**P.A. AZIZ COLLEGE OF ENGINEERING AND TECHNOLOGY (PAACET)**

Green Hills, Karakulam P.O., Thiruvananthapuram, Kerala 695564



**LABORATORY MANUAL**

**ADVANCED ABMS LAB**

**SUBJECT CODE: 20MCA134**

**MASTER OF COMPUTER APPLICATION DEPARTMENT**

**MCA 2nd SEMESTER**

**NAME: ACHSAH MARIYAM GEORGE**

**REGISTER NO: PAA21MCA-2001**

**YEAR: ACADEMIC YEAR 2021-’23**

**P.A. AZIZ COLLEGE OF ENGINEERING AND TECHNOLOGY**

GREEN HILLS, KARAKULAM P.O., THIRUVANANTHAPURAM, KERALA 695564



**CERTIFICATE**

***This is to certified that Mrs. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (Register No. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_) has satisfactorily completed the course in \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_as by the APJ Abdul Kalam University for \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ year, of semester \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ of Master of Computer Applications in the Academic year \_\_\_\_\_\_\_\_\_\_\_\_\_\_.***

**Head of the Department (MCA) Faculty-in-charge**

**Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Internal Examiner External Examiner**

**CONTENT**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl. No.** | **Date** | **Name of Experiment** | **Page No.** | **Remarks** |
| 1. | 21/03/2022 | To find the product having the lowest price | 4 |  |
| 2. | 21/03/2022 | Read 2 matrices from the console and perform matrix addition | 5 |  |
| 3. | 28/03/2022 | Add complex numbers | 7 |  |
| 4. | 28/03/2022 | Create CPU with attribute and its Objects – RAM and Processor | 8 |  |
| 5. | 04/04/2022 | Perform string manipulations | 10 |  |
| 6. | 04/04/2022 | Program to sort the strings | 13 |  |
| 7. | 11/04/2022 | Search an element in an array | 15 |  |
| 8. | 11/04/2022 | Area of different shapes using overloaded functions | 17 |  |
| 9. | 18/04/2022 | Use array of objects to display details of n teachers | 20 |  |
| 10. | 18/04/2022 | Display the academic and sports score of a student | 22 |  |
| 11. | 25/04/2022 | Compare the area and color of two rectangles | 24 |  |
| 12. | 25/04/2022 | Menu driven program to find area of objects | 26 |  |
| 13. | 02/05/2022 | Test the package by implementing all operations on two given numbers | 28 |  |
| 14. | 02/05/2022 | Find the average of positive integers, raising a user defined exception for each negative input | 30 |  |
| 15. | 09/05/2022 | Generate a multiplication table and displaying N prime numbers using threads | 32 |  |
| 16. | 09/05/2022 | Generate Fibonacci numbers and displaying even numbers in a given range using threads | 34 |  |
| 17. | 23/05/2022 | Write a user defined exception class to authenticate the username and password | 36 |  |
| 18. | 23/05/2022 | Write a program to compare two hash set | 37 |  |
| 19. | 30/05/2022 | Implement a simple calculator using AWT components | 38 |  |
| 20. | 30/05/2022 | Program to find maximum of three numbers using AWT | 40 |  |
| 21. | 06/06/2022 | Program that reads from a file having integers. Copy even and odd numbers to separate files | 42 |  |
| 22. | 13/06/2022 | Program to write to a file, then read from the file and display the contents on the console | 44 |  |

**Program : 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.

**Algorithm** :

Step 1: Start.

Step 2: Define a class having name Product and members as pcode, pname and price.

Step 3: Declare three objects in the class and add the values of each data members into objects.

Step 4: Using if condition check which object has the lowest price and print it.

Step 5: Stop.

**Program Code** :

public class product {

String pcode;

String pname;

int price;

public product(String pcode\_get, String pname\_get, int price\_get) {

pcode = pcode\_get;

pname = pname\_get;

price = price\_get;

}

public int get\_price() {

return price;

}

public static void main(String[] args) {

product p\_1 = new product("A123", "TV", 2);

product p\_2 = new product("B123", "RADIO", 21);

product p\_3 = new product("C123", "DVD", 3);

if (p\_1.price <= p\_3.price && p\_1.price <= p\_2.price)

System.out.println("Lowest product ID is : " + p\_1.pcode + " " + p\_1.pname);

if (p\_3.price <= p\_1.price && p\_3.price <= p\_2.price)

System.out.println("Lowest product ID is : " + p\_3.pcode + " " + p\_3.pname);

if (p\_2.price <= p\_3.price && p\_2.price <= p\_1.price)

System.out.println("Lowest product ID is : " + p\_2.pcode + " " + p\_2.pname);

}

}

**Result** : The above program is executed and obtained the output.

**Program : 2**

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

**Algorithm** :

Step 1: Start.

Step 2: Define a class having name AddMatrix.

Step 3: Read row number(m),column number (n) and initialize the double dimensional arrays mat1[][],mat2[][],res[][] with same row number ,column number.

Step 4: Store the first matrix elements into the two-dimensional array matrix mat1[][] using two for loops. i indicates row number, j indicates column index. Similarly second matrix elements in to mat2[][].

