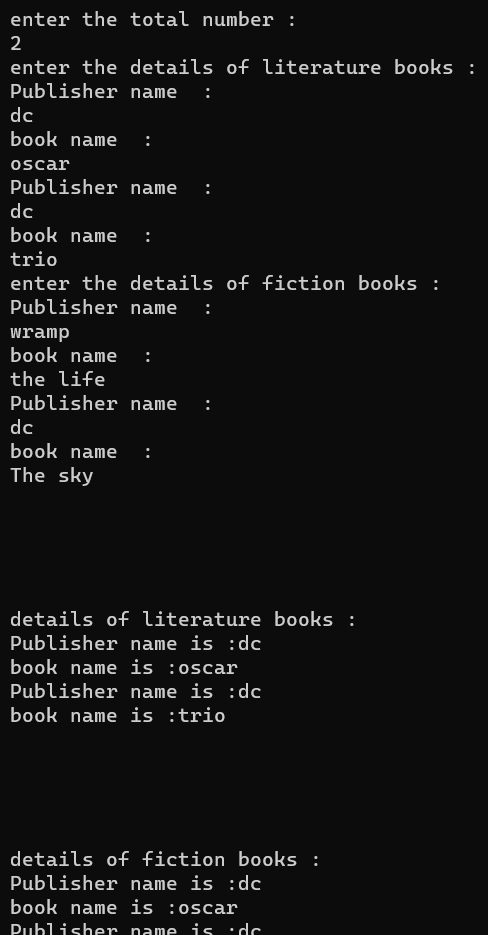
arr1[i].display();

}

}

}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO3 was obtained.

**Experiment No.: 13**

**Aim:** Create classes Student and Sports. Create another class Result inherited from Student and Sports.Display the academic and sports score of a student.

**CO3**: Understand and implement object-oriented concepts like inheritance, overloading and interfaces.

**Procedure**:

import java.util.Scanner;

interface Student {

void get();

void disp();

}

interface Sports {

void get1();

void disp1();

}

public class Result implements Student, Sports {

float t, p;

float m1, m2, m3;

int rno;

String name, sport, grade;

public void get() {

Scanner sc = new Scanner(System.in);

System.out.print("Enter student rollno: ");

rno = sc.nextInt();

System.out.print("Enter student name: ");

name = sc.next();

System.out.print("Enter mark of Subject1 out of 100: ");

m1 = sc.nextFloat();

System.out.print("Enter mark of subject2 out of 100: ");

m2 = sc.nextFloat();

System.out.print("Enter mark of subject3 out of 100: ");

m3 = sc.nextFloat();}

public void get1() {

Scanner sc = new Scanner(System.in);

System.out.print("Enter sports item: ");

sport = sc.next();

System.out.print("Enter grade: ");

grade = sc.next();}

void cal() {

t = m1 + m2 + m3;

p = (t / 300) \* 100;

}

public void disp() {

System.out.println("Student Name: " + name);

System.out.println("Rollno: " + rno);

System.out.println("Subject1 mark: " + m1);

System.out.println("Subject2 mark: " + m2);

System.out.println("Subject3 mark: " + m3);

System.out.println("Total mark: " + t);

System.out.println("Percentage: " + p);

}

public void disp1() {

System.out.println("Sports item: " + sport);

System.out.println("Grade: " + grade);

}

public static void main(String[] args) {

Result s = new Result();

s.get();

s.get1();

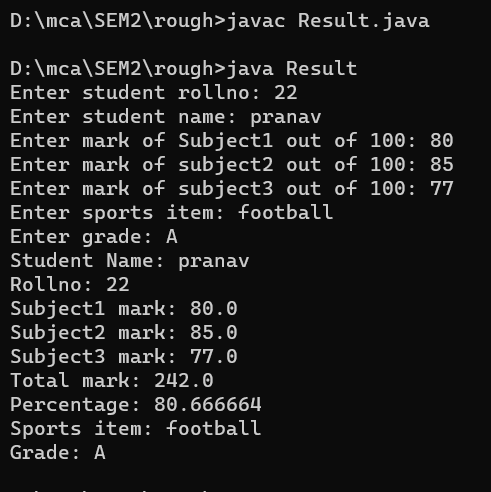
s.cal();

s.disp();

s.disp1();

}}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO3 was obtained.

**Experiment No.: 14**

**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.

**CO3**: Understand and implement object-oriented concepts like inheritance, overloading and interfaces.

**Procedure**:

Main.java

import java.util.Scanner;

public class Main {

public static void main(String[] args){

Scanner scanner = new Scanner(System.in);

int shape,operation;

System.out.println("Choose a Shape 1)Circle 2)Rectangle : ");

shape = scanner.nextInt();

System.out.println("Choose an Operation 1)Perimeter 2)Area : ");

operation = scanner.nextInt();

if(shape==1){

Circle circle = new Circle();

if(operation==1){

circle.perimeter();

}

else if(operation==2)

{

circle.area();

}

else {

System.out.println("Operation code.");

}

}

else if(shape==2)

{

Rectangle rectangle = new Rectangle();

if(operation==1){

rectangle.perimeter();

}

else if(operation==2)

{

rectangle.area();

}

else {

System.out.println("Operation code :");

System.exit(0);

}

}

else {

System.out.println("Incorrect Shape code.");

}

}

}

Shapes.java

public interface Shapes {

public void perimeter();

public void area();

}

Circle.java

import java.util.Scanner;

public class Circle implements Shape{

int radius;

Scanner sc = new Scanner(System.in);

public void perimeter() {

System.out.println("Input radius of circle : ");

radius = sc.nextInt();

String perimeter = Double.toString(Math.PI\*radius\*2);

System.out.println("Circumference of the circle is : "+perimeter);

}

public void area() {

System.out.println("Input radius of circle : ");

radius = sc.nextInt();

String area = Double.toString(Math.PI\*radius\*radius);

System.out.println("Area of the circle is : "+area);

}

}

Rectangle.java

import java.util.Scanner;

public class Rectangle implements Shape{

int length;

int breadth;

Scanner scanner = new Scanner(System.in);

public void perimeter() {

System.out.println("Input length of rectangle : ");

length = scanner.nextInt();

System.out.println("Input breadth of rectangle : ");

length = scanner.nextInt();

String perimeter = Double.toString(2\*(length+breadth));

System.out.println("Perimeter of the rectangle is : "+perimeter);

}

public void area() {

System.out.println("Input length of rectangle : ");

length = scanner.nextInt();

System.out.println("Input breadth of rectangle : ");

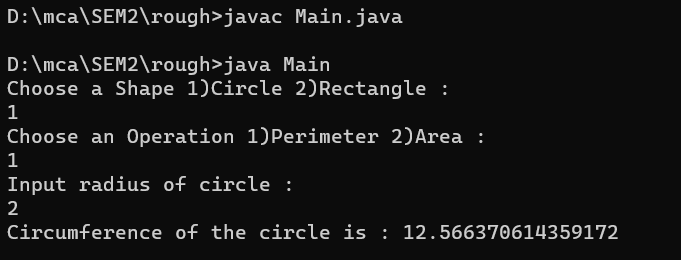
length = scanner.nextInt();

String area = Double.toString(length\*breadth);

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

}}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO3 was obtained.

**Experiment No.: 15**

**Aim:** Prepare bill with the given format using calculate method from interface.

Order No.

