**DR B.R. AMBEDKAR NATIONAL INSTITUTE OF**

**TECHNOLOGY JALANDHAR,**



**Java Laboratory**

**CSX-351**

**Session: July-Dec 2019**

**SUBMITTED TO- SUBMITTED BY-**

Dr. Rajneesh Rani NAME: RITIK KUMAR

ROLL NO: 17103073

GROUP: G-3

BRANCH: CSE

**INDEX**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S No** | **Topic** | **Page No** | **Date** | **Sign** |
| **1** | LAB 1 | **3-9** |  |  |
| **2** | LAB 2 | **10-16** |  |  |
| **3** | LAB 3 | **17-24** |  |  |
| **4** | LAB 4 | **25-30** |  |  |
| **5** | LAB 5 | **31-41** |  |  |
| **6** | LAB 6 | **42-49** |  |  |
| **7** | LAB 7 | **50-63** |  |  |
| **8** | LAB 8 | **64-70** |  |  |
| **9** | LAB 9 | **71-78** |  |  |
| **10** | LAB 10 | **79-86** |  |  |
| **11** | LAB 11 | **86-96** |  |  |
| **12** |  |  |  |  |

**LAB - 1**

**1. Write an application that ask the user to enter 2 integers, obtain them from user and displays a large number followed by the work “is larger”. If the number are equal, print the message “ These numbers are equal”.**

import java.util.\*;

class Lab1 {

public static void main(String[] args) {

int a, b;

Scanner in = new Scanner(System.in);

a = in.nextInt();

b = in.nextInt();

if(a > b) {

System.out.println(a + " is larger.");

}

else if(b > a) {

System.out.println(b + " is larger.");

}

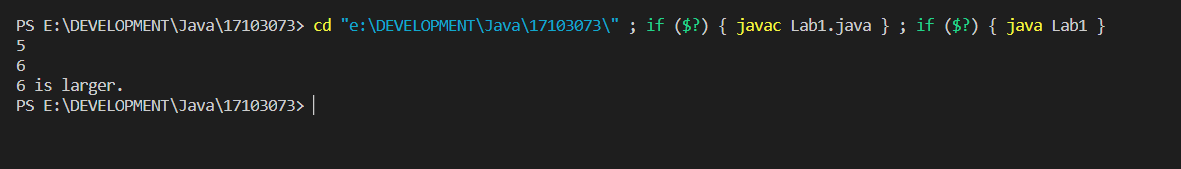
else {

System.out.println("The numbers are equal.");

}

}

}



**2. Write an application in which user passes Command Line arguments and the output shows the count of the arguments passed on command line and also print them all.**

import java.util.\*;

public class Lab1 {

public static void main(String[] args) {

System.out.println("\nThe number of arguments are : " + args.length);

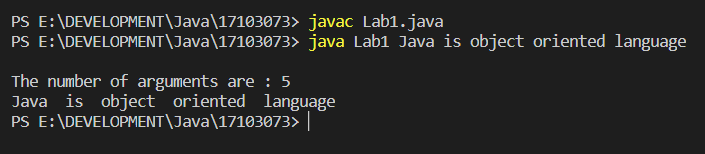
for (int i = 0; i < args.length; i++) {

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

}

}

}



**3. Write an application to implement Pascal Triangle.**

import java.util.\*;

public class Lab1 {

static public int factorial(int n) {

int fact = 1;

for(int i=2; i<=n; i++) {

fact \*= i;

}

return fact;

}

static public int nCr(int n, int r) {

return factorial(n)/(factorial(n-r) \* factorial(r));

}

public static void main(String[] args) {

int rows;

Scanner in = new Scanner(System.in);

rows = in.nextInt();

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

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

System.out.print(nCr(n,r) + " ");

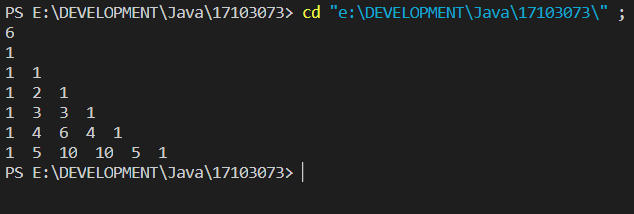
}

System.out.println();

}

}

}



**4. Write a menu driven application using switch-case statements in which case 1 : print a box, case 2 : an oval, case 3 : an arrow and case 4 : a diamond using loop and control statements.**

import java.util.\*;

public class Lab1 {

public static void printSquare(int number) {

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

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

System.out.print("\*");

}

System.out.println();

for (int i = 0; i < number - 2; i++) {

System.out.print("\*");

for (int j = 0; j < number - 2; j++) {

System.out.print(" ");

}

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

}

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

System.out.print("\*");

}

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

}

public static void printCircle(int radius) {

double dist;

for (int i = 0; i <= 2 \* radius; i++) {

for (int j = 0; j <= 2 \* radius; j++) {

dist = Math.sqrt((i - radius) \* (i - radius) + (j - radius) \* (j - radius));

if (dist > radius - 0.5 && dist < radius + 0.5)

System.out.print("\*");

else

System.out.print(" ");

}

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

}

}

public static void printUpwardArrow(int number) {

int count = number - 1;

for (int k = 1; k <= number; k++) {

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

System.out.print(" ");

count--;

for (int i = 1; i <= 2 \* k - 1; i++)

System.out.print("\*");

System.out.println();

}

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

for (int j = 0; j < number - 1; ++j) {

System.out.print(" ");

}

System.out.print("\*");

System.out.println();

}

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

}

public static void printDiamond(int n) {

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

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

System.out.print(" ");

}

int k = 0;

while (k != (2 \* i - 1)) {

if (k == 0 || k == 2 \* i - 2)

System.out.print("\*");

else

System.out.print(" ");

k++;

}

System.out.println();

}

n--;

for (int i = n; i >= 1; i--) {

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

System.out.print(" ");

}

int k = 0;

while (k != (2 \* i - 1)) {

if (k == 0 || k == 2 \* i - 2)

System.out.print("\*");

else

System.out.print(" ");

k++;

}

System.out.println();

}

}

public static void main(String[] args) {

Scanner in = new Scanner(System.in);

byte option;

System.out.println("\nEnter the option pattern you wonna build: ");

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

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

System.out.println("3. Arrow");

System.out.println("4. Diamond");

option = in.nextByte();

switch (option) {

case 1:

printSquare(3);

break;

case 2:

printCircle(3);

break;

case 3:

printUpwardArrow(3);

break;

case 4:

printDiamond(3);

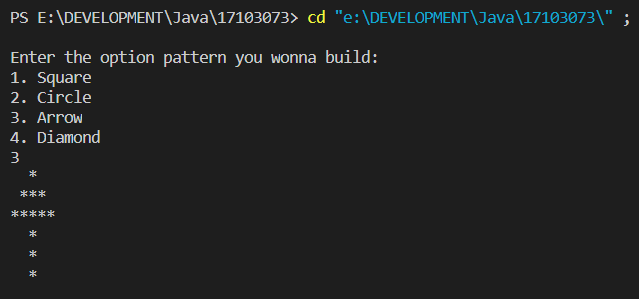
break;

}

in.close();

}

}



**LAB – 2**

**1. Write an application to that asks the user to enter three integer values(2 < n < 100) , tand then displays that these values are Pythagorean triplets or not. The application should try all the combinations of the values.**

import java.util.\*;

class Lab2 {

public static void main(String[] args) {

Scanner in = new Scanner(System.in);

System.out.println("Enter the three numbers(Enter the hypotenuese first) : ");

int a = in.nextInt();

int b = in.nextInt();

int c = in.nextInt();

if(a \* a == b \* b + c \* c) {

System.out.println("\nThe three numbers are PythagoreanTriplets");

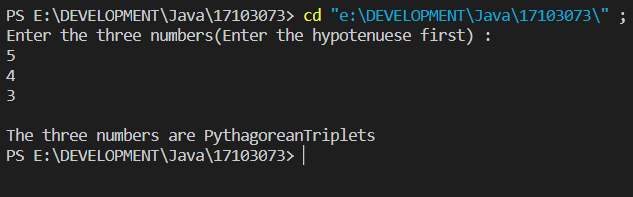
} else {

System.out.println("\nThe three numbers are not PythagoreanTriplets\n");

}

}

}



**2. Write an application that inputs from the user the radius of the circle as a floating point and prints the diameter, circumference and area. Use the following formulas :**

**Take PI = 3.14159, should be declared as final .**

import java.util.\*;

class circle {

public float radius;

circle(int radius) {

this.radius = radius;