Step 5: Add the two matrices using for loop. for i=0 to i

**Program Code** :

import java.util.\*;

public class matrix {

int row;

int column;

int[][] array = new int[10][10];

public void get\_metrix(){

int rc,cc;

Scanner sc= new Scanner(System.in);

System.out.print("Enter size of matrix , row count : ");

this.row = sc.nextInt();

System.out.print("Enter size of matrix , column count : ");

this.column = sc.nextInt();

System.out.print("Enter matrix elements : ");

for(rc=0;rc<this.row;rc++){

for(cc=0;cc<this.column;cc++){

this.array[rc][cc] = sc.nextInt();

}

}

}

public static matrix sum(matrix c1, matrix c2) {

int rc, cc;

matrix temp = new matrix();

if (c1.row == c2.row && c1.column == c2.column) {

temp.row =c1.row;

temp.column = c1.column;

for (rc = 0; rc < c1.row; rc++) {

for (cc = 0; cc < c1.column; cc++) {

temp.array[rc][cc] = c1.array[rc][cc] + c2.array[rc][cc];

}

}

}

else {

System.out.println("Addition not possible ");

}

return temp;

}

public void display\_matrix(){

int rc,cc;

for(rc=0;rc<this.row;rc++){

for(cc=0;cc<this.column;cc++){

System.out.print(this.array[rc][cc] );

}

System.out.println("");

}

}

public static void main(String[] args) {

matrix first = new matrix();

matrix second = new matrix();

matrix temp = sum(first, second);

first.get\_metrix();

second.get\_metrix();

temp = sum(first,second);

//first.display\_matrix();

//second.display\_matrix();

System.out.println("......SUM MATRIX.......");

temp.display\_matrix();

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

}

}

**Result** : The above program is executed and obtained the output.

**Program : 3**

**Aim** : Add complex numbers.

**Algorithm** :

Step 1: Start.

Step 2: Define a class having name ComplexNumber and data members are real and imaginary number.

Step 3: Define a function ComplexNumber and add values to variables.

Step 4 : Define a function ComplexNumber sum to add complex number and return the value.

Step 5: Print the sum value.

Step 6: Stop.

**Program Code** :

public class complex {

int real;

int imaginary;

public complex(int real\_get, int imag\_get){

this.real = real\_get;

this.imaginary = imag\_get;

}

public static complex sum(complex c1, complex c2) {

complex temp = new complex(0,0);

temp.real = c1.real + c2.real;

temp.imaginary = c1.imaginary + c2.imaginary;

return temp;

}

public static void main(String []args) {

complex first = new complex(1,2);

complex second = new complex(32,22);

complex temp = sum(first,second);

System.out.println("Sum is : " + temp.real + " + " +temp.imaginary + "i");

}

}

**Result** : The above program is executed and obtained the output.

**Program : 4**

**Aim** : Create CPU with attribute price. Create 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** :

Step 1: Start.

Step 2: Define a class cpu with data member price and class processor.

Step 3: Class processor contain data members no\_cores, manufacturer and a nested class RAM.

Step 4: class RAM contain memory and manufacturer as data members.

Step 5: Create objects in corresponding classes and display its details.

Step 6: Stop.

**Program Code** :

class Cpu {

int price;

Cpu(int p) {

this.price = p;

}

class Processor {

int cores;

String manufacture;

Processor(int n, String m) {

this.cores = n;

this.manufacture = m;

}

void display() {

System.out.println("No of Cores : " + this.cores);

System.out.println("Processor manufactures : " + this.manufacture);

}

}

static class Ram {

int memory;

String manufacture;

Ram(int n, String m) {

this.memory = n;

this.manufacture = m;

}

void display() {

System.out.println("Memory Size : " + this.memory);

System.out.println("Memory manufactures : " + this.manufacture);

}

}

void display() {

System.out.println("Price of CPU : " + this.price);

}

public static void main(String[] args) {

Cpu intel = new Cpu(23000);

Cpu.Processor i\_processor = intel.new Processor(4, "intel");

Cpu.Ram i\_ram = new Ram(1024, "Asus");

intel.display();

i\_processor.display();

i\_ram.display();

}

}

**Result** : The above program is executed and obtained the output.

**Program : 5**

**Aim** : Perform string manipulations

**Algorithm** :

Step 1: Start

Step 2: Take the strings provided by the user and concatenate them.

Step 3: Display the String.

Step 4: Display the count of each word in the string.

Step 5: Display the replaced word of string.

Step 6: Display the string after reversing the each word.