Date :

ProductId Name Quantity unitprice Total 101 A 2 25 50

102 B 1 100 100

Net.Amount 150

**CO3**: Understand and implement object-oriented concepts like inheritance, overloading and interfaces.

**Procedure**:

import java.util.Scanner;

interface Bill {

void inputs();

void prints();

double getTotal();

double getNetAmount();

}

class SingleProduct implements Bill {

Scanner sc = new Scanner(System.in);

int productId, quantity, unitPrice;

String name;

double total;

public void inputs() {

System.out.print("Enter the Product Id: ");

productId = sc.nextInt();

System.out.print("Enter the Product Name: ");

name = sc.next();

System.out.print("Enter the Quantity: ");

quantity = sc.nextInt();

System.out.print("Enter the Unit Price: ");

unitPrice = sc.nextInt();

total = quantity \* unitPrice;

}

public void prints() {

System.out.println(productId + "\t " + name + "\t " + quantity + "\t " + unitPrice + " \t " + total);

}

public double getTotal() {

return total;

}

public double getNetAmount() {

return getTotal();

}

}

class BillGenerator implements Bill {

int orderNo, num;

String date;

double amount;

SingleProduct[] products;

Scanner sc = new Scanner(System.in);

public void inputs() {

System.out.print("Enter the Order No: ");

orderNo = sc.nextInt();

System.out.print("Enter the date of purchase: ");

date = sc.next();

System.out.print("Enter the No of items: ");

num = sc.nextInt();

products = new SingleProduct[num];

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

products[i] = new SingleProduct();

products[i].inputs();

}

}

public void prints() {

System.out.println("Order No: " + orderNo);

System.out.println("Date: " + date);

System.out.println("ProductID Name Quantity UnitPrice Total");

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

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

products[i].prints();

}

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

System.out.println("\t\t\t\tNet Amount: " + getNetAmount());

}

public double getTotal() {

double total = 0.0;

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

total += products[i].getTotal();}

return total;

}

public double getNetAmount() {

return getTotal();

}

}

public class Order {

public static void main(String[] args) {

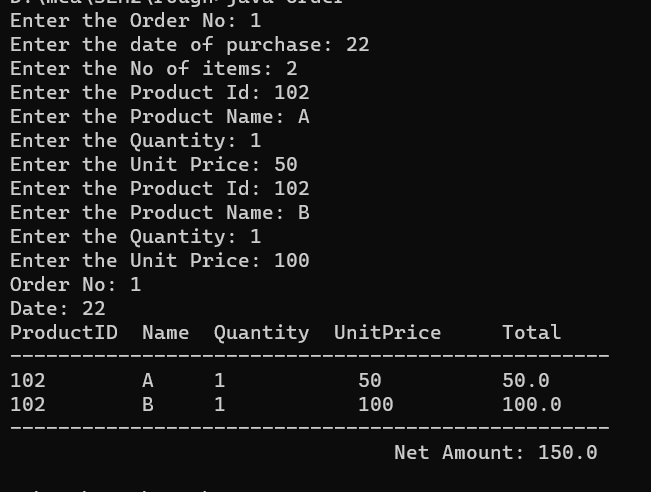
BillGenerator billGenerator = new BillGenerator();

billGenerator.inputs();

billGenerator.prints();

}}

**Output Screenshot**



**Result**

The program was executed and the result was successfully obtained. Thus CO3 was obtained.

**Experiment No.: 16**

**Aim:** Create a Graphics package that has classes and interfaces for figures Rectangle, Triangle, Square and Circle. Test the package by finding the area of these figures.

**CO4**: Implement packages, exception handling, multithreading and generic programming by using the java.util package and Collection framework.

**Procedure**:

package Graphiccs;

interface Area1

{

public void Rectangle();

public void Triangle();

public void Square();

public void Circle();

public void getRect();

public void getTri();

public void getSqr();

public void getCrl();

}

 //shapes.java

package Graphiccs;

import java.util.\*;

public class Shapes1 implements Area1

{

double lr,lb,ra,th,tb,ta,saa,sa,cr,cc;

public void getRect()

{

Scanner ab= new Scanner(System.in);

System.out.println("Enter the length of the rectangle");

lr=ab.nextInt();

System.out.println("Enter the breadth of the rectangle");

lb=ab.nextInt();

}

public void Rectangle()

{

ra=lr\*lb;

System.out.println("Area of Rectangle is "+ra);

}

public void getTri()

{

Scanner cb= new Scanner(System.in);

System.out.println("Enter the height of the Triangle");

th=cb.nextInt();

System.out.println("Enter the base of the Triangle");

tb=cb.nextInt();

}

public void Triangle()

{

ta=0.5\*th\*tb;

System.out.println("Area of Triangle angle is "+ta);

}

public void getSqr()

{

Scanner sq= new Scanner(System.in);

System.out.println("Enter the Side of the Square");

sa=sq.nextInt();

}

public void Square()

{

saa=sa\*sa;

System.out.println("Area of Square is "+saa);

}

public void getCrl()

{

Scanner sc= new Scanner(System.in);

System.out.println("Enter the radius of the Circle");

cc=sc.nextInt();

}

public void Circle()

{

cr=3.14\*cc\*cc;

System.out.println("Area of Square is "+cr);

}

public static void main(String[] args)

{

Shapes1 o= new Shapes1();

o.getRect();

o.Rectangle();

o.getTri();

o.Triangle();

o.getSqr();

o.Square();

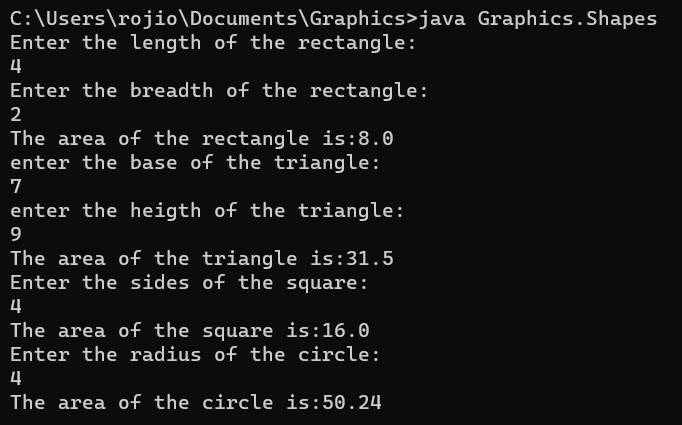
o.getCrl();

o.Circle();

}

}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO4 was obtained.

**Experiment No.: 17**

**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.

**CO4**: Implement packages, exception handling, multithreading and generic programming by using the java.util package and Collection framework.

**Procedure**:

//operations.java

package Arithematic;

interface operations{

public void input();

public void add();

public void substract();

public void multiply();

public void division();

}

//basic.jav

package Arithematic;

import java.util.\*;

public class basic implements operations

{

double a,b,ad,dif,mult,div;

public void input()

{

Scanner ab=new Scanner(System.in);

System.out.println("Enter two numbers");

a=ab.nextInt();

b=ab.nextInt();

}

public void add()

{

ad=a+b;

System.out.println("Sum is "+ad);

}

public void substract()

{

dif=a-b;

System.out.println("Difference is "+dif);

}

public void multiply()

{

mult=a\*b;

System.out.println("Product is "+mult);

}

public void division()

{

div=a/b;

System.out.println("Quotient is "+div);

}

public static void main(String[] args)

{

basic o=new basic();

o.input();

o.add();

o.substract();

o.multiply();

o.division();

}

}

**Output Screenshot**

C:\your\project\directory> java TestArithmetic

Addition: 15.0

Subtraction: 5.0

Multiplication: 50.0

Division: 2.0

**Result**

The program was executed and the result was successfully obtained. Thus CO4 was obtained.

**Experiment No.: 18**

**Aim:** Write a user defined exception class to authenticate the user name and password.

**CO4**: Implement packages, exception handling, multithreading and generic programming by using the java.util package and Collection framework.

**Procedure**:

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 CheckLoginCredential

{

public static void main(String[] args)

{

Scanner s=new Scanner(System.in);

String username;

String 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("hello"))

throw new PasswordException("incorrect password\n Type correct password!!!");

else

System.out.println("Login Suceessful!!");

}

catch(UsernameException u)

{

u.printStackTrace();

}

catch(PasswordException p)

{

p.printStackTrace();

}

finally

{

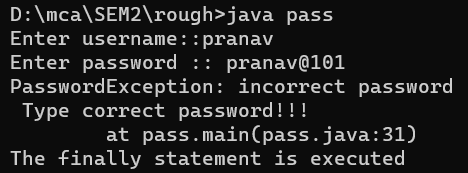
System.out.println("The finally statement is executed");

}

}

}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO4 was obtained.