}

final float PI = 3.14159f;

public float diameter() {

return 2 \* this.radius;

}

public float area() {

return (PI \* this.radius \* this.radius);

}

public float perimeter() {

return (2 \* PI \* this.radius);

}

}

public class Lab2 {

public static void main(String[] args) {

int radius;

System.out.println("\nEnter the radius of the circle");

Scanner in = new Scanner(System.in);

radius = in.nextInt();

circle C = new circle(radius);

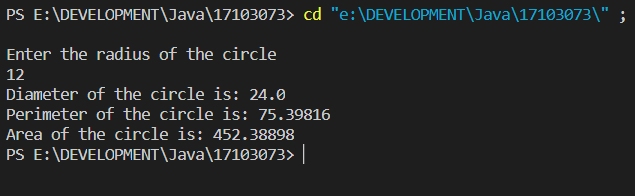
System.out.println("Diameter of the circle is: " + C.diameter());

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

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

}

}



**3. Write an application which contains a class named “Account” that maintains the balance of a bank account. The Account class has method debit() that withdraws money from the account. Ensure that the debit amount does not exceed the account’s balance. If it does, the balance should remain unchanged with a message indicating “Debit amount exceeded account balance”.**

import java.util.\*;

class Account {

private int balance;

Account(int balance) {

this.balance = balance;

}

int getBalance() {

return balance;

}

void creditBalance(int money) {

balance += money;

}

void withdrawMoney(int money) {

if (money > getBalance()) {

System.out.println("Debit account exceeded account balance");

return;

}

debit(money);

System.out.println("Money withdrawn successful!");

System.out.println("Money left : " + balance + "\n");

}

void debit(int moneyWithdrawn) {

balance -= moneyWithdrawn;

}

}

class Lab2 {

public static void main(String[] args) {

Scanner in = new Scanner(System.in);

System.out.print("Enter the initial amount of balance: ");

int balance = in.nextInt();

Account A = new Account(balance);

while (true) {

System.out.print("\nWonna withdraw some money? Press Y : ");

char response = in.next().charAt(0);

if (response != 'Y' && response != 'y') {

System.out.print("\nThank You for using our Banking service.\n");

break;

}

System.out.print(" Enter the amount of money in Rupees: ");

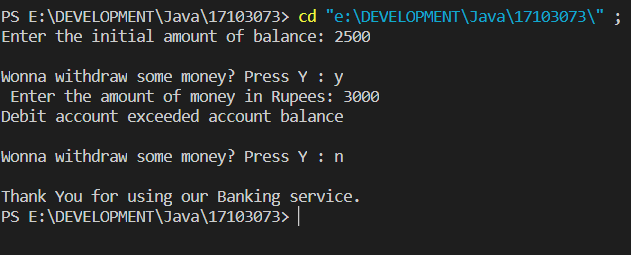
int money = in.nextInt();

A.withdrawMoney(money);

}

}

}



**4. Write an application to determine the gross pay for 3 employees. The company pays strainght for first 40 hours and half after that for overtime. Salary per hour and number of hours of each employee should be input by user.**

import java.util.\*;

class Employee {

private int numberOfHours;

private int salaryPerHour;

Employee(int n, int s) {

numberOfHours = n;

salaryPerHour = s;

}

int extraHours() {

if (numberOfHours <= 40) {

return 0;

}

return numberOfHours - 40;

}

int getSalary() {

return ((numberOfHours \* salaryPerHour) + (extraHours() \* (salaryPerHour / 2)));

}

}

class Lab2 {

public static void main(String[] args) {

Employee[] E = new Employee[3];

for (int i = 1; i <= 3; i++) {

int salaryPerHour;

Scanner in = new Scanner(System.in);

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

salaryPerHour = in.nextInt();

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

int numberOfHours = in.nextInt();

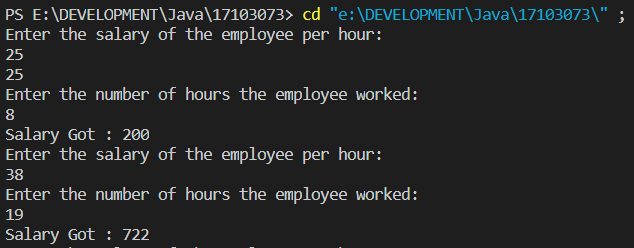
Employee E1 = new Employee(numberOfHours, salaryPerHour);

System.out.println("Salary Got : " + E1.getSalary());

}

}

}



**LAB - 3**

**1. Write a program to find an element taken as the input by the user in a 2D array.**

import java.util.\*;

public class Lab3 {

public static void main(String[] args) {

Scanner in = new Scanner(System.in);

System.out.println("\nEnter the row and column of the matrix : ");

int row = in.nextInt();

int column = in.nextInt();

int arr[][] = new int[row][column];

System.out.println("\nEnter the elements : ");

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

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

arr[i][j] = in.nextInt();

}

}

System.out.println("\nEnter the element to be found : ");

int toFindElement = in.nextInt();

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

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

if(arr[i][j] == toFindElement) {

System.out.println("Element was found");

return;

}

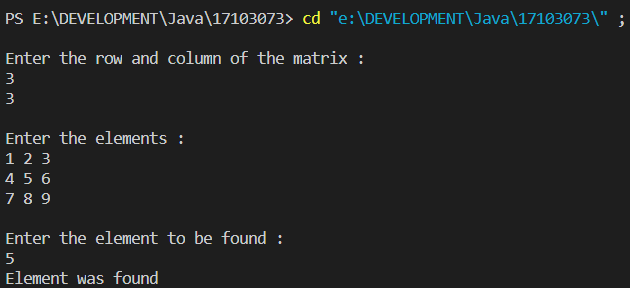
}

}

System.out.println("Element was not found");

}

}



**2. Write a program to implement all the functions of the String class.**

import java.util.\*;

public class Lab3 {

public static void main(String[] args) {

Scanner in = new Scanner(System.in);

System.out.println("\nEnter the string to work with: ");

String str = in.nextLine();

int option = 1;

do {

System.out.println("\nWhat do you want to do with the string? ");

System.out.println("1. Convert to Lower case.");

System.out.println("2. Convert to Upper case.");

System.out.println("3. Replace a character.");

System.out.println("4. Check the equality with other string.");

System.out.println("5. Find the length of the string.");

System.out.println("6. Find the character at a given position.");

System.out.println("7. Concatenate another string.");

System.out.println("8. Check if substring of the given string.");

System.out.println("0. Exit\n");

option = in.nextInt();

switch(option) {

case 1:

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

break;

case 2:

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

break;

case 3:

System.out.println("Tell me the index where to replace the character: ");

int x = in.nextInt();

System.out.println("Enter the character to put in its place: ");

char ch = in.next().charAt(0);

String replace = str.substring(0, x) + ch + str.substring(x+1);

System.out.println(replace);

break;

case 4:

System.out.println("Enter another string: ");

String string = in.next();

if(str.equals(string))

System.out.println("The two strings are equal.");

else

System.out.println("The two strings aren't equal.");

break;

case 5:

System.out.println("The length of string is : " + str.length());

break;

case 6:

System.out.println("Enter the index position of the character(indexing starts from 0): ");

int index = in.nextInt();

System.out.println(str.charAt(index));

break;

case 7:

System.out.println("Enter the other string to concatenate: ");

string = in.next();

String string3 = str.concat(string);

System.out.println("The final string is: "+ string3);

break;

case 8:

System.out.println("Enter the string to check as a substring: ");

String substring = in.next();

System.out.println("The given string is " );

if(str.contains(substring)) {

System.out.println("contains the substring.");

}

else {

System.out.println("doesn't contain the substring. ");

}

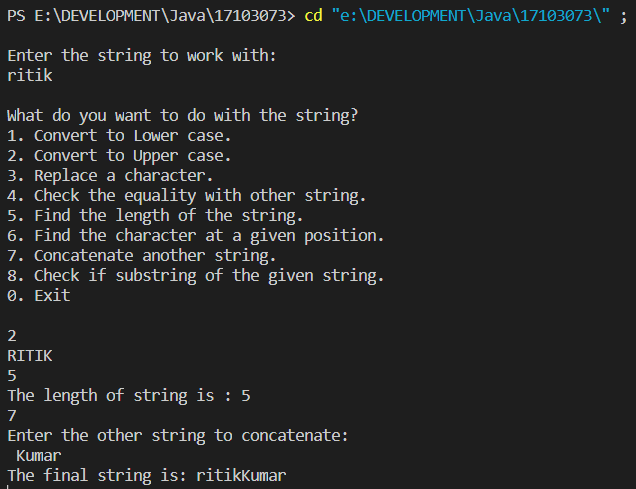
break;