Step 7: Stop

**Program Code** :

import java.util.\*;

class stringMenu {

String sentance;

public void read(){

Scanner sc = new Scanner(System.in);

sentance = sc.nextLine();

}

public void count(){

String sentannce = this.sentance;

String[] arrOfStr = sentannce.split(" ");

//System.out.println("<- After split up ->");

//for (String a : arrOfStr)

//System.out.println(a);

Set<String> set = new HashSet<>(Arrays.asList(arrOfStr));

//System.out.println("<- After convert to set ->");

//for (String a : set)

//System.out.println(a);

for(String each:set){

int count=0;

for(String eacha:arrOfStr){

if((each.compareTo(eacha)) == 0){

count++;

}

}

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

}

}

public void replace(){

Scanner sc = new Scanner(System.in);

System.out.print("Enter the word that need to replace : ");

String word = sc.nextLine();

System.out.print("Enter the word that need to place : ");

String rword = sc.nextLine();

String sentannce = this.sentance;

String sent="";

String[] arrOfStr = sentannce.split(" ");

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

if((arrOfStr[i].compareTo(word))==0){

arrOfStr[i] = rword;

}

sent = sent + arrOfStr[i] + " ";

}

this.sentance = sent;

}

public void reverse(){

String sentannce = this.sentance;

String sent="";

String temp;

String[] arrOfStr = sentannce.split(" ");

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

temp="";

String[] temparray = arrOfStr[i].split("");

for(int j=arrOfStr[i].length()-1;j>=0;j--){

temp = temp + temparray[j];

}

sent = sent + temp + " ";

}

this.sentance = sent;

}

public String dis(){

return sentance;

}

public static void main(String[] args) {

int opt;

stringMenu Obj = new stringMenu();

while(true){

Scanner sc = new Scanner(System.in);

System.out.println("1) Enter Sentence");

System.out.println("2) Display Sentence");

System.out.println("3) Count of each word");

System.out.println("4) Replace word");

System.out.println("5) Reverse each word ");

System.out.println("6) Exit");

System.out.print("Option :: ");

opt = Integer.parseInt(sc.nextLine());

switch(opt){

case 1:

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

Obj.read();

break;

case 2:

System.out.println("Sentance : "+Obj.dis());

break;

case 3:

System.out.println("Counts");

Obj.count();

break;

case 4:

System.out.println("Replace words");

Obj.replace();

break;

case 5:

System.out.println("Reverse each word");

Obj.reverse();

break;

case 6:

System.exit(0);

default:

System.out.println("try another menu!");

}

}

}

}

**Result** : The above program is executed and obtained the output.

**Program : 6**

**Aim** : Program to Sort strings

**Algorithm** :

Step 1: Start

Step 2: Select the first element of the list (i.e., Element at first position in the list).

Step 3: Compare the selected element with all the other elements in the list.

Step 4: In every comparison, if any element is found smaller than the selected element (for Ascending order), then both are swapped.

Step 5: Repeat the same procedure with element in the next position in the list till the entire list is sorted.

Step 6: Stop

**Program Code** :

class Book {

int ISBN;

String title;

String author;

int price;

String Publisher;

Book(int isbn, String title, String author, int price, String publisher) {

this.ISBN = isbn;

this.title = title;

this.author = author;

this.price = price;

this.Publisher = publisher;

}

public void display() {

System.out.print(this.ISBN + "\t");

System.out.print(this.title + "\t");

System.out.print(this.author + "\t");

System.out.print(this.price + "\t");

System.out.print(this.Publisher + "\t");

System.out.println();

}

public static void main(String []args){

Book books[] = new Book[5];

books[0] =new Book(1,"A","S",11,"AD");

books[1] =new Book(2,"B","D",12,"AA");

books[2] =new Book(3,"AC","F",13,"AA");

books[3] =new Book(4,"E","A",14,"AA");

books[4] =new Book(5,"DA","G",15,"AA");

int size = 5;

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

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

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

Book temp = books[i];

books[i] = books[j];

books[j] = temp;

}

}

}

for(int i=0;i<5;i++){

books[i].display();

}

}

}

**Result** : The above program is executed and obtained the output.

**Program : 7**

**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.

**Algorithm** :

Step 1: Start

Step 2: Search the ‘eNo’ attribute of the list of Employee Objects for the ‘eNo’ provided by the user.

Step 3: If user provided ‘eNo’ is found inside the Employee object list, display the details of the corresponding employee.

Step 4: Stop

**Program Code** :

import java.util.\*;

class employee {

int eNo;

String eName;

int eSalary;

public void read(){

Scanner sc= new Scanner(System.in);

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

eNo = Integer.parseInt(sc.nextLine());

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

eName = sc.nextLine();

System.out.print("Enter monthly salary : ");

eSalary = Integer.parseInt(sc.nextLine());

}

public void display(){

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

}

public static void main(String []args){

int i,n=3;

int No;

employee emp[] = new employee[n];

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

emp[i] = new employee();

emp[i].read();

}

System.out.println("Search");

while(true){

Scanner sc= new Scanner(System.in);

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

No = Integer.parseInt(sc.nextLine());

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

if(emp[i].eNo == No){

emp[i].display();

break;

}

}

}

}

}

**Result** : The above program is executed and obtained the output.

**Program : 8**

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

**Algorithm** :

Step 1: Start

Step 2: Define the main class

Step 3: Define methods with the same methodname that performs the area operation for each shape

Step 4: Display the areas of each shapes.