**Experiment No.: 19**

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

**CO4**: Implement packages, exception handling, multithreading and generic programming by using the java.util package and Collection framework.

**Procedure**:

import java.util.Scanner;

public class AverageException{

public static class InvalidNumberException extends Exception {

public InvalidNumberException() {

super("Please provide a valid number!");

}

}

public static void main(String [] args){

Scanner sc=new Scanner(System.in);

int c,num,sum=0;

double avg;

System.out.println("enter the count");

c=sc.nextInt();

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

try{

num=sc.nextInt();

if(num>0){

sum+=num;

}else{

i--;

throw new InvalidNumberException();

}

}

catch(InvalidNumberException e){

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

}

}

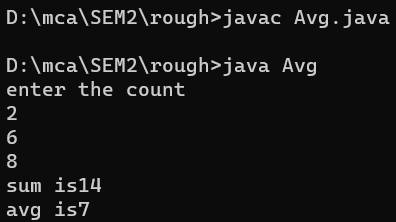
System.out.println("sum is"+sum);

System.out.println("avg is"+sum/c);

}

}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO4 was obtained.

**Experiment No.: 20**

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

**CO4**: Implement packages, exception handling, multithreading and generic programming by using the java.util package and Collection framework.

**Procedure**:

import java.util.Scanner;

class MulTable extends Thread{

public void run(){

int num=5;

System.out.println("\_\_Multiplication Table of 5 \n");

for(int i=1;i<=10;i++)

{

System.out.printf("%d \* %d=%d \n",num,i,num\*i);

}

}

}

class PrimeNo extends Thread{

public void run(){

int i,j,flag;

Scanner sc=new Scanner(System.in);

System.out.println("\n To generate first N prime numbers ");

System.out.println("Enter the limit(N):");

int N=sc.nextInt();

System.out.println("prime numbers between 1 and" + N +" are:");

for(i=1;i<=N;i++)

{

if(i==1||i==0)

continue;

flag=1;

for(j=2;j<=i/2;j++)

{

if(i%j==0)

{

flag=0;

break;

}

}

if(flag==1)

System.out.print(i + " ");

}

}

}

public class MulPrime {

public static void main(String arg[]) throws InterruptedException

{

MulTable obj1=new MulTable();

obj1.start();

obj1.sleep(2000);

PrimeNo obj2=new PrimeNo();

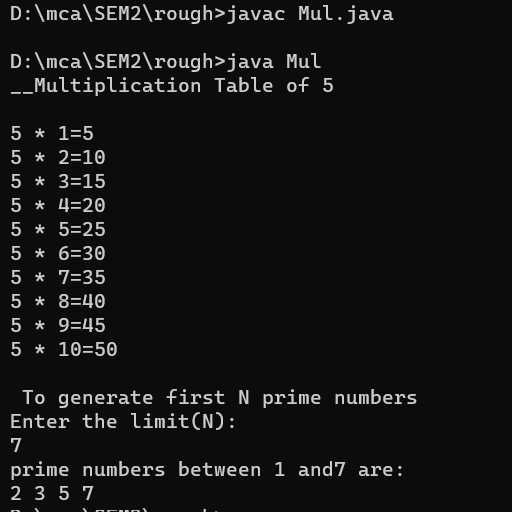
obj2.start();

obj2.sleep(1000);

}

}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO4 was obtained.

**Experiment No.: 21**

**Aim:** Define 2 classes; one for generating Fibonacci numbers and other for displaying even numbers in a given range. Implement using threads. (Runnable Interface) .

**CO4**: Implement packages, exception handling, multithreading and generic programming by using the java.util package and Collection framework.

**Procedure**:

import java.util.Scanner;

class Fibonacci implements Runnable{

public void run(){

int first=0,second=1,next;

Scanner sc=new Scanner(System.in);

System.out.println("\_\_\_To Generate Fibonacci Series\_\_");

System.out.println("Enter the no.of terms required:");

int n=sc.nextInt();

System.out.println("Series Generated!!");

for(int i=1;i<=n;i++){

System.out.print(first + "");

next=first + second;

first=second;

second=next;

}

}

}

class EvenNo implements Runnable{

public void run(){

Scanner sc=new Scanner(System.in);

int lower,upper;

System.out.println("\n\n\_\_\_\_\_To Generate Even numbers of given range\_\_\_\_");

System.out.println("Enter the lower limit:");

lower=sc.nextInt();

System.out.println("Enter the upper limit:");

upper=sc.nextInt();

System.out.println("Even numbers from" + lower +" and" +upper+"are:");

for(int i=lower;i<=upper;i++){

if(i%2!=0)

continue;

else

{

System.out.print(i+" ");

}}

}

}

public class ThreadRunner {

public static void main(String arg[]) throws InterruptedException

{

Fibonacci obj1=new Fibonacci();

Thread a=new Thread(obj1);

a.start();

a.sleep(2000);

EvenNo obj2=new EvenNo();

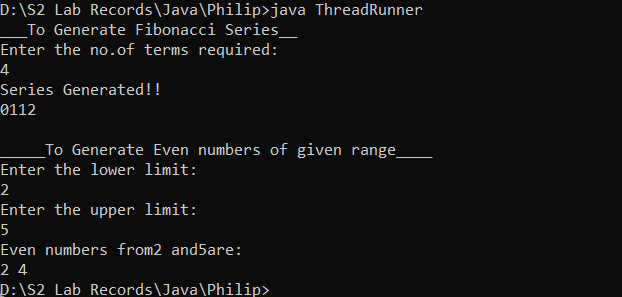
Thread b=new Thread(obj2);

b.start();

b.sleep(1000);

}

}

**Output Screenshot**

**Result**

The program was executed and the result was successfully obtained. Thus CO4 was obtained.

**Experiment No.: 22**

**Aim:** Program to create a generic stack and do the Push and Pop operations.

**CO4**: Implement packages, exception handling, multithreading and generic programming by using the java.util package and Collection framework.

**Procedure**:

import java.util.\*;

public class StackDemo

{

public static void main(String args[])

{

Stack<String> stack = new Stack<String>();

Scanner sc=new Scanner(System.in);

System.out.println("Enter the no.of items into the stack:");

int n=sc.nextInt();

for(int i=0;i<n;i++)

{

System.out.println("Enter the name of the item:");

String item=sc.next();

stack.add(item);

}

System.out.println("The items in the stack are:");

System.out.println(stack);

String rem\_ele=stack.remove(1);

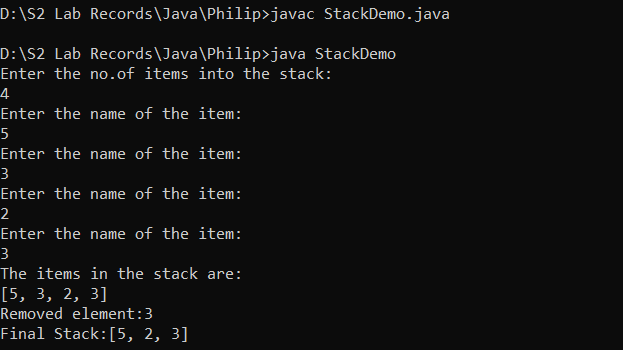
System.out.println("Removed element:"+rem\_ele);

System.out.println("Final Stack:"+stack);

}

}

**Output Screenshot**



**Result**

The program was executed and the result was successfully obtained. Thus CO4 was obtained.

**Experiment No.: 23**

**Aim:** Using generic method perform Bubble sort.

**CO4**: Implement packages, exception handling, multithreading and generic programming by using the java.util package and Collection framework.