}

}while(option != 0);

}

}



**3. A string is a double string if it has even length and half of the string is equal to the next half of the string. Write a program to check if the given string is a double string.**

import java.util.\*;

public class Lab3 {

    public static void main(String[] args) {

        Scanner in = new Scanner(System.in);

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

        String str = in.next();

        int n = str.length();

        if(n % 2 != 0) {

            return;

        }

        boolean bool = true;

        for (int i = 0; i < n / 2; i++) {

            if (str.charAt(i) != str.charAt(n / 2 + i)) {

                System.out.println("No, it is not a double string.");

                bool = false;

                break;

            }

        }

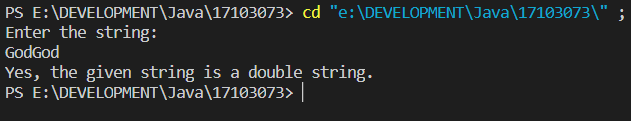
if(bool == true) {

            System.out.println("Yes, the given string is a double string.");

}

     }

}



**4. Write a program in which you have an array of integers. Implement copy paste which allows you to copy any continuous subsequences of array and paste it int any position of the array.**

import java.util.\*;

class Lab3{

public static void main(String[] args){

Scanner sc=new Scanner(System.in);

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

int size=sc.nextInt();

int[] arr= new int[size];

int i,j,length;

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

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

arr[i]=sc.nextInt();

}

int sCopy,eCopy,sPaste;

int choice=1;

while(choice==1){

System.out.println("Enter starting and ending index for copying");

sCopy=sc.nextInt();

eCopy=sc.nextInt();

if(sCopy<0 || eCopy>=size){

System.out.println("Error: Indices out of bound ");

} else {

length=eCopy-sCopy+1;

int[] copiedArray= new int[length];

for(i=sCopy,j=0;i<=eCopy;i++,j++){

copiedArray[j]=arr[i];

}

System.out.println("Enter starting index for pasting");

sPaste=sc.nextInt();

sPaste++;

if(sPaste+length<size){

for(i=sPaste-1,j=0;i<sPaste+length-1;i++,j++){

arr[i]=copiedArray[j];

}

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

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

}

}

}

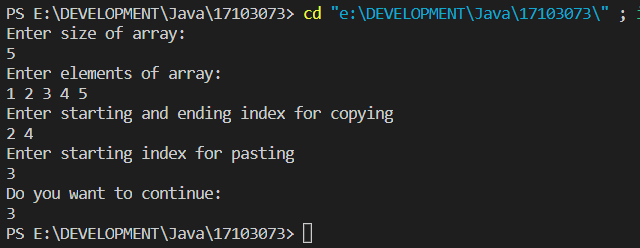
System.out.println("Do you want to continue: ");

choice = sc.nextInt();

}

}

}



**LAB – 4**

**1. Create a package package1 which contains three classes, one is super class and others are subclasses. Make some of the components private, some of them protected and some public, show access control using the above description. Then, create another package p2 in which the class can access public components of package package1. Now import the classes of package package1 and package2 for the class of the main() method.**

package package1;

public class problem1 {

public int variable1 = 1234;

protected String variable2 = " Java is Beautiful";

private float variable3 = 2.345f;

static public int variable4 = 4321;

void CallMe() {

System.out.println("Calling a function of the problem1 class\n");

System.out.println("We have called you. And you answered, so nice of you! I think you are default or public(may be protected if I am your child:)");

}

}

package package1;

class problem1a extends problem1 {

private int variable = 1234;

}

package package1;

class problem1b extends problem1 {

public int stuff = 1000000;

private float a = 0.00023495320f;

}

package p2;

import package1.\*;

class anotherClass {

public String variable = "This is my public Variable";

void Public() {

System.out.println("\nThe called function is my public.\n\n");

}

void CallingProblem1() {

System.out.println("\nThis is the public member of the problem1 class of package1 : " + package1.problem1.variable4);

}

}

import package1.\*;

class checkingPackage {

public static void main(String[] args) {

problem1 p = new problem1();

p.variable1 = 2;

System.out.println(p.variable1);

}

}

package package1;

import java.util.\*;

class mainClass {

public static void main(String[] args) {

System.out.println("\nAccessing the problem1b(which extends the problem1) variables and functions.");

problem1b p1b = new problem1b();

System.out.println("\nWow! I can access my public members here: " + p1b.stuff);

System.out.println("\nAlso, I can use my parents function : ");

p1b.CallMe();

}

}

**2. Create a class student which takes the roll number of the students, another class Test extends Student and add up marks of two tests. Now an interface Sports declares the sports marks and has method putMarks. Implement multiple inheritance in Class Result using above classes and interface.**

import java.util.\*;

class Student {

int rollNumber;

void SetRollNumber(int rollNumber) {

this.rollNumber = rollNumber;

}

int GetRollNumber() {

return this.rollNumber;

}

}

class Test extends Student {

int marks1, marks2;

public void setMarks(int marks1, int marks2) {

this.marks1 = marks1;

this.marks2 = marks2;

}

int AddMarksUp() {

return this.marks1 + this.marks2;

}

}

interface Sports {

public void setSportsMarks(int sM);

public int putMarks();

}

class Result extends Test implements Sports {

int sportsMarks;

public void setSportsMarks(int sM) {

sportsMarks = sM;

}

public int putMarks() {

int totalMarks = sportsMarks + marks1 + marks2;

return totalMarks;

}

}

public class Lab4 {

public static void main(String[] args) {

Result R = new Result();

Test T = new Test();

Scanner input = new Scanner(System.in);

System.out.println("Enter the marks in the two tests: ");

int marks1 = input.nextInt();

int marks2 = input.nextInt();

T.setMarks(marks1, marks2);

System.out.println("Enter the marks in the sports test: ");

int sportsMarks = input.nextInt();

R.setSportsMarks(sportsMarks);

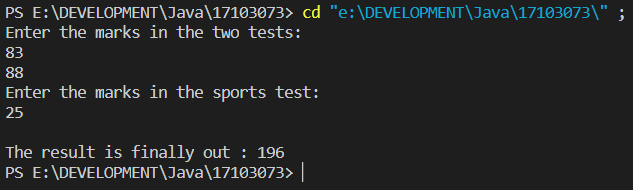
R.marks1 = T.marks1;

R.marks2 = T.marks2;

System.out.println("\nThe result is finally out : " + R.putMarks());

}

}



**3. Write an application which takes a 1D array of the numbers 0<=n<=9 only as input and displays the number of times n repeated most consecutively. For example :**

import java.util.\*;

public class Lab4 {

public static void main(String[] args){

Scanner sc= new Scanner(System.in);

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

int n=sc.nextInt();

int arr[]= new int[n];

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

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

arr[i]=sc.nextInt();

}

int count=1;

int max=0;

int number=arr[0];

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

if(arr[i]==arr[i+1]){

count++;

if(count>max){

max=count;

number=arr[i];

}

} else {

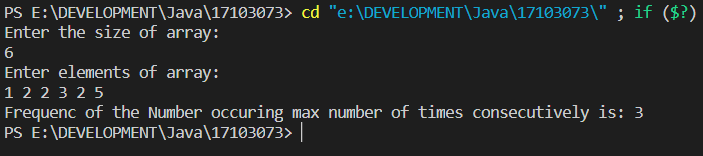
count = 1;

}

} System.out.println("Frequenc of the Number occuring max times is: "+ (max+1));

}

}



**4. Write an application to store all the addition combination of the two dices in 2D array and display as follows :**

import java.util.\*;

public class Lab4 {

public static void main(String[] args) {

int arr[][] = new int[7][7];

for(int i=1; i<=6; i++) {

for(int j=1; j<=6; j++) {

arr[i][j] = i+j;

}

}

int k=1;

for(int i=1; i<=6; i++) {

arr[0][i] = k;

arr[i][0] = k;

k++;

}

for(int i=0; i<7; i++) {

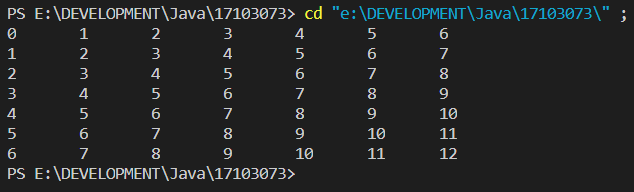
for(int j=0; j<7; j++) {

System.out.print(arr[i][j] + "\t");