**Program Code** :

import java.util.\*;

import java.lang.\*;

interface Shape {

float pi = 3.14F;

float area();

float perimeter();

}

class Circle implements Shape {

Scanner sc = new Scanner(System.in);

int r;

public float area() {

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

r = Integer.parseInt(sc.nextLine());

return (pi \* r \* r);

}

public float perimeter() {

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

r = Integer.parseInt(sc.nextLine());

return (2 \* pi \* r);

}

}

class Rectangle implements Shape {

Scanner sc = new Scanner(System.in);

int l, b;

public float area() {

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

l = Integer.parseInt(sc.nextLine());

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

b = Integer.parseInt(sc.nextLine());

return (l \* b);

}

public float perimeter() {

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

l = Integer.parseInt(sc.nextLine());

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

b = Integer.parseInt(sc.nextLine());

return (2 \* (l + b));

}

}

class Square implements Shape {

Scanner sc = new Scanner(System.in);

int a;

public float area() {

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

a = Integer.parseInt(sc.nextLine());

return (a \* a);

}

public float perimeter() {

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

a = Integer.parseInt(sc.nextLine());

return (4 \* a);

}

}

class Area {

public static void main(String args[]) {

Scanner sc = new Scanner(System.in);

Circle c = new Circle();

Rectangle r = new Rectangle();

Square s = new Square();

int ch;

while (true) {

System.out.println("1:Area of Circle");

System.out.println("2:Perimeter of Circle");

System.out.println("3:Area of Rectangle");

System.out.println("4:Perimter of Rectangle");

System.out.println("5:Area of Square");

System.out.println("6:Perimter of Square");

System.out.println("7:EXIT");

System.out.println("enter choice ");

ch = Integer.parseInt(sc.nextLine());

switch (ch) {

case 1:

float ar = c.area();

System.out.println("Area : " + ar);

break;

case 2:

float pr = c.perimeter();

System.out.println(pr);

break;

case 3:

float a = r.area();

System.out.println("Area : " + a);

break;

case 4:

float pr1 = r.perimeter();

System.out.println(pr1);

break;

case 5:

float are = s.area();

System.out.println("Area : " + are);

break;

case 6:

float pr2 = s.perimeter();

System.out.println(pr2);

break;

case 7:

System.out.println("Exiting the Program");

System.exit(0);

default:

System.out.println("invalid!");

}

}

}

}

**Result** : The above program is executed and obtained the output.

**Program : 9**

**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** :

Step 1: Start

Step 2: create class “employee” with the provided data members and define the constructors

Step 3: create another class “Teachers” that performs inheritance of employee class and define constructors for the same

Step 4: create an array of objects in the corresponding class

Step 5: Display the details for the number of teachers provided.

**Program Code** :

class Employees {

String Empid;

String Name;

String Address;

int Salary;

Employees(String id,String name,String addr,int salary){

this.Empid = id;

this.Name = name;

this.Address = addr;

this.Salary = salary;

}

void display(){

System.out.println("EmpID : " + this.Empid);

System.out.println("Name : " + this.Name);

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

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

}

}

class Teacher extends Employees{

String Department;

String Subject;

Teacher(String id,String name,String addr,int salary,String dept,String subj){

super(id,name,addr,salary);

this.Department=dept;

this.Subject=subj;

}

void display(){

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

super.display();

System.out.println("Dept Name : " + this.Department);

System.out.println("Subject Name : " + this.Subject);

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

}

}

class Main {

public static void main(String[] args) {

Teacher teacObj[] = new Teacher[2];

teacObj[0] = new Teacher("1","A","AA",11,"AAA","AAAA");

teacObj[1] = new Teacher("2","B","BB",11,"BBB","BBBB");

teacObj[0].display();

teacObj[1].display();

}

}

**Result** : The above program is executed and obtained the output.

**Program : 10**

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

**Algorithm** :

Step 1: Start

Step 2:Create a class named ‘Student’ with data members as subject names; a constructor.

Step 3: Create a class named ‘Sports’ which is derived ‘Student’ with data members goals,matches,won; a constructor.

Step 4: Create a class named ‘Result’ which is derived from Sports member functions displayStud() and displaySport() to display details.

Step 5: Create an 3 objects of type Student, Sports and Result, and display the details.

Step 6: Stop

**Program Code** :

import java.util.\*;

import java.io.\*;

class Sports {

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

InputStreamReader isr = new InputStreamReader(System.in);

BufferedReader br = new BufferedReader(isr);

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

String n = br.readLine();

System.out.print("Enter roll no : ");

int roll = Integer.parseInt(br.readLine());

System.out.print("Enter mark in first subject : ");

float m1 = Float.parseFloat(br.readLine());

System.out.print("Enter mark in second subject : ");

float m2 = Float.parseFloat(br.readLine());

Result re = new Result(n, roll, m1, m2);

re.sports\_getData();

re.display();

re.academic();

re.sports\_dispData();

}

}

interface Sport {

public void sports\_getData();

public void sports\_dispData();

}

class Student {

String name;

int roll\_no;

float mark1, mark2;

Student(String n, int r, float m1, float m2) {

name = n;

roll\_no = r;

mark1 = m1;

mark2 = m2;

}

void display() {

System.out.println("Student Details");

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

System.out.println("Roll No. of Student: " + roll\_no);

System.out.println("Marks of Subject 1: " + mark1);