**Procedure**:

import java.util.Arrays;

import java.util.Scanner;

public class Main {

static void bubbleSort(int array[])

{ int size = array.length;

for (int i = 0; i < size - 1; i++)

for (int j = 0; j < size - i - 1; j++)

if (array[j] > array[j + 1])

{ int temp = array[j];

array[j] = array[j + 1];

array[j + 1] = temp;

} }

public static void main(String args[]) {

Scanner scanner = new Scanner(System.in);

System.out.println("Number of items to be inserted : ");

int count = scanner.nextInt();

int[] data = new int[count];

System.out.println("Enter the array items : ");

for(int i=0;i<count;i++)

{

data[i] = scanner.nextInt();

}

Main.bubbleSort(data);

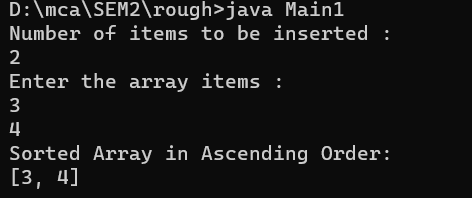
System.out.println("Sorted Array in Ascending Order:");

System.out.println(Arrays.toString(data));

}

}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO4 was obtained.

**Experiment No.: 24**

**Aim:** Maintain a list of Strings using ArrayList from collection framework, perform built-in operations.

**CO4**: Implement packages, exception handling, multithreading and generic programming by using the java.util package and Collection framework.

**Procedure**:

import java.util.\*;

public class array{

public static void main(String args[]){

ArrayList <String> list1=new ArrayList<String>();

Scanner sc=new Scanner(System.in);

System.out.println("Enter the no.of items into the list:");

int n=sc.nextInt();

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

System.out.println("Enter the name of the item:");

String item=sc.next();

list1.add(item);

}

//list1.add("benz");

//list1.add("audi");

System.out.println("The list items are:");

//for(int i=0;i<n;i++)

//{

System.out.println(list1);

//}

System.out.println("After sorting the list items:");

Collections.sort(list1);

for(String items:list1)

System.out.println(items);

int size=list1.size();

System.out.println("The length of the array list is:"+size);

System.out.println("Enter the name of the item to search:");

String p=sc.next();

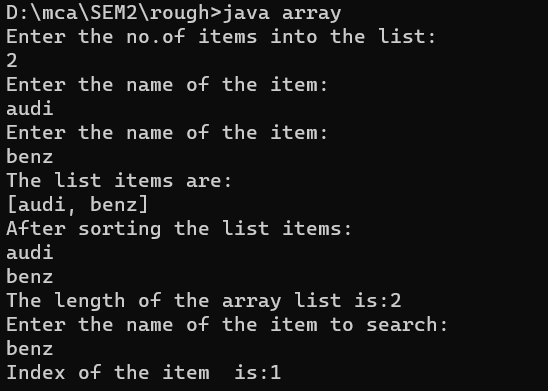
int pos=list1.indexOf(p);

System.out.println("Index of the item is:"+pos);

}

}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO4 was obtained.

**Experiment No.: 25**

**Aim:** Program to remove all the elements from a linked list.

**CO4**: Implement packages, exception handling, multithreading and generic programming by using the java.util package and Collection framework.

**Procedure**:

import java.util.\*;

public class LinkedListDemo {

public static void main(String args[])

{

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

Scanner sc=new Scanner(System.in);

System.out.println("Enter the no.of items into the linked list :");

int n=sc.nextInt();

for(int i=0;i<n;i++)

{

System.out.println("Enter the name of the item to the list:");

String item=sc.next();

list.add(item);

}

//System.out.println("The items in the list are:");

//list.add("car");

//list.add("bikes");

//list.add("10");

//list.add("20");

System.out.println("Orginal LinkedList:" +list);

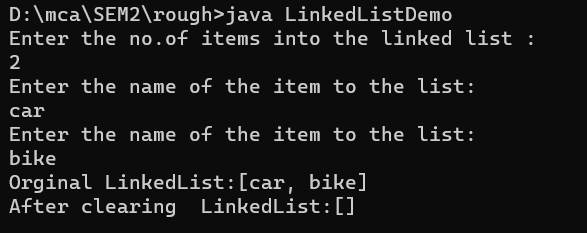
list.clear();

System.out.println("After clearing LinkedList:" +list);

}

}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO4 was obtained.

**Experiment No.: 26**

**Aim:** Program to remove an object from the Stack when the position is passed as parameter.

**CO4**: Implement packages, exception handling, multithreading and generic programming by using the java.util package and Collection framework.

**Procedure**:

import java.util.Scanner;

import java.util.Stack;

public class Mainnn {

public static void main(String[] args) {

Stack<String> stack = new Stack<String>();

Scanner scanner=new Scanner(System.in);

System.out.println("Enter the size of Stack : ");

int num=scanner.nextInt();

for(int i =0;i<num;i++)

{

System.out.println("Add element for Stack : ");

String str=scanner.next();

stack.add(str);

}

System.out.println();

System.out.println("Initial Stack: " + stack);

System.out.println("Enter the position of element in stack to be removed: ");

int pos=scanner.nextInt();

String rem = stack.remove(pos);

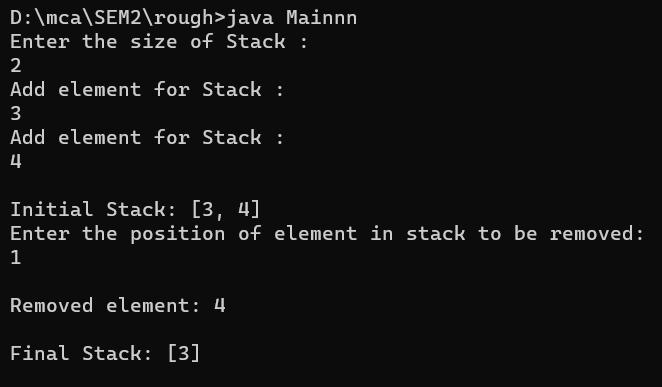
System.out.println("\nRemoved element: "+ rem);

System.out.println("\nFinal Stack: " + stack);

}

}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO4 was obtained.

**Experiment No.: 27**

**Aim:** Program to demonstrate the creation of queue object using the PriorityQueue class.

**CO4**: Implement packages, exception handling, multithreading and generic programming by using the java.util package and Collection framework.

**Procedure**:

import java.util.\*;

public class PriorityQueueDemo {

public static void main(String args[])

{

PriorityQueue<Integer> pQueue = new PriorityQueue<Integer>();

Scanner sc=new Scanner(System.in);

System.out.println("Enter the no.of items into the PriorityQueue:");

int n=sc.nextInt();

for(int i=0;i<n;i++)

{

System.out.println("Enter the the element:");

int item=sc.nextInt();

pQueue.add(item);

}

System.out.println("The items in the PriorityQueue are:");

System.out.println(pQueue);

//pQueue.add(10);

//pQueue.add(20);

//pQueue.add(30);

System.out.println("Printing the top element of PriorityQueue:");

System.out.println(pQueue.peek());

System.out.println("Printing the top element of PriorityQueue and removing it :");

System.out.println(pQueue.poll());

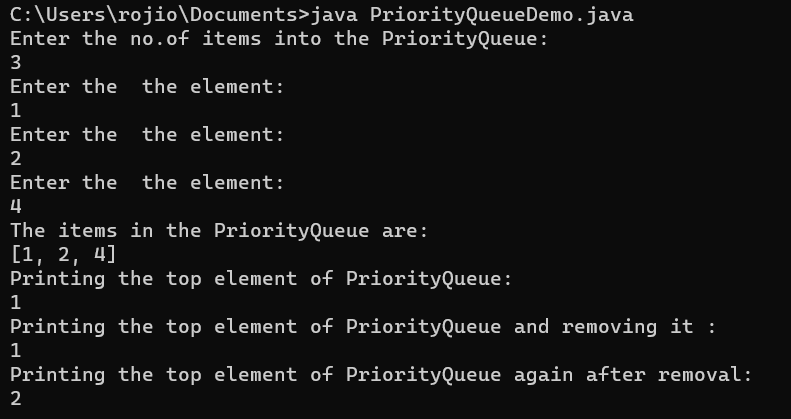
System.out.println("Printing the top element of PriorityQueue again after removal:");

System.out.println(pQueue.peek());

}

}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO4 was obtained.

**Experiment No.: 28**

**Aim:** Program to demonstrate the addition and deletion of elements in deque.

**CO4**: Implement packages, exception handling, multithreading and generic programming by using the java.util package and Collection framework.