}

System.out.println();

} } }



**LAB – 5**

**1. Write an application to show access protection in user defined packages. Create a package p1 which has three classes : Protection, Derived and SamePackage. First class contains variables of all the access types. Derived is subclass of Protection but SamePackage is not. Other package p2 consisting of two class : one deriving p1.Protection and other not. Import the package considering all the cases of access protection.**

package p1;

class Protection {

public int publicVariable = 2323;

private float floatVariable = 23.223f;

protected String stringVariable = "Protected";

}

class Derived extends Protection {

public int stuff = 23421;

public void print() {

System.out.println("I am a function of Derived class");

}

public void printProtected() {

System.out.println("I can access the protected variable of Protection class. Its value is: " + stringVariable);

}

}

class SamePackage {

int variable = 23;

protected String is = "Hello";

}

package p2;

import p1.\*;

public class first extends Protection {

int anyNumber = 23422;

public String anyName = "Henry Ford";

}

class second {

double doubleStuff = 2.3324;

Integer integer = new Integer(23);

public void print() {

System.out.println("I am a function of second class.");

System.out.println("I have only two variable : " + doubleStuff + "and " + integer);

}

}

import p1.Protection;

import p1.Derived;

import p1.SamePackage;

import p2.first;

import p2.second;

public class Problem1 {

public static void main(String[] args) {

Protection p = new Protection();

System.out.println("Public Variable's value : "+ p.publicVariable);

System.out.println("\n\nThe variables and functions of Protection class are: ");

// System.out.println("Protected Varible: " + stringVariable);

System.out.println("Can't access Private and Protected variables\n\n");

Derived d = new Derived();

System.out.println("The variables and functions of Derived class are: ");

System.out.println("Public variable's value : " + d.stuff);

d.print();

d.printProtected();

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

SamePackage s = new SamePackage();

System.out.println("The variables and functions of SamePackage class are: ");

System.out.println("The default variable of this class is not visible.");

System.out.println("The protected variable of this class is not visible.\n\n");

first f = new first();

System.out.println("The variables and functions of first class are: ");

System.out.println("The default variable of the class is not accessible.");

System.out.println("Public variable's value is : " + f.anyName + "\n\n");

second sec = new second();

System.out.println("The default variables of the class are not accessible.");

System.out.println("But there is a public function. Lets call that : ");

sec.print();

}}

**2. Multiple Inheritance in java is when a class implements more than one interface. Write an application showing Multiple Inheritance and Dynamic Polymorphism (Method Overriding) using Interface.**

import java.util.\*;

class Game {

    public void type() {

        System.out.println("We are talking about Indoor and Outdoor games.");

    }

    public void numberOfPlayers() {

        System.out.println("Number of player is a relative term, depends on the game.");

    }

}

class Football extends Game {

    public void type() {

        System.out.println("Football is an outdoor game.");

    }

    public void numberOfPlayers() {

        System.out.println("In Football, we have 11 players in each team.");

    }

}

class Kabaddi extends Game {

    public void type() {

        System.out.println("Kabaddi is an outdoor game.");

    }

    public void numberOfPlayers() {

        System.out.println("In Kabaddi, we have 12 players in each team.");

    }

}

public class Lab4 {

    public static void main(String[] args) {

        Game g = new Game();

        System.out.println("Game class functions : ");

        g.type();

        g.numberOfPlayers();

        System.out.println("\n\n");

        System.out.println("Football class functions : ");

        g = new Football();

        g.type();

        g.numberOfPlayers();

        System.out.println("\n\n");

        System.out.println("Kabaddi class functions : ");

        g = new Kabaddi();

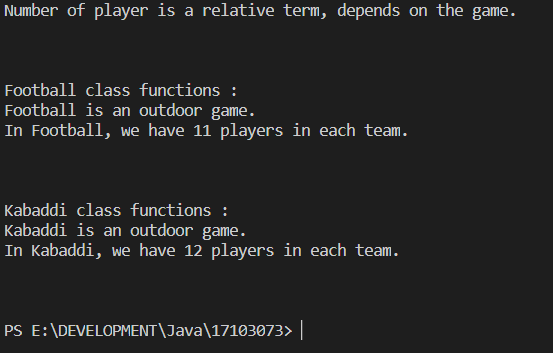
        g.type();

        g.numberOfPlayers();

        System.out.println("\n\n");

    }

}



**3. Define a package named area which contains classes : Square, Rectangle, Circle, Elipse and Triangle. Write an application in which the main class, FindArea outside the package area imports the package area and calculate the area of different shapes. Output of your application should be menu driven.**

import java.util.Scanner;

class Square {

private int side;

public Square() {

System.out.println("You have chosen to find the area of Square. Please enter the length of side of square : ");

Scanner input = new Scanner(System.in);

side = input.nextInt();

}

public double getArea() {

return side \* side;

}

}

class Rectangle {

private int length, breadth;

public Rectangle() {

System.out.println("You have chosen to find the area of Rectangle. Please enter the length and breadth of the rectangle : ");

Scanner input = new Scanner(System.in);

length = input.nextInt();

breadth = input.nextInt();

}

public double getArea() {

return length \* breadth;

}

}

class Triangle {

private int height, base;

public Triangle() {

System.out.println("You have chosen to find the area of Triangle. Please enter the length of height and base of the Triangle : ");

Scanner input = new Scanner(System.in);

height = input.nextInt();

base = input.nextInt();

}

public double getArea() {

return 0.5 \* base \* height;

}

}

class Ellipse {

private int majorAxesLength, minorAxesLength;

public Ellipse() {

System.out.println("You have chosen to find the area of Elipse. Please enter the length of major and minor axes of ellipse : ");

Scanner input = new Scanner(System.in);

majorAxesLength = input.nextInt();

minorAxesLength = input.nextInt();

}

public double getArea() {

return 3.14 \* majorAxesLength \* minorAxesLength;

}

}

class Circle {

private int radius;

public Circle() {

System.out.println("You have chosen to find the area of Circle. Please enter the length of radius of circle : ");

Scanner input = new Scanner(System.in);

radius = input.nextInt();

}

public double getArea() {

return 3.14 \* radius \* radius;

}

}

public class Lab6 {

public static void main(String[] args) {

System.out.println("\nThe official name of the application is : FINDAREA");

Scanner input = new Scanner(System.in);

byte option;

do {

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

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

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

System.out.println("3. Circle");

System.out.println("4. Ellipse");

System.out.println("5. Triangle");

System.out.println("0. Exit");

option = input.nextByte();

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

switch(option) {

case 1:

Square s = new Square();

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

System.out.println("The area is : " + s.getArea());

break;

case 2:

Rectangle r = new Rectangle();

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

System.out.println("The area is : " + r.getArea());

break;

case 3:

Circle c = new Circle();

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

System.out.println("The area is : " + c.getArea());

break;

case 4:

Ellipse e = new Ellipse();

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

System.out.println("The area is : " + e.getArea());

break;

case 5:

Triangle t = new Triangle();

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

System.out.println("The area is : " + t.getArea());

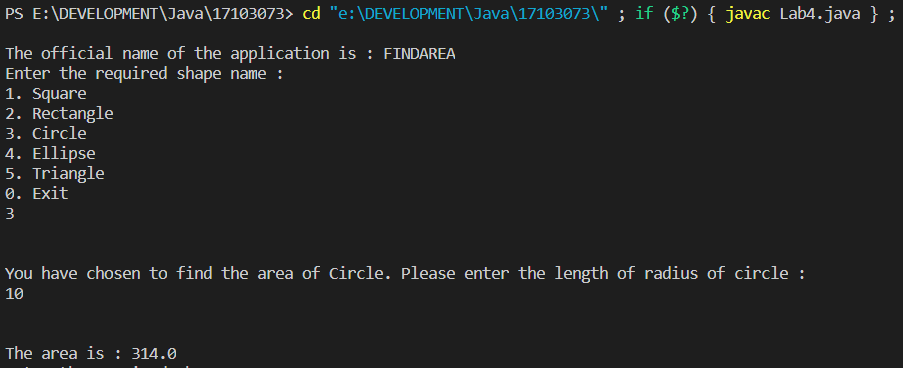
break;

}

}while(option != 0);