System.out.println("Marks of Subject 2: " + mark2);

}

}

class Result extends Student implements Sport {

int r;

String item;

Scanner sc = new Scanner(System.in);

Result(String n, int r, float m1, float m2) {

super(n, r, m1, m2);

}

public void academic() {

float total = (mark1 + mark2);

float percent = total \* 100 / 200;

System.out.println("\_\_\_Academic Info\_\_\_");

System.out.println("Percentage: " + percent + "%");

}

public void sports\_getData() {

System.out.print("Enter the sports item which student participated : ");

item = sc.nextLine();

System.out.print("Enter the rank position that the obtained : ");

r = sc.nextInt();

}

public void sports\_dispData() {

System.out.println("\_\_\_\_sports Info\_\_\_\_");

System.out.println("Item :" + item);

System.out.println("Rank :" + r);

}

}

**Result** : The above program is executed and obtained the output.

**Program : 11**

**Aim** : Compare the area and color of two rectangles

**Algorithm** :

Step 1: Start

Step 2: Create a class for find the areas.

Step 3: Create objects for both these classes and call functions color() and area() to display the same.

Step 4: Stop.

**Program Code** :

import java.util.\*;

class rectangle {

double width;

double length;

double area;

String color;

rectangle() {

}

rectangle(double w, double l, String c) {

this.width = w;

this.length = l;

this.area = w \* l;

this.color = c;

}

public void read() {

Scanner sc = new Scanner(System.in);

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

this.color = sc.nextLine();

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

this.length = sc.nextDouble();

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

this.width = sc.nextDouble();

this.area = this.length \* this.width;

}

public String get\_color() {

System.out.println("color : " + this.color);

return this.color;

}

public double get\_area() {

System.out.println("area : " + this.area);

return this.area;

}

public static void main(String[] args) {

rectangle rectangle\_1 = new rectangle(10, 5, "red");

rectangle rectangle\_2 = new rectangle();

rectangle\_2.read();

if ((rectangle\_1.get\_area() == rectangle\_2.get\_area()) && (rectangle\_1.get\_color().equals(rectangle\_2.get\_color()))) {

System.out.println("rectangles are same");

} else {

System.out.println("rectangles are not same");

}

}

}

**Result** : The above program is executed and obtained the output.

**Program : 12**

**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.

**Algorithm** :

Step 1: Start

Step 2: To create a package named graphics, create a folder of the same name in the directory. Here inside that we have another module named calculate

Step 3: Inside the graphics folder, create modules for finding the areas of rectangle, circle, triangle and square.

Step 4: Outside the graphics folder, write a program to access the modules mention above and print the output.

Step 5: Stop

**Program Code** :

package\_graphics.java

interface interface\_graphics{

public float recArea(int l, int h);

public float cirArea(int r);

public float squArea(int a);

public float triArea(int l, int h);

}

public class package\_graphics implements interface\_graphics {

public float recArea(int l, int h){

return l\*h;

}

public float cirArea(int r){

return r\*r\*(float)3.14;

}

public float squArea(int a){

return a\*a;

}

public float triArea(int l, int h){

return l\*h\*(float)(.5);

}

}

Main\_graphics.java

//import mca.myapp.CO4.package\_graphics.\*;

public class main\_graphics {

public static void main(String []args){

package\_graphics testObj = new package\_graphics();

System.out.println(testObj.recArea(1,2));

System.out.println(testObj.cirArea(10));

System.out.println(testObj.squArea(10));

System.out.println(testObj.triArea(10,2));

}

}

**Result** : The above program is executed and obtained the output.

**Program : 13**

**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** :

Step 1: Start

Step 2: To create a package named arithmetic, create a folder of the same name in the directory. Here inside that we have another module named operation

Step 3: Inside arithmetic package, create modules to perform addition, subtraction, multiplication and division of 2 numbers.

Step 4: Outside the folder, write another program that accesses the above module and print the output.

Step 5:Stop

**Program Code** :

arithamatic\_package.java

interface interface\_graphics{

public float add(int a, int b);

public float divide(int a, int b);

public float multiple(int a, int b);

public float substract(int a, int b);

public float remainder(int a, int b);

}

public class arithamatic\_package implements interface\_graphics {

public float add(int a, int b){

return a+b;

}

public float divide(int a, int b){

return a/b;

}

public float multiple(int a, int b){

return a\*b;

}

public float substract(int a, int b){

return a-b;

}

public float remainder(int a, int b){

return a%b;

}

}

Main\_arithamatics.java

class main\_arithamatic {

public static void main(String []args){

arithamatic\_package testObj = new arithamatic\_package();

System.out.println("Addition of two numbers : "+testObj.add(1,2));

System.out.println("Subtraction of two numbers : "+testObj.substract(1,2));

System.out.println("Multiplication of two numbers : "+testObj.multiple(1,2));

System.out.println("Division of two numbers : "+testObj.divide(5,2));

System.out.println("Reminder of two numbers : "+testObj.remainder(11,2));

}

}

**Result** : The above program is executed and obtained the output.

**Program : 14**

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

**Algorithm** :

Step 1: Start

Step 2: Create a class named MyExceptions that inherits Exception class with a constructor inside which we call the Exception class constructor and pass error message.