**Procedure**:

import java.util.\*;

import java.util.Deque;

public class Example {

public static void main(String[] args)

{

Deque<Integer> deque=new ArrayDeque<>();

Scanner sc=new Scanner(System.in);

System.out.println("Enter the no.of items into the deque:");

int n=sc.nextInt();

for(int i=0;i<n;i++)

{

System.out.println("Enter the name of the element:");

int item=sc.nextInt();

deque.add(item);

}

System.out.println("The elements in the deque are:");

System.out.println(deque);

//deque.addFirst(1);

//deque.addLast(2);

int first = deque.removeFirst();

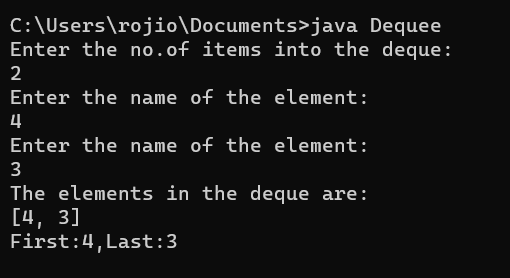
int last =deque.removeLast();

System.out.println("First:"+first +",Last:" + last);

}

}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO4 was obtained.

**Experiment No.: 29**

**Aim:** Program to demonstrate the creation of Set object using the LinkedHashset class.

**CO4**: Implement packages, exception handling, multithreading and generic programming by using the java.util package and Collection framework.

**Procedure**:

import java.util.LinkedHashSet;

public class Main {

public static void main(String[] args) {

LinkedHashSet<String> linkedset = new LinkedHashSet<String>();

// Adding element to LinkedHashSet

linkedset.add("Maruti");

linkedset.add("BMW");

linkedset.add("Honda");

linkedset.add("Audi");

linkedset.add("Maruti"); //This will not add new element as Maruti already

exists

linkedset.add("WalksWagon");

System.out.println("Size of LinkedHashSet=" + linkedset.size());

System.out.println("Original LinkedHashSet:" + linkedset);

System.out.println("Removing Audi from LinkedHashSet: " +

linkedset.remove("Audi"));

System.out.println("Trying to Remove Z which is not present: "

+ linkedset.remove("Z"));

System.out.println("Checking if Maruti is present=" +

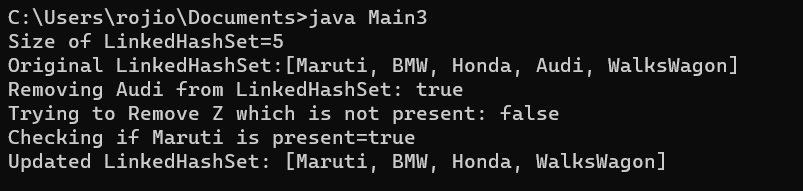
linkedset.contains("Maruti"));

System.out.println("Updated LinkedHashSet: " + linkedset);

}

}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO4 was obtained.

**Experiment No.: 30**

**Aim:** Write a Java program to compare two hash set.

**CO4**: Implement packages, exception handling, multithreading and generic programming by using the java.util package and Collection framework.

**Procedure**:

import java.util.\*;

public class Mainss{

public static void main(String[] args) {

int n;

String str;

HashSet<String> set1= new HashSet<String>();

System.out.println("HashSet 1");

System.out.println("Enter No. of countries:");

Scanner sc=new Scanner(System.in);

n=sc.nextInt();

System.out.println("Enter the name of countries:");

Scanner sc1=new Scanner(System.in);

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

str=sc1.nextLine();

set1.add(str);

}

System.out.println("HashSet 2");

HashSet<String> set2= new HashSet<String>();

System.out.println("Enter No. of countries:");

n=sc.nextInt();

System.out.println("Enter the name of countries:");

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

str=sc1.nextLine();

set2.add(str);

}

System.out.println("Set1:"+set1);

System.out.println("Set2:"+set2);

HashSet<String> a= new HashSet<String>(set1);

a.addAll(set2);

System.out.println("Union of country set:"+a);

HashSet<String> b= new HashSet<String>(set1);

b.retainAll(set2);

System.out.println("Intersection of country set:"+b);

HashSet<String> c= new HashSet<String>(set1);

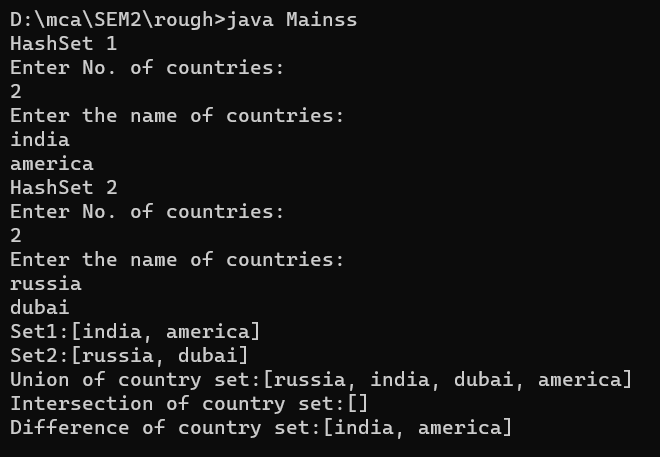
c.removeAll(set2);

System.out.println("Difference of country set:"+c);

}

}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO4 was obtained.

**Experiment No.: 31**

**Aim:** Program to demonstrate the working of Map interface by adding, changing and removing elements.

**CO4**: Implement packages, exception handling, multithreading and generic programming by using the java.util package and Collection framework.

**Procedure**:

// Java program to illustrate HashMap class of java.util

// package

// Importing HashMap class

import java.util.HashMap;

// Main class

public class Main {

// Main driver method

public static void main(String[] args)

{

// Create an empty hash map by declaring object

// of string and integer type

HashMap<String, Integer> map = new HashMap<>();

// Adding elements to the Map

// using standard add() method

map.put("Germany", 4);

map.put("England", 1);

map.put("Brazil", 5);

// Print size and content of the Map

System.out.println("Size of map is:- "

+ map.size());

// Printing elements in object of Map

System.out.println(map);

// Checking if a key is present and if

// present, print value by passing

// random element

if (map.containsKey("England")) {

// Mapping

Integer a = map.get("England");

// Printing value fr the corresponding key

System.out.println("value for key"

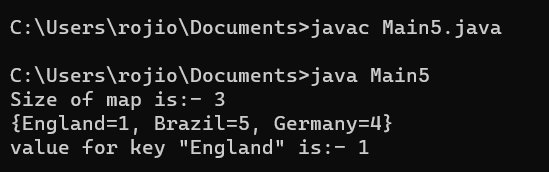
+ " \"England\" is:- " + a);

}

}

}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO4 was obtained.

**Experiment No.: 32**

**Aim:** Program to Convert HashMap to TreeMap .

**CO4**: Implement packages, exception handling, multithreading and generic programming by using the java.util package and Collection framework.

**Procedure**:

import java.util.\*;

public class Main {

public static void main(String[] args) {

// TODO Auto-generated method stub

Map<Integer,String> hm=new LinkedHashMap<>();

hm.put(1,"England");

hm.put(1,"Spain");

hm.put(2,"France");

hm.put(5,"Brazil");

hm.put(2,"Argentina");

System.out.println("HashMap:"+hm);

Map<Integer,String> tm=new TreeMap<>(hm);

System.out.println("\n \*\*\*\*\*\*\* Convert HashMap to TreeMap

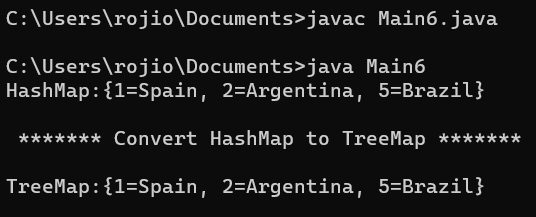
\*\*\*\*\*\*\*\n");

System.out.println("TreeMap:"+tm);

}

}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO4 was obtained.