}

}



**4. Define an interface Volume having a final variable P1 and a method to calculate volumes of different shapes. Implement the interface Volume in class Cube, Cylinder, Cone and Sphere to calculate the volume of these different shapes. Calculate the volume using the main class FindVolume. Output of your application should be menu driven.**

interface Volume {

final double pi = 3.14;

public double volume();

}

class Sphere implements Volume{

public int radius;

public Sphere(int radius){

this.radius = radius;

}

public double volume(){

double v = (4 \* pi \* radius \* radius \* radius ) / 3;

return v;

}

}

class Cone implements Volume{

public int radius,height;

public Cone(int radius,int height){

this.radius=radius;

this.height=height;

}

public double volume(){

return (pi\*radius\*radius\*height)/3;

}

}

class Cube implements Volume{

public int side;

public Cube(int side){

this.side=side;

}

public double volume(){

return side\*side\*side;

}

}

class Cylinder implements Volume {

public int radius,h;

public Cylinder(int radius, int h){

this.radius=radius;

this.h=h;

}

public double volume(){

return pi\*radius\*h;

}

}

class Problem4 {

public static void main(String[] args) {

int option;

Scanner input = new Scanner(System.in);

do {

System.out.println("Choose the solid you want the volume for: ");

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

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

System.out.println("3. Cube");

System.out.println("4. Cone");

System.out.println("0. Exit");

option = input.nextInt();

switch(option) {

case 1:

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

int r = input.nextInt();

Sphere s = new Sphere(r);

System.out.println("Volume of the sphere is : " + s.volume()+ "\n");

break;

case 2:

System.out.println("Enter radius and height of the cylinder : ");

int radius = input.nextInt();

int height = input.nextInt();

Cylinder c = new Cylinder(radius, height);

System.out.println("Volume of the cylinder : " + c.volume() + "\n");

break;

case 3:

System.out.println("Enter the side of the cube : ");

int side = input.nextInt();

Cube cB = new Cube(side);

System.out.println("Volume of cone is : " + cB.volume() + "\n");

break;

case 4:

System.out.println("Enter the side of the Cone : ");

radius = input.nextInt();

height = input.nextInt();

Cone cN = new Cone(radius, height);

System.out.println("Volume of cone is : " + cN.volume() + "\n");

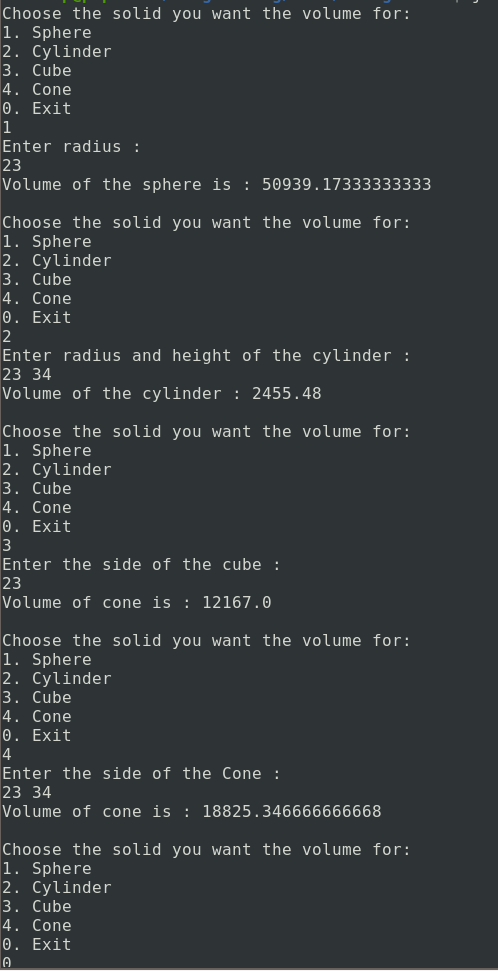
break;

}

}while(option != 0);

}

}



**LAB - 6**

**1. Use inheritance to create an exception superclass (called ExceptionA) and exception subclasses ExceptionB and ExceptionC, where ExceptionB inherits from ExceptionA and ExceptionC inherits from ExceptionB. Write a program to demonstrate that the catch block for type ExceptionA catches exceptions of type ExceptionB and ExceptionC.**

class ExceptionA extends Exception {

protected int detailA;

public ExceptionA(int a) {

detailA = a;

}

public String toString() {

return "The Number is greater than " + detailA + "\n";

}

}

class ExceptionB extends ExceptionA {

protected int detailB;

public ExceptionB(int a) {

super(a);

detailB = a;

}

public String toString() {

return "The number is greater than " + detailB + "\n";

}

}

class ExceptionC extends ExceptionB {

protected int detailC;

public ExceptionC(int a) {

super(a);

detailC = a;

}

public String toString() {

return "the number is greater than " + detailC + "\n";

}

}

public class Problem1 {

public static void main(String[] args) {

int a;

Scanner in = new Scanner(System.in);

a = in.nextInt();

try {

if(a == 1) {

throw new ExceptionA(a);

}

if(a == 3) {

throw new ExceptionB(a);

}

if(a == 5) {

throw new ExceptionC(a);

}

} catch(ExceptionC eC) {

System.out.println("Caught " + eC);

}

catch(ExceptionB eB) {

System.out.println("Caught " + eB);

}

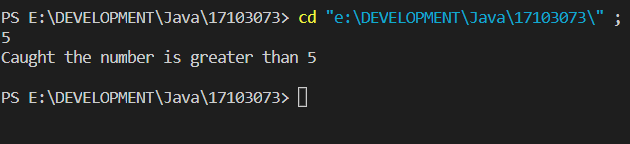
catch(ExceptionA eA) {

System.out.println("Caught " + eA);

}

}

}



**2. Write a program that demonstrates how various exceptions are caught with catch(Exception exception).**

**This time, define classes ExceptionA (which inherits from class Exception) and ExceptionB (which inherits from class ExceptionA). In your program, create try blocks that throw exceptions of types ExceptionA, ExceptionB, NullPointerException and IOException. All exceptions should be caught with catch blocks specifying type Exception.**

import java.io.IOException;

import java.util.Scanner;

class A extends Exception

{

public String toString()

{

return "Exception A";

}

}

class B extends A

{

public String toString()

{

return "Exception B";

}

}

class demo

{

void call(int a) throws A,B, IOException

{

if(a==0)

throw new NullPointerException("Null pointer");

if(a==-1)

throw new IOException("device error");

if(a<50)

throw new A();

if(a<100)

throw new B();

}

}

public class exception2 {

public static void main(String[] args) {

demo d=new demo();

int k;

System.out.println("Enter the value of k ");

Scanner input=new Scanner(System.in);

k=input.nextInt();

try {

d.call(k);

}catch(Exception e)

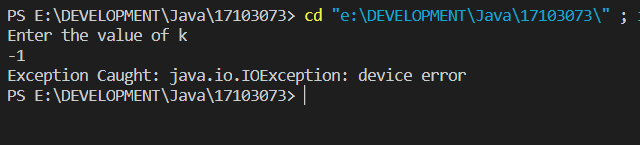
{

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

}

}

}



**3. Write a program that shows that the order of catch blocks is important. If you try a catch a superclass exception type before a subclass type, the compiler should generate errors.**

Import java.util.\*;

class rethrow

{

public void somemethod2() throws ArithmeticException

{

int a=4/0;

}

public void somemethod() throws ArithmeticException

{

try

{

somemethod2();

}

catch(Exception e)

{

System.*out*.println("Exception Caught in somemethod: "+e);

throw new ArithmeticException();

}

}

}

public class exception4 {

public static void main(String[] args) {

rethrow r=new rethrow();

try

{

r.somemethod();

}

catch(Exception e)

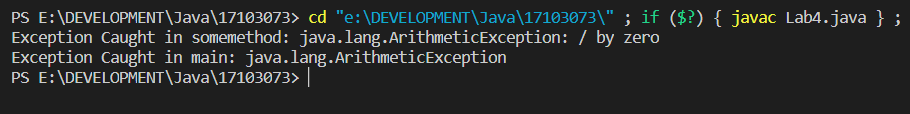
{

System.*out*.println("Exception Caught in main: "+e );

}

}

}



**5. Write an application to define two different Exceptions. Enter an issue\_date and a return\_date and define two different exceptions. First, if difference of issue-date and return\_date is more than 15 days, throw an exception displayking calculated fine(Rs. 50 per day) after 15 days. Second, if return date is before issue\_date, throw an excpetion displaying “Return date < Issue date”.**

import java.sql.Date;

import java.util.Scanner;

class A extends Exception

{ public String toString()

{

return "Date less Than 15";