Step 3: Inside the MyException(), Read the limit of array

Step 4: Inside the try block, read the array and check if any element is less than 0

Step 5: If true, throw MyExceptions with appropriate message.

Step 6: Calculate the average of the array and print it

Step 7: Inside the catch exception, Print “ – Not a positive number” Step

8:Stop.

**Program Code** :

import java.util.\*;

class MyException {

public static void main(String args[]) {

int totalNums;

int i;

int temp, count = 0;

int sum = 0;

Scanner sc = new Scanner(System.in);

System.out.println("Total numbers");

totalNums = Integer.parseInt(sc.nextLine());

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

try {

temp = Integer.parseInt(sc.nextLine());

if (temp > 0) {

sum += temp;

count += 1;

} else {

throw new MyExceptions(Integer.toString(temp));

}

} catch (MyExceptions ex) {

System.out.print(ex.getMessage());

System.out.println(" - Not a positive number");

}

}

System.out.print("Count : ");

System.out.println(count);

System.out.print("sum: ");

System.out.println(sum);

System.out.print("Average : ");

System.out.println(sum / count);

}

}

class MyExceptions extends Exception {

public MyExceptions(String value) {

super(value);

}

}

**Result** : The above program is executed and obtained the output.

**Program : 15**

**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** :

Step 1: Start

Step 2: Create a class named mul that inherits Thread class with member function as run()

Step 3: Inside run(), Print the multiplication table for 5

Step 4: Create a class named prime that inherits Thread class with memebr function run()

Step 5: Inside run(),Print the prime numbers upto the limit of user’s choice

Step 6: Inised the main(), create an object for the classes and call start() using each object

Step 7: Stop.

**Program Code** :

import java.util.\*;

class Multiplications extends Thread {

public void run() {

System.out.println("\n\nMULTIPLICATION TABLE OF 5\n");

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

System.out.println("5 \* " + i + " = " + (5 \* i));

}

}

}

class Prime extends Thread {

public void run() {

int count = 0, i = 1, j, n, no = 0;

Scanner sc = new Scanner(System.in);

System.out.println("\n\nEnter value for N :");

n = sc.nextInt();

System.out.println("Prime Numbers are :");

while (no != n) {

count = 0;

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

if (i % j == 0) {

count++;

break;

}

}

if (count == 0 && i != 1) {

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

no++;

}

i++;

}

}

}

class Multiplication {

public static void main(String args[]) {

Multiplications m = new Multiplications();

m.start();

Prime p = new Prime();

p.start();

}

}

**Result** : The above program is executed and obtained the output.

**Program : 16**

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

**Algorithm** :

Step 1: Start

Step 2: Create a class named even that implements Runnable interface with function run()

Step 3: Inside run(), we read the limit for printing even numbers and print it using for loop

Step 4: Create object e of even and create an object t1 of Thread with its parameterized constructor passing e as parameter

Step 5: Call start() using t1

Step 6: Do the same for class odd with Thread object t2 and call start() using t2

Step 7: Stop

**Program Code** :

package mca.myapp.CO4;

import java.util.\*;

class FibonacciNo implements Runnable {

int l;

FibonacciNo(int n) {

l = n;

}

public void run() {

int c;

int a = 0, b = 1;

System.out.print(a + " " + b);

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

c = a + b;

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

a = b;

b = c;

}

}

}

class even implements Runnable {

int l;

even(int n) {

l = n;

}

public void run() {

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

if (i % 2 == 0)

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

}

System.out.println("");

}

}

class fibonacci {

public static void main(String args[]) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter Limit :");

int l = sc.nextInt();

FibonacciNo f = new FibonacciNo(l);

Thread T1 = new Thread(f);

T1.start();

even e = new even(l);

Thread T2 = new Thread(e);

T2.start();

}

}

**Result** : The above program is executed and obtained the output.

**Program : 17**

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

**Algorithm** :

Step 1: Start

Step 2: Create a class named CustomException that inherits Exception class with a constructor that calls Exception class constructor and pass error meaasage.

Step 3: A username and password mention in main()

Step 4: Inside the main(), Read the username and password.

Step 5: Inside the try block, we throw username and password with message “Authenticate successfully” if the condition is true: username.equals(“user”) && password.equals(“pass”)

Step 6: Inside the catch block with parameter InvalidUserException , print “Invalid username / password provided!”

Step 7: Stop

**Program Code** :

public class CustomException {

public static class InvalidUserException extends Exception {

public InvalidUserException() {

super("Invalid username / password provided!");

}

}

public static void main(String[] args) {

String username = "john";

String password = "pass";

try {

if (username.equals("user") && password.equals("pass")) {

System.out.println("Authenticated successfully!");

} else {

throw new InvalidUserException();

}

} catch (InvalidUserException e) {

System.out.println(e);

}

}

}

**Result** : The above program is executed and obtained the output.

**Program : 18**

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

**Algorithm** :

Step 1: Start

Step 2: Create an object named ‘h\_set’ of type HashSet.

Step 3: Add values into h\_set using add() function.