**Experiment No.: 33**

**Aim:** Program to draw Circle, Rectangle, Line in Applet.

**CO5**: Develop applications to handle events using applets

**Procedure**:

import java.applet.Applet;

import java.awt.Color;

import java.awt.Graphics;

public class Shape extends Applet {

  public void paint(Graphics g) {

    g.drawOval(50, 50, 100, 100);

    g.setColor(Color.GREEN);

    g.drawRect(200, 50, 100, 150);

    g.setColor(Color.BLUE);

    g.drawLine(350, 50, 350, 200);

  }

}

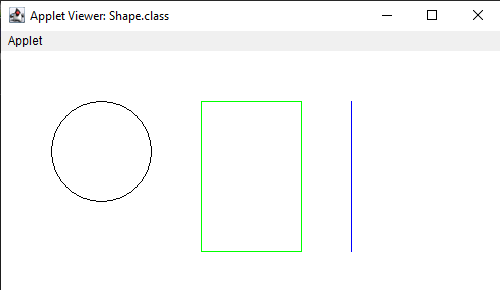
/\*

<applet code="Shape.class" width="500" height="500">

</applet>

\*/

**Output Screenshot**



**Result**

The program was executed and the result was successfully obtained. Thus CO5 was obtained.

**Experiment No.: 34**

**Aim:** Program to find maximum of three numbers using AWT.

**CO5**: Develop applications to handle events using applets

**Procedure**:

import java.awt.\*;

import java.awt.event.\*;

import java.applet.\*;

public class Maxthree extends Applet  implements ActionListener

{

    TextField t1 = new TextField(10);

    TextField t2 = new TextField(10);

    TextField t3 = new TextField(10);

    TextField t4 = new TextField(10);

    Label l1 = new Label("FIRST NUMBER=:");

    Label l2 = new Label("SECOND NUMBER:");

    Label l3 = new Label("THIRD NUMBER:");

    Label l4 = new Label("RESULT IS");

    Button b = new Button("Find MAXIMUM");

    public void init()

    {

        add(l1);

        add(t1);

        add(l2);

        add(t2);

        add(l3);

        add(t3);

        add(l4);

        add(t4);

        add(b);

        b.addActionListener(this);

    }

    public void actionPerformed(ActionEvent e)

    {

        if (e.getSource() == b)

        {

            int num1 = Integer.parseInt(t1.getText());

            int num2 = Integer.parseInt(t2.getText());

            int num3 = Integer.parseInt(t3.getText());

            if (num1 >= num2 && num1 >= num3)

                t4.setText("Result"+num1);

            else if (num2 >= num1 && num2 >= num3)

                t4.setText(""+num2);

            else

                t4.setText("Result"+num3);

        }

    }

}

<html>

<head>

<title> First Applet </title>

</head>

<body>

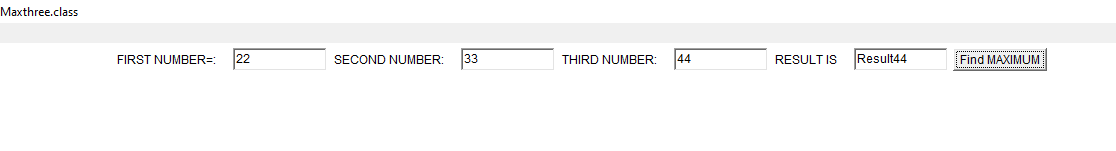
<APPLET CODE="Maxthree.class" width="400" height="400">

</applet>

</body>

</html>

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO5 was obtained.

**Experiment No.: 35**

**Aim:** Find the percentage of marks obtained by a student in 5 subjects. Display a happy face if he secures above 50% or a sad face if otherwise.

**CO5**: Develop applications to handle events using applets

**Procedure**:

import java.applet.\*;

import java.awt.\*;

import java.awt.Graphics;

import java.awt.event.\*;

public class smileyy extends Applet implements ActionListener {

Label l1,l2,l3,l4,l5,l6;

TextField t1,t2,t3,t4,t5,t6;

Button b;

public void init(){

l1 = new Label("MARK 1:");

t1 = new TextField();

l2 = new Label("MARK 2:");

t2 = new TextField();

l3 = new Label("MARK 3:");

t3 = new TextField();

l4 = new Label("MARK 4:");

t4 = new TextField();

l5 = new Label("MARK 5:");

t5 = new TextField();

l6 = new Label("PERCENTAGE:");

t6 = new TextField();

b = new Button("STATUS");

setLayout(null);

l1.setBounds(450,50,70,20);

t1.setBounds(520,50,100,20);

l2.setBounds(450,80,70,20);

t2.setBounds(520,80,100,20);

l3.setBounds(450,110,70,20);

t3.setBounds(520,110,100,20);

l4.setBounds(450,140,70,20);

t4.setBounds(520,140,100,20);

l5.setBounds(450,170,70,20);

t5.setBounds(520,170,100,20);

l6.setBounds(450,200,100,20);

t6.setBounds(550,200,100,20);

b.setBounds(450,290,80,30);

add(l1);

add(l2);

add(l3);

add(l4);

add(l5);

add(l6);

add(t1);

add(t2);

add(t3);

add(t4);

add(t5);

add(t6);

add(b);

b.addActionListener(this);

}

public void actionPerformed(ActionEvent e){

float m1, m2,m3, m4,m5,percent;

m1= Float.parseFloat(t1.getText());

m2= Float.parseFloat(t2.getText());

m3= Float.parseFloat(t3.getText());

m4= Float.parseFloat(t4.getText());

m5= Float.parseFloat(t5.getText());

percent=((m1+m2+m3+m4+m5)\*100)/500;

t6.setText(String.valueOf(percent));

repaint();

}

public void paint(Graphics g){

float p;

p= Float.parseFloat(t6.getText());

if(p> 50.0) {

g.setColor(Color.ORANGE);

g.fillOval(470,400,100,100);

g.setColor(Color.black);

g.fillOval(500,418,10,10);

g.fillOval(525,418,10,10);

g.setColor(Color.black);

g.fillArc (493,435,50,50,0,-180);

}

else {

g.setColor(Color.ORANGE);

g.fillOval(470,400,100,100);

g.setColor(Color.black);

g.fillOval(500,418,10,10);

g.fillOval(525,418,10,10);

g.setColor(Color.black);

g.drawArc(493,440,50,50,0,180);