}

}

class B extends Exception

{

public String toString(){

return "Return\_date<Issue\_date";

}

}

class dot

{

public void check(Date d1, Date d2) throws A,B{

if(d1.after(d2)){

throw new B();

}

int diffInDays = (int) ((d2.getTime() - d1.getTime()) / (1000 \* 60 \* 60 \* 24));

if(diffInDays<15){

throw new A();

}

System.out.println("The Charges are: "+ diffInDays\*5);

}

}

public class exception5 {

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

int d1,d2,m1,m2,y1,y2;

Scanner input=new Scanner(System.in);

System.out.println("Enter Issue date(dd/mm/yy)");

d1=input.nextInt();

m1=input.nextInt();

y1=input.nextInt();

System.out.println("Enter Return Date(dd/mm/yy)");

d2=input.nextInt();

m2=input.nextInt();

y2=input.nextInt();

Date dd1 = new Date(y1,m1,d1);

Date dd2= new Date(y2,m2,d2);

try{

dot p=new dot();

p.check(dd1,dd2);

}

catch(A e){

System.out.println("Exception Caught: "+ e.toString());

System.out.println("Description: "+e.toString());

}

catch(B b){

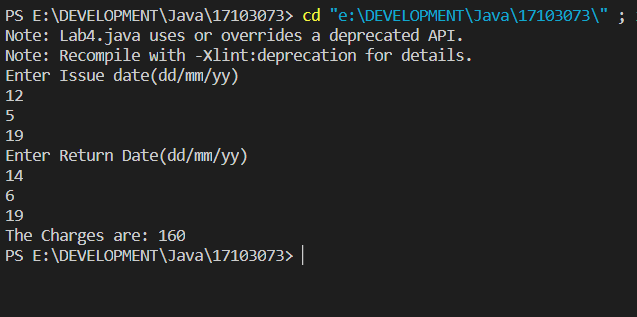
System.out.println("Exception Caught: "+b.toString());

System.out.println("Description: "+b.toString());

}

}

}



**LAB – 7**

**1. Write an application of multithreading. Create three different classes(threads) that inherit Thread class. Each class consists of a for loop that prints identity of the class with a number series in increasing order. Start all three threads together. Now run the application 2 or 3 timese and show the output.**

class NewThread1 extends Thread {

String name;

NewThread1(String threadName) {

super("Thread: ");

name = threadName;

System.out.println("New Thread1 : " + this);

start();

}

public void run() {

try {

for (int i = 5; i > 0; --i) {

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

Thread.sleep(1000);

}

} catch (InterruptedException e) {

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

}

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

}

}

class NewThread2 extends Thread {

String name;

NewThread2(String threadName) {

super("Thread: ");

name = threadName;

System.out.println("New Thread2 : " + this);

start();

}

public void run() {

try {

for (int i = 5; i > 0; --i) {

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

Thread.sleep(1000);

}

} catch (InterruptedException e) {

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

}

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

}

}

class NewThread3 extends Thread {

String name;

NewThread3(String threadName) {

super("Thread: ");

name = threadName;

System.out.println("New Thread3 : " + this);

start();

}

public void run() {

try {

for (int i = 5; i > 0; --i) {

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

Thread.sleep(1000);

}

} catch (InterruptedException e) {

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

}

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

}

}

public class Lab4 {

public static void main(String[] args) {

NewThread1 obj1 = new NewThread1("first thread");

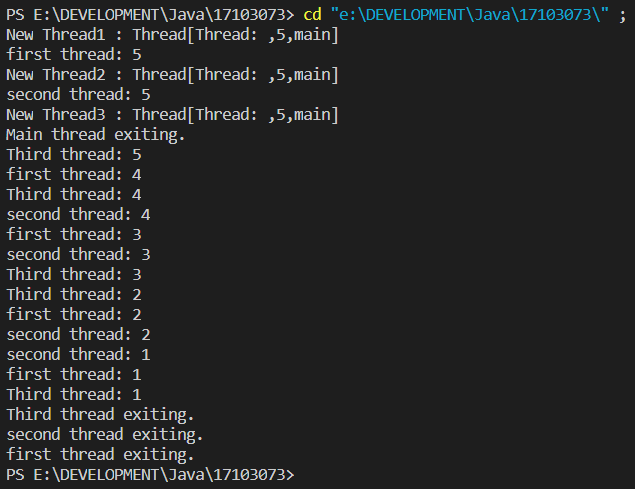
NewThread2 obj2 = new NewThread2("second thread");

NewThread3 obj3 = new NewThread3("Third thread");

System.out.println("Main thread exiting.");

}

}



**2. Write an application of multithreading to create three different threads namely: One, Two and Three. Crete the threads implementing Runnable Interace. Start all three threads together and print the thread with increasing number till the thread exit. Make sure that main thread exits at last.**

**NOTE : Use Thread.sleep().**

class NewThread1 implements Runnable {

    String name;

    Thread t;

    NewThread1(String threadName) {

        name = threadName;

        t = new Thread(this, name);

        System.out.println("New Thread : " + t);

        t.start();

    }

    public void run() {

        try {

            for (int i = 5; i > 0; --i) {

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

                Thread.sleep(1000);

            }

        } catch (InterruptedException e) {

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

        }

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

    }

}

class NewThread2 implements Runnable {

    String name;

    Thread t;

    NewThread2(String threadName) {

        name = threadName;

        t = new Thread(this, name);

        System.out.println("New Thread : " + t);

        t.start();

    }

    public void run() {

        try {

            for (int i = 5; i > 0; --i) {

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

                Thread.sleep(1000);

            }

        } catch (InterruptedException e) {

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

        }

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

    }

}

class NewThread3 implements Runnable {

    String name;

    Thread t;

    NewThread3(String threadName) {

        name = threadName;

        t = new Thread(this, name);

        System.out.println("New Thread : " + t);

        t.start();

    }

    public void run() {

        try {

            for (int i = 5; i > 0; --i) {

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

                Thread.sleep(1000);

            }

        } catch (InterruptedException e) {

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

        }

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

    }

}

public class Lab4 {

    public static void main(String[] args) {

        NewThread1 n1 = new NewThread1("One");

        NewThread2 n2 = new NewThread2("Two");

        NewThread3 n3 = new NewThread3("Thread");

        try {

            n1.t.join();

            n2.t.join();

            n3.t.join();

        } catch (InterruptedException e) {

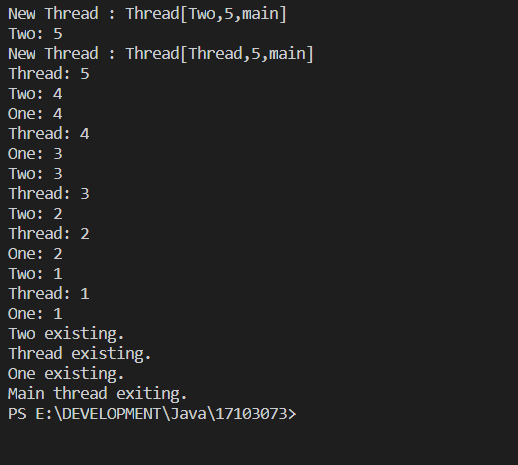
            System.out.println("Main Thread Interrupted.");

        }

        System.out.println("Main thread exiting.");

    }

}



**3. Write an application of multithreading with priority. Create three different threads of min, max and norm priority each. Now run each thread together for 10 seconds. Each thread should consist of a variable count that increases itself with time(click++). Stop all threads after 10 seconds and print the value of click for each and show the use priority in multithreading.**

class clicker implements Runnable {

    long click = 0;

    Thread t;

    private volatile boolean running = true;

    public clicker(int p) {

        t = new Thread(this);

        t.setPriority(p);

    }

    public void run() {

        while (running) {

            click++;

        }

    }

    public void stop() {

        running = false;

    }

    public void start() {

        t.start();

    }

}

public class Lab4 {

    public static void main(String[] args) {

        clicker hi = new clicker(Thread.MAX\_PRIORITY);

        clicker medium = new clicker(Thread.NORM\_PRIORITY);

        clicker low = new clicker(Thread.MIN\_PRIORITY);

        low.start();

        hi.start();

        medium.start();

        try {

            Thread.sleep(10000);

        } catch (InterruptedException e) {

            System.out.println("Main Thread interrupted");

        }

        low.stop();

        hi.stop();

        medium.stop();

        try {

            hi.t.join();

            low.t.join();

            medium.t.join();