Step 4: Create another object named ‘h\_set2’ of type HashSet

Step 5: Add values into h\_set2 using add() function.

Step 6: Create another object named ‘result\_set’ of type HashSet

Step 7: While traversing through the hashset using for loop, compare the two hashset objects h\_set and h\_set2 using contains() function and print the same.

Step 8: Stop.

**Program Code** :

import java.util.\*;

public class Hash {

public static void main(String[] args) {

// Create a empty hash set

HashSet<String> h\_set = new HashSet<String>();

// use add() method to add values in the hash set

h\_set.add("Red");

h\_set.add("Green");

h\_set.add("Black");

h\_set.add("White");

HashSet<String>h\_set2 = new HashSet<String>();

h\_set2.add("Red");

h\_set2.add("Pink");

h\_set2.add("Black");

h\_set2.add("Orange");

//comparison output in hash set

HashSet<String>result\_set = new HashSet<String>();

for (String element : h\_set){

System.out.println(h\_set2.contains(element) ? "Yes" : "No");

}

}

}

**Result** : The above program is executed and obtained the output.

**Program : 19**

**Aim** : To implement a simple calculator using AWT components.

**Algorithm** :

Step.1: Start the program.

Step.2: Define a class ‘calculator’ that extends Frame and implements ActionListener interface.

Step.3: Using TextField class object, construct the required no. of textfields wide enough to hold the values entered by the user.

Step.4: Using Label class object, construct and provide the appropriate labels.

Step.5: Using Button class object, construct labeled buttons that send the instances of ActionEvent.

Step.6: Call addActionListener() method to send events from the button to the new listener.

Step.7: Get the string values from textfields and then parse them as integers.

Step.8: Perform various methods to add, subtract, multiply and divide those integers.

Step.9: Stop the program.

**Program Code** :

import java.awt.\*;

import java.awt.event.\*;

class Calculator implements ActionListener {

Frame f = new Frame("CALCULATOR");

Label l1 = new Label("First Number");

Label l2 = new Label("Second Number");

Label l3 = new Label("Result");

TextField t1 = new TextField();

TextField t2 = new TextField();

TextField t3 = new TextField();

Button b1 = new Button("Add");

Button b2 = new Button("Sub");

Button b3 = new Button("Mul");

Button b4 = new Button("Div");

Button b5 = new Button("Cancel");

Calculator() {

l1.setBounds(50,100,100,20);

l2.setBounds(50,140,100,20);

l3.setBounds(50,180,100,20);

t1.setBounds(200,100,100,20);

t2.setBounds(200,140,100,20);

t3.setBounds(200,180,100,20);

b1.setBounds(50,250,50,20);

b2.setBounds(110,250,50,20);

b3.setBounds(170,250,50,20);

b4.setBounds(230,250,50,20);

b5.setBounds(290,250,50,20);

f.add(l1);

f.add(l2);

f.add(l3);

f.add(t1);

f.add(t2);

f.add(t3);

f.add(b1);

f.add(b2);

f.add(b3);

f.add(b4);

f.add(b5);

b1.addActionListener(this);

b2.addActionListener(this);

b3.addActionListener(this);

b4.addActionListener(this);

b5.addActionListener(this);

f.setLayout(null);

f.setVisible(true);

f.setSize(400,350);

}

public void actionPerformed(ActionEvent e) {

int n1 = Integer.parseInt(t1.getText());

int n2 = Integer.parseInt(t2.getText());

if(e.getSource()==b1) {

t3.setText(String.valueOf(n1 + n2));

}

if(e.getSource()==b2) {

t3.setText(String.valueOf(n1 - n2));

}

if(e.getSource()==b3) {

t3.setText(String.valueOf(n1 \* n2));

}

if(e.getSource()==b4) {

t3.setText(String.valueOf(n1 / n2));

}

if(e.getSource()==b5) {

System.exit(0);

}

}

public static void main(String...s) {

new Calculator();

}

}

**Result** : The above program is executed and obtained the output.

**Program : 20**

**Aim** : To write a program to find maximum of three numbers using AWT.

**Algorithm** :

Step.1: Start the program.

Step.2: Define a class ‘Main’ that extends Applet class and implements ActionListener interface.

Step.3: Using TextField class object, construct the required no. of textfields wide enough to hold the values entered by the user.

Step.4: Using Button class object, construct a labeled button that sends an instance of ActionEvent.

Step.5: Call addActionListener() method to send events from the button to the new listener.

Step.6: Get the string values from textfields and then parse them as integers.

Step.7: Compare each value using if-else statements to find the maximum value and set the result accordingly.

Step.8: Stop the program.