}

}}

HTML FILE

<html>

<head>

</head>

<title> APPLET SMILE/SAD FACE  </title>

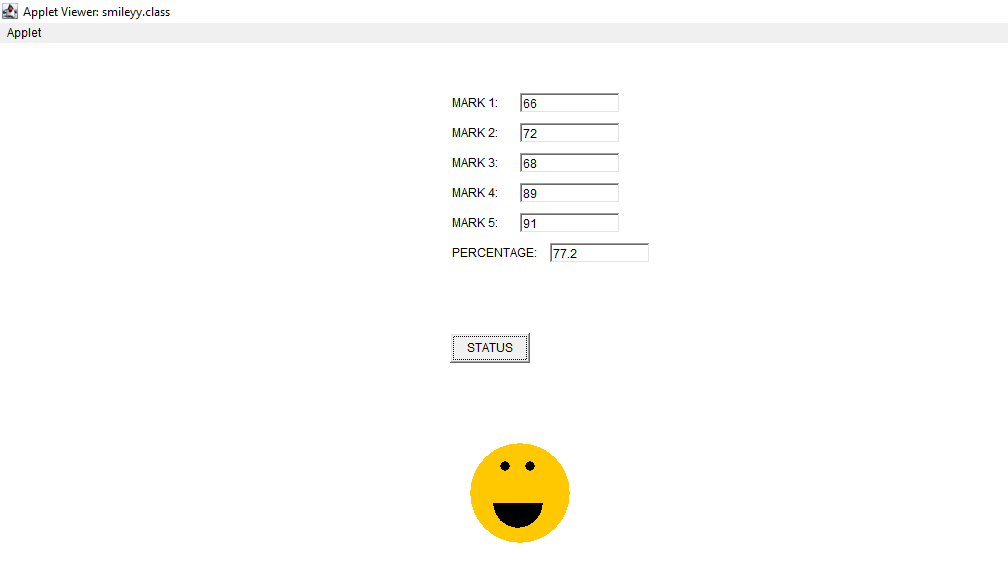
<body>

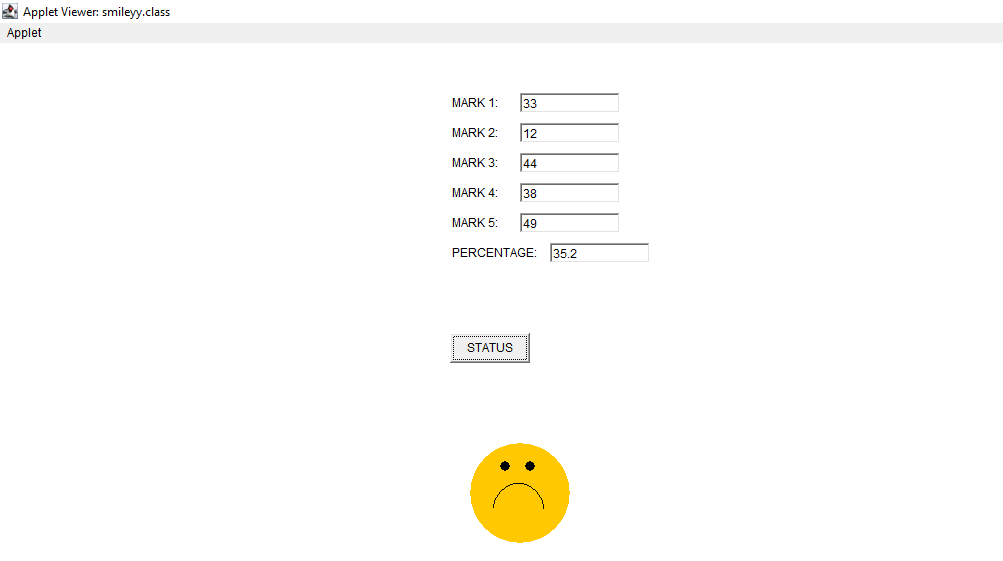
<applet code="smileyy.class" height="300" width="300">  </applet>

</body>

</html>

**Output Screenshot**





**Result**

The program was executed and the result was successfully obtained. Thus CO5 was obtained.

**Experiment No.: 36**

**Aim:** Using 2D graphics commands in an Applet, construct a house. On mouse click event, change the color of the door from blue to red.

**CO5**: Develop applications to handle events using applets

**Procedure**:

import java.awt.\*;

import java.applet.\*;

import java.awt.event.\*;

public class Homee extends Applet implements MouseListener

{

 int a,b;

 public void init()

 {

 addMouseListener( this);

 }

 public void paint(Graphics g)

 {

 int x[]={150,300,225};

 int y[]={150,150,25};

 g.drawPolygon(x,y,3);

 g.setColor(Color.green);

 g.fillPolygon(x,y,3);

 g.drawRect(150,150,150,200);//House

 g.setColor(Color.orange);

 g.fillRect(150,150,150,200);

 g.drawRect(200,200,50,150);//Door

 g.setColor(Color.blue);

 g.fillRect(200,200,50,150);

 if(a>200 && a<300 && b>200 && b<300)

 {

 g.setColor(Color.red);

 g.fillRect(200, 200, 50, 150);

 }

 }

 public void mouseClicked(MouseEvent e)

 {

 }

 public void mouseEntered(MouseEvent e)

 {

 }

 @Override

 public void mouseExited(MouseEvent e) {

 }

 public void mousePressed(MouseEvent e)

 {

 a=e.getX();

 b=e.getY();

 repaint();

 }

 public void mouseReleased(MouseEvent e)

 {

 }

}

HTML FILE

<html>

<body>

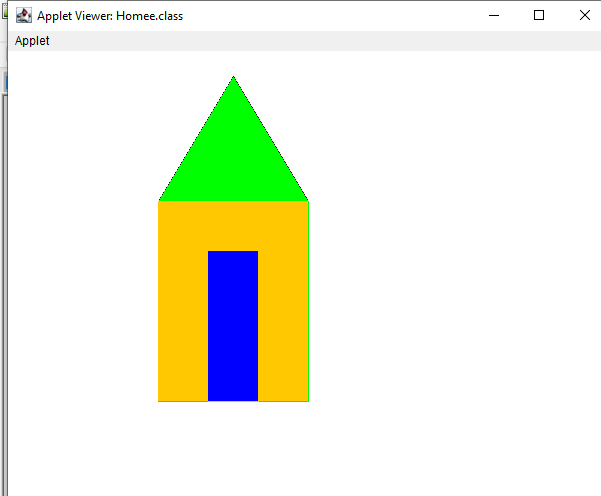
<applet code="Homee.class" width="600" height="600">

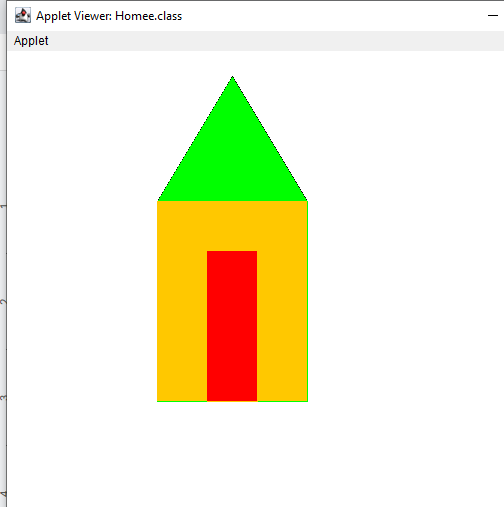
</applet>

</body>

</html>

**Output Screenshot**





**Result**

The program was executed and the result was successfully obtained. Thus CO5 was obtained.

**Experiment No.: 37**

**Aim:** Develop a program that has a Choice component which contains the names of shapes such as rectangle, triangle, square and circle. Draw the corresponding shapes for given parameters as per user’s choice.

**CO5**: Develop applications to handle events using applets

**Procedure**:

import java.applet.\*;

import java.awt.\*;

import java.awt.Graphics;

import java.awt.event.\*;

public class Shape extends Applet implements ItemListener

{

Choice figure = new Choice();

int Select;

public void init()

{

figure.addItem("Select your choice");

figure.addItem("Rectangle");

figure.addItem("Square");

figure.addItem("Circle");

figure.addItem("Triangle");

add(figure);

figure.addItemListener(this);

}

public void itemStateChanged (ItemEvent e)

{

Select = figure.getSelectedIndex();

repaint();

}

public void paint(Graphics g)

{

g.setColor(Color.red);

super.paint(g);

if (Select == 1)

{

g.drawRect(280, 100, 160,40);

}

if (Select == 2)

{

g.drawRect(50,50,100,100);

}

if (Select == 3)

{

g.drawOval(150,150,100,100);

}

if (Select ==4)

{

g.drawLine(120, 130, 280, 130);

g.drawLine(120, 130, 200, 65);

g.drawLine(200, 65, 280, 130);

}}}

<html>

<body>

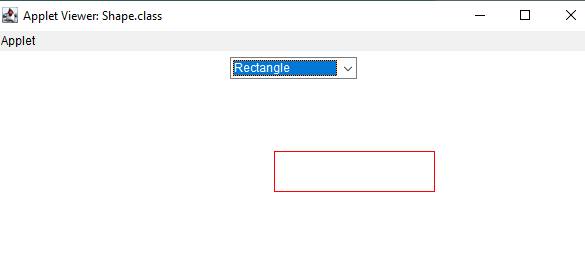
<applet code="Shape.class" width="600" height="600">

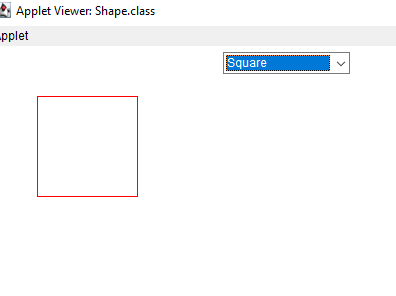
</applet>

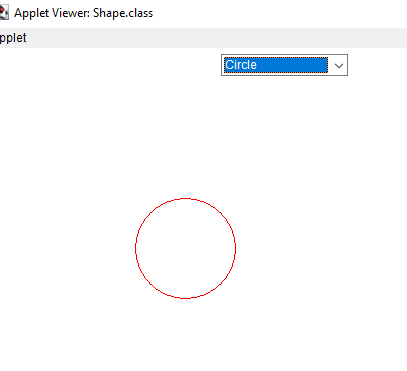
</body>

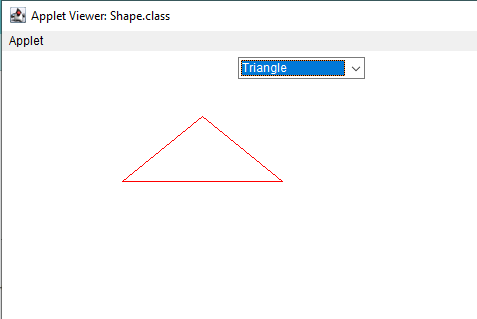
</html>

**Output Screenshot**









**Result**

The program was executed and the result was successfully obtained. Thus CO5 was obtained.

**Experiment No.: 38**

**Aim:** Develop a program to handle all mouse events and window events.

**CO5**: Develop applications to handle events using applets

**Procedure**:

import java.awt.\*;

import java.applet.\*;

import java.awt.event.\*;

public class co5q7 extends Applet implements

MouseListener,MouseMotionListener

{

int mx=0;

int my=0;

String msg="";

public void init()

{

addMouseListener(this);

addMouseMotionListener(this);

}

public void mouseClicked(MouseEvent me)

{

mx=20;

my=40;

msg="Mouse Clicked";

repaint();

}

public void mousePressed(MouseEvent me)

{

mx=30;

my=60;

msg="Mouse Pressed";

repaint();

}

public void mouseReleased(MouseEvent me)

{

mx=30;

my=60;

msg="Mouse Released";

repaint();

}

public void mouseEntered(MouseEvent me)

{

mx=40;

my=80;

msg="Mouse Entered";

repaint();

}

public void mouseExited(MouseEvent me)

{

mx=40;

my=80;

msg="Mouse Exited";

repaint();

}

public void mouseDragged(MouseEvent me)

{

mx=me.getX();

my=me.getY();

showStatus("Currently mouse dragged"+mx+" "+my);

repaint(); }

public void mouseMoved(MouseEvent me)

{

mx=me.getX();

my=me.getY();

showStatus("Currently mouse is at"+mx+" "+my);

repaint();

}

public void paint(Graphics g)

{

g.drawString("Handling Mouse Events",30,20);

g.drawString(msg,60,40);

}

}

<html>

<body>

<applet code="co5q7.class" width="600" height="600">

</applet>

</body>

</html>

**Output Screenshot**



**Result**

The program was executed and the result was successfully obtained. Thus CO5 was obtained.

**Experiment No.: 39**

**Aim:** Write a program to write to a file, then read from the file and display the contents on the console.

**CO6**: Design applications using files and network concepts.

**Procedure**:

import java.io.BufferedReader;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

public class co63 {

public static void main(String[] args) {

try {

FileWriter writer = new FileWriter("java\_write.txt",true);

writer.write("new file is created");

writer.close();

FileReader reader = new FileReader("java\_write.txt");

BufferedReader br= new BufferedReader(reader);

String line;

System.out.println("Data read from the file");

while ((line = br.readLine()) != null) {

System.out.println(line);

}

reader.close();

} catch (IOException e) {

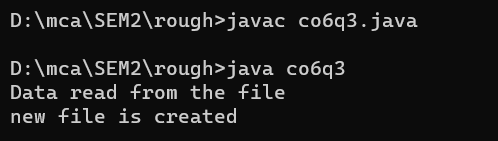
System.out.println("-----Error-----");

}

}

}

**Output Screenshot**

****

**Result**

The program was executed and the result was successfully obtained. Thus CO6 was obtained.

**Experiment No.: 40**

**Aim:** Write a program to copy one file to another.

**CO6**: Design applications using files and network concepts.

**Procedure**:

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.IOException;

public class co6q2 {

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

FileInputStream fileinput = new FileInputStream("source.txt");

FileOutputStream fileoutput = new FileOutputStream("destination.txt");

int i;

while((i = fileinput.read()) != -1){

fileoutput.write(i);

}

System.out.println("copied");

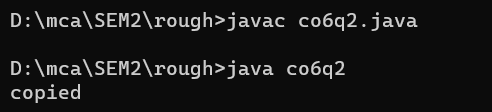
fileinput.close();

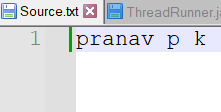
fileoutput.close();

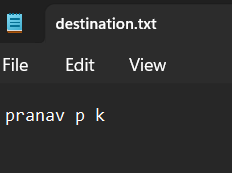
}

}

**Output Screenshot**

****

****

****

**Result**

The program was executed and the result was successfully obtained. Thus CO6 was obtained.

**Experiment No.: 41**

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

**CO6**: Design applications using files and network concepts.

**Procedure**:

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.IOException;

public class co61 {

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

FileInputStream source = new FileInputStream ("Source1.txt");

FileOutputStream destination\_odd = new FileOutputStream ("odd.txt");

FileOutputStream destination\_even = new FileOutputStream

("even.txt");

int i;

while((i = source.read()) != -1){

if(i%2==0) {

destination\_even.write(i);

}

else {

destination\_odd.write(i);

}}

System.out.println("copied");

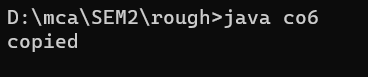
source.close();

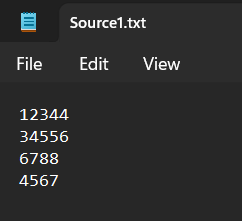
destination\_even.close();

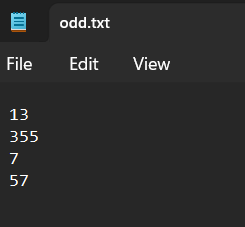
destination\_odd.close();

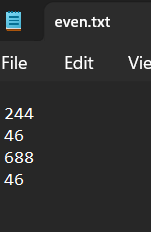
}}

**Output Screenshot**

****

****

****

****

**Result**

The program was executed and the result was successfully obtained. Thus CO6 was obtained.

**Experiment No.: 42**

**Aim:** Client Server communication using DatagramSocket - UDP

**CO6**: Design applications using files and network concepts.

**Procedure**:

import java.util.\*;

import java.io.\*;

import java.net.\*;

public class client{

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

DatagramSocket client=new DatagramSocket();

InetAddress add=InetAddress.getByName("localhost");

Scanner sc=new Scanner(System.in);

System.out.println("Message server:");

String str=sc.next();

byte[] bufBytes=str.getBytes();

DatagramPacket datagramPacket=new DatagramPacket(bufBytes,bufBytes.length,add,4220);

client.send(datagramPacket);

client.close();

}

}

import java.io.\*;

import java.net.\*;

public class server{

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

DatagramSocket server=new DatagramSocket(4220);

//InetAddress add=InetAddress.getByName("localhost");

//String str="Ping from Client!!!";

byte[] buf=new byte[256];

DatagramPacket packet=new DatagramPacket(buf,buf.length);

server.receive(packet);

String response=new String(packet.getData());

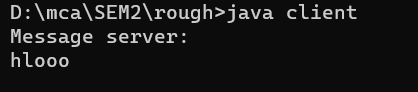
System.out.println("Server:"+response);

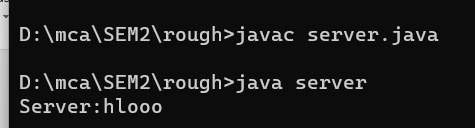
server.close();

}

}

**Output Screenshot**

****

****

**Result**

The program was executed and the result was successfully obtained. Thu s CO6 was obtained.