        } catch (InterruptedException e) {

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

        }

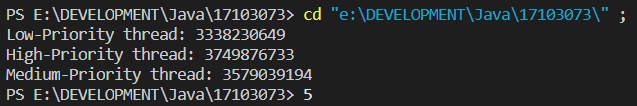
        System.out.println("Low-Priority thread: " + low.click);

        System.out.println("High-Priority thread: " + hi.click);

        System.out.println("Medium-Priority thread: " + medium.click);

    }

}



**4. Inherit a class from Thread and override the run() method. Inside run(), print a message, and then call sleep(). Repeat this three times, then return from run(). Put a start-up message in the constructor and override finalize() to print a shut down message. Make a separate thread class that calls System.gc() and System.runFinalization() inside run(), printing a message as it does so. Make several thread objects of both types and run them to see what happens.**

class Aaa extends Thread {

    Aaa(String threadName) {

        super(threadName);

        System.out.println("started " + this);

        start();

    }

    public void run() {

        try {

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

            Thread.sleep(2000);

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

            Thread.sleep(2000);

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

        } catch (InterruptedException e) {

            System.out.println(this + " Interrupted");

        }

    }

    public void finalize() {

        System.out.println(this + " exiting.");

    }

}

class Bbb extends Thread {

    Bbb(String threadName) {

        super(threadName);

        System.out.println("started " + this);

        start();

    }

    public void run() {

        try {

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

            System.gc();

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

            System.runFinalization();

            Thread.sleep(1);

        } catch (InterruptedException e) {

            System.out.println(this + " Interrupted");

        }

        System.out.println(this + " exiting.");

    }

}

public class Lab4 {

    public static void main(String args[]) {

        Aaa a = new Aaa("a");

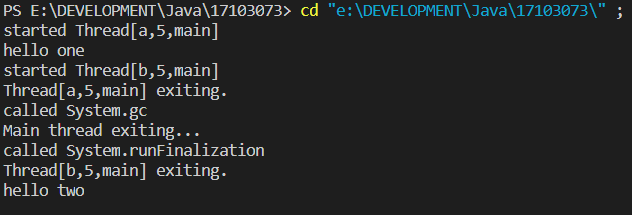
        Bbb b = new Bbb("b");

        a.finalize();

        System.out.println("Main thread exiting...");

    }

}



**5. Create two Thread subclasses, one with a run() that starts up, captures the reference of the second Thread object and then calls wait(). The other class’ run() should call notifyAll() for the first thread after some number of seconds have passed, so the first thread can print a message.**

class aA implements Runnable {

    Thread t;

    aA() {

        t = new Thread(this, "one");

        System.out.println("Constructor Called A");

        t.start();

    }

    public void run() {

        synchronized (this) {

            System.out.println("Thread one is running");

            notifyAll();

        }

    }

}

class bB implements Runnable {

    aA a;

    Thread t;

    bB(aA a) {

        this.a = a;

        t = new Thread(this, "Second");

        System.out.println("Constructor Called B");

        t.start();

    }

    public void run() {

        synchronized (this) {

            System.out.println("Thread two is running");

            try {

                a.t.start();

                wait();

            } catch (Exception e) {

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

            }

            System.out.println("Thread two resumed");

        }

    }

}

public class Lab4 {

    public static void main(String args[]) {

        System.out.println("Main thread running");

        aA a = new aA();

        bB b = new bB(a);

        try {

            Thread.sleep(1000);

        } catch (InterruptedException e) {

            // TODO Auto-generated catch block

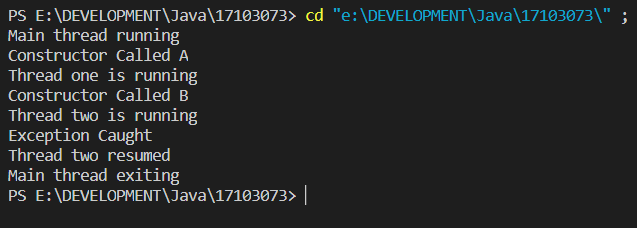
            e.printStackTrace();

        }

        System.out.println("Main thread exiting");

    }

}



**6. There are two processes, a producer and a consumer, that share a common buffer with a limited size. The producer “produces” data and stores it in the buffer, and the consumer “consumes” the data, removing it from the buffer. Having two processes that run in parallel, we need to make sure that the producer won’t put new data in the buffer when the buffer is full nad the consumer won’t try to remove data from the buffer if the buffer is empty.**

import java.util.concurrent.Semaphore;

class Q {

    int item;

    static Semaphore semCon = new Semaphore(0);

    static Semaphore semProd = new Semaphore(1);

    void get() {

        try {

            semCon.acquire();

        } catch (InterruptedException e) {

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

        }

        System.out.println("Consumer consumed item : " + item);

        semProd.release();

    }

    void put(int item) {

        try {

            semProd.acquire();

        } catch (InterruptedException e) {

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

        }

        this.item = item;

        System.out.println("Producer produced item : " + item);

        semCon.release();

    }

}

class Producer implements Runnable {

    Q q;

    Producer(Q q) {

        this.q = q;

        Thread t = new Thread(this, "Producer");

        t.start();

    }

    public void run() {

        for (int i = 0; i < 5; i++)

            q.put(i);

    }

}

class Consumer implements Runnable {

    Q q;

    Consumer(Q q) {

        this.q = q;

        new Thread(this, "Consumer").start();

    }

    public void run() {

        for (int i = 0; i < 5; i++)

            q.get();

    }

}

public class Lab4 {

    public static void main(String args[]) {

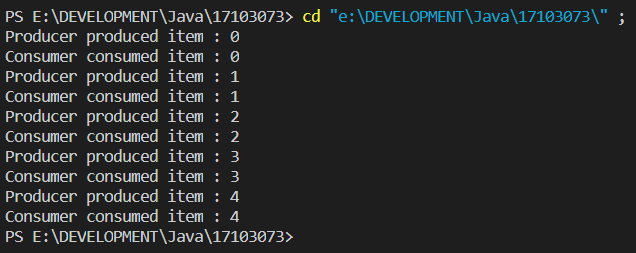
        Q q = new Q();

        new Consumer(q);

        new Producer(q);

    }

}



**LAB – 8**

**1. Write an applet program to draw a string, “This is my first Applet” on given coordinates and run the applet in both the browser and applet viewer.**

import java.applet.\*;

import java.awt.\*;

/\*\*

\*

\* @author hp

\*/

public class NewApplet extends Applet {

/\*\*

\* Initialization method that will be called after the applet is loaded into

\* the browser.

\*/

@Override

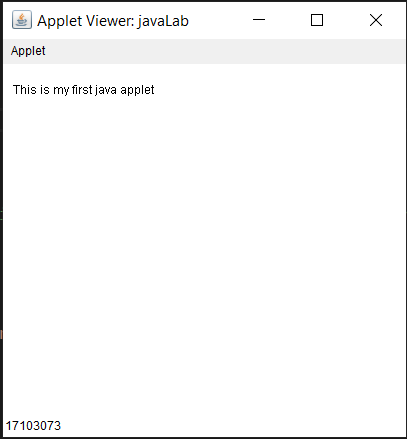
public void paint(Graphics g){

String msg = "This is my first java applet";

g.drawString(msg, 10, 30);

}

}



**2 .Write an applet program that asks the user to enter two floaing point numbers obtains the two numbers from the user and draws their sum, product, difference and division.**

**Note : Use PARAM tag for user input.**

import java.awt.\*;

import java.applet.\*;

public class Lab8 extends Applet {

Float num1, num2;

public void init() {

String s;

s = getParameter("num1");

num1 = Float.valueOf(s).floatValue();

s = getParameter("num2");

num2 = Float.valueOf(s).floatValue();

}

public void paint(Graphics g) {

String msg = "The Product is : " + (num1 \* num2);

g.drawString(msg, 10, 40);

msg = "The Sum is : " + (num1 + num2);

g.drawString(msg, 10, 60);

msg = "The Division is : " + (num1 / num2);

g.drawString(msg, 10, 80);

showStatus("Ritik 17103073");

}

}

<!DOCTYPE html>

<html lang="en">

<body>

<applet code="Lab8" width="350" height="350">

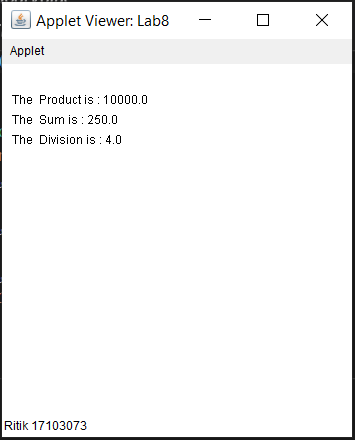
<param name="num1" value="200">

<param name="num2" value="50">

</applet>

</body>

</html>



**3. Write an applet that draws a checkerboard pattern as follows:**

**\*\*\*\***

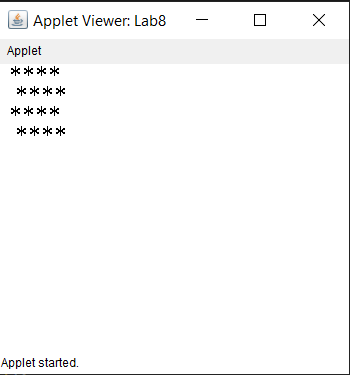
**\*\*\*\***

**\*\*\*\***

**\*\*\*\***

import java.applet.\*;  
import java.awt.\*;  
  
public class Applet2 extends Applet{  
  
 String msg1="\*\*\*\* ";  
 String msg2=" \*\*\*\*";  
  
 public void init(){  
 setBackground(Color.green);  
 setForeground(Color.black);  
 }  
 public void start(){}  
 public void stop(){}  
 public void destroy(){}

public void paint(Graphics g){  
 g.setFont(new Font("TimesRoman", Font.PLAIN, 25));  
 g.drawString(msg1, 10, 20);  
 g.drawString(msg2, 10, 40);  
 g.drawString(msg1, 10, 60);  
 g.drawString(msg2, 10, 80);  
 }  
}



**4. Write an applet in which background is Red and foreground (text color) is Blue. Also show the message “Background is Red and Foreground is Blue” in the status window.**

import java.applet.\*;

import java.awt.\*;

/\*\*

\*

\* @author hp

\*/

public class java\_17103073 extends Applet {

public void init() {

setBackground(Color.red);

setForeground(Color.blue);

}

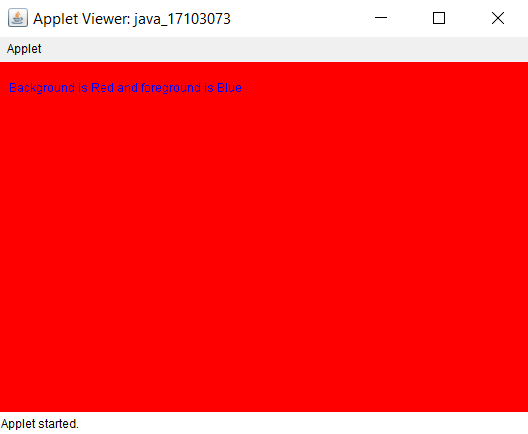
public void paint(Graphics g) {

String msg = "Background is Red and foreground is Blue";

g.drawString(msg, 10, 30);

}

}



**5. Write an applet program showing URL of code base and document vase in the applet window. NOTE : Use getCodeBase() and getDocumentBase().**

import java.applet.Applet;

import java.awt.\*;

import java.net.\*;

public class Applet4 extends Applet {

public void paint(Graphics g){

g.setFont(new Font("TimesRoman", Font.PLAIN, 15));

String msg;

URL url= getCodeBase();

msg= "Code Base: "+ url.toString();

g.drawString(msg, 10, 20);

url= getDocumentBase();

msg= "Document Base: "+url.toString();

g.drawString(msg, 10, 60);

}

}



**LAB 9**

**1. Write an applet to print the message of click, enter, exit, press and release messages whom respective Mouse event happens in the applet and print dragged and moved when respective mouse motion event happens in the applet.**

import java.awt.\*;

import java.awt.event.\*;

import java.applet.\*;

public class Lab8 extends Applet implements MouseListener, MouseMotionListener {

    String msg = "";

    int mouseX = 0;

    int mouseY = 0;

    public void init() {

        addMouseListener(this);

        addMouseMotionListener(this);

    }

    public void mouseClicked(MouseEvent me) {

        mouseX = 0;

        mouseY = 10;

        msg = "Mouse clicked.";

        repaint();

    }

    public void mouseEntered(MouseEvent me) {

        mouseX = 0;

        mouseY = 10;

        msg = "Mouse entered.";

        repaint();

    }

    public void mouseExited(MouseEvent me) {

        mouseX = 0;

        mouseY = 10;

        msg = "Mouse exited.";

        repaint();

    }

    public void mousePressed(MouseEvent me) {

        mouseX = me.getX();

        mouseY = me.getY();

        msg = "Down";

        repaint();

    }

    public void mouseReleased(MouseEvent me) {

        mouseX = me.getX();

        mouseY = me.getY();

        msg = "Up";

        repaint();

    }

    public void mouseDragged(MouseEvent me) {

        mouseX = me.getX();

        mouseY = me.getY();

        msg = "\*";

        showStatus("Dragging mouse at " + mouseX + ", " + mouseY);

        repaint();

    }

    public void mouseMoved(MouseEvent me) {

        showStatus("Moving mouse at " + me.getX() + ", " + me.getY());

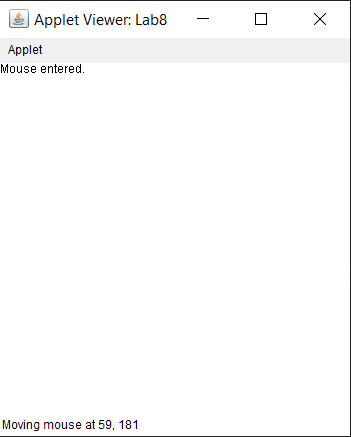
    }

    public void paint(Graphics g) {

        g.drawString(msg, mouseX, mouseY);

    }

}



**2.Write an applet to print the message of click, enter, exit, press and release messages when respective Mouse event happens in the applet and print dragged and moved when mouse motion event happens in the applet.**

import java.awt.\*;

import java.awt.event.\*;

import java.applet.\*;

public class Lab8 extends Applet {

    public String msg = "";

    public void init() {

        addMouseListener(new MyMouseAdapter(this));

        addMouseMotionListener(new MyMouseMotionAdapter(this));

    }

    public void paint(Graphics g) {

        g.drawString(msg, 15, 15);

    }

}

class MyMouseAdapter extends MouseAdapter {

    Lab8 adapterDemo;

    public MyMouseAdapter(Lab8 adapterDemo) {

        this.adapterDemo = adapterDemo;

    }

    // Handle mouse clicked.

    public void mouseClicked(MouseEvent me) {

        adapterDemo.showStatus("Mouse clicked");

        adapterDemo.msg = "Mouse clicked";

        adapterDemo.repaint();

    }

    public void mouseEntered(MouseEvent me) {

        adapterDemo.showStatus("Mouse entered");

        adapterDemo.msg = "Mouse entered";

        adapterDemo.repaint();

    }

    public void mouseExited(MouseEvent me) {

        adapterDemo.showStatus("Mouse exited");

        adapterDemo.msg = "Mouse exited";

        adapterDemo.repaint();

    }

   public void mousePressed(MouseEvent me) {

        adapterDemo.showStatus("Mouse pressed");

        adapterDemo.msg = "Mouse pressed";

        adapterDemo.repaint();

    }

    public void mouseReleased(MouseEvent me) {

        adapterDemo.showStatus("Mouse released");

        adapterDemo.msg = "Mouse released";

        adapterDemo.repaint();

    }

}

class MyMouseMotionAdapter extends MouseMotionAdapter {

    Lab8 adapterDemo;

    public MyMouseMotionAdapter(Lab8 adapterDemo) {

        this.adapterDemo = adapterDemo;

    }

    // Handle mouse dragged.

    public void mouseDragged(MouseEvent me) {

        adapterDemo.showStatus("Mouse dragged");

        adapterDemo.msg = "mouse dragged";

        adapterDemo.repaint();

    }

    public void mouseMoved(MouseEvent me) {

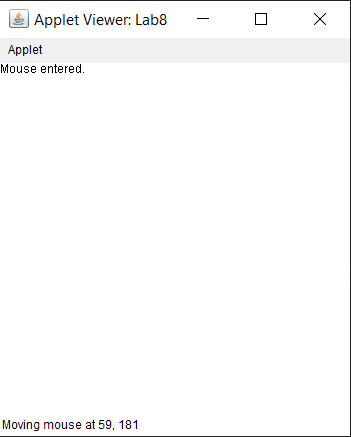
        adapterDemo.showStatus("Mouse moved");

        adapterDemo.msg = "mouse moved";

        adapterDemo