**Program Code** :

import java.awt.\*;

import java.awt.event.\*;

class MaxFind implements ActionListener {

MaxFind() {

Frame mainFrame = new Frame();

Label L1 = new Label("NO 1");

Label L2 = new Label("NO 2");

Label L3 = new Label("NO 3");

Label L4 = new Label("MAX NO");

Button B1 = new Button("Submit");

Button B2 = new Button("Cancel");

TextField T1 = new TextField();

TextField T2 = new TextField();

TextField T3 = new TextField();

TextField T4 = new TextField();

// set posiotion and size

L1.setBounds(50, 50, 80, 50);

L2.setBounds(150, 50, 80, 50);

L3.setBounds(250, 50, 80, 50);

L4.setBounds(350, 50, 80, 50);

T1.setBounds(50, 100, 80, 50);

T2.setBounds(150, 100, 80, 50);

T3.setBounds(250, 100, 80, 50);

T4.setBounds(350, 100, 80, 50);

B1.setBounds(150, 200, 80, 30);

B2.setBounds(250, 200, 80, 30);

// Add componants to frame

mainFrame.add(B1);

mainFrame.add(B2);

mainFrame.add(L1);

mainFrame.add(L2);

mainFrame.add(L3);

mainFrame.add(L4);

mainFrame.add(T1);

mainFrame.add(T2);

mainFrame.add(T3);

mainFrame.add(T4);

// Action events

B1.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

int a = Integer.parseInt(T1.getText());

int b = Integer.parseInt(T2.getText());

int c = Integer.parseInt(T3.getText());

int large=0;

if (a >= b && a >= c) {

large = a;

System.out.println("Maximum of three numbers is " + (large=a));

} else if (b >= a && b >= c) {

large = b;

System.out.println("Maximum of three numbers is " + (large=b));

} else if (c >= a && c >= b) {

large = c;

System.out.println("Maximum of three numbers is " + (large=c));

}

T4.setText(Integer.toString(large));

}

});

mainFrame.setSize(600, 300);

mainFrame.setTitle("Max of 3 Numbers");

mainFrame.setLayout(null);

mainFrame.setVisible(true);

B2.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

System.exit(0);

}

});

}

public static void main(String args[]) {

MaxFind maxObj = new MaxFind();

}

public void actionPerformed(ActionEvent e) {

}

}

**Result** : The above program is executed and obtained the output.

**Program : 21**

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

**Algorithm** :

Step.1: Start the program.

Step.2: Create a class named ‘Main’.

Step.3: Create an object for the class File to initialize its constructor with the given file.

Step.4: Get user inputs via the console, for the integers to be inserted into the file.

Step.5: Using an object for the Main class, write those integers into the file.

Step.6: Using objects for the FileWriter class, create two separate files to store even and odd integers respectively and copy the integers accordingly to separate files just created.

int count = 0;

for (String i : values) {

valuesInt[count++] = Integer.parseInt(i);

//System.out.println(Integer.parseInt(i));

if (Integer.parseInt(i) % 2 == 0) {

evenFile.write(i+" ");

} else {

oddFile.write(i+" ");

}

}

Step.7: Stop the program.

**Program Code** :

import java.io.FileWriter;

import java.io.IOException;

import java.io.File;

//import java.io.FileNotFoundException;

import java.util.Scanner;

class Main {

static String data = "";

static File dataFile = new File("DATA.txt");

public static void main(String[] arg) {

try {

FileWriter oddFile = new FileWriter("odd.txt", true);

FileWriter evenFile = new FileWriter("even.txt", true);

System.out.println("Files created for odd numbers and even numbers");

Scanner dataRead = new Scanner(dataFile);

while (dataRead.hasNextLine()) {

data += dataRead.nextLine();

}

dataRead.close();

String values[] = data.split(" ");

int valuesInt[] = new int[values.length + 1];

int count = 0;

for (String i : values) {

valuesInt[count++] = Integer.parseInt(i);

//System.out.println(Integer.parseInt(i));

if (Integer.parseInt(i) % 2 == 0) {

evenFile.write(i+" ");

} else {

oddFile.write(i+" ");

}

}

oddFile.close();

evenFile.close();

} catch (IOException ex) {

System.out.println("An error occured !");

ex.printStackTrace();

} catch (Exception ex) {

System.out.print("An error occured : ");

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

}

}

}

**Result** : The above program is executed and obtained the output.

**Program : 22**

**Aim** : To write a program to copy one file to another

**Algorithm** :

Step.1: Start the program.

Step.2: Create a class named ‘Main’.

Step.3: Create and use an object for the dataFile class to read the stream of characters from the specified file.

Step.4: Create and use an object for the FileWriter class to write the stream of characters read by the dataWriter, to the file.

Step.5: Stop the program

**Program Code** :

import java.io.FileWriter;

import java.io.IOException;

import java.io.File;

import java.io.FileNotFoundException;

import java.util.Scanner;

class Main {

static String data="";

public static void main(String[] arg) {

// Read

try {

File dataFile = new File("DATA.txt");

Scanner dataRead = new Scanner(dataFile);

while (dataRead.hasNextLine()) {

data += dataRead.nextLine();

data +="\n";

}

dataRead.close();

} catch (FileNotFoundException ex) {

System.out.println("An error occured !");

ex.printStackTrace();

}

// Write

try {

FileWriter dataWriter = new FileWriter("DATA2.txt");

System.out.println("Data read from file DATA.txt and copied in DATA2.txt");

dataWriter.write(data);

dataWriter.close();

} catch (IOException ex) {

System.out.println("An error occured !");

ex.printStackTrace();

}

}

}

**Result** : The above program is executed and obtained the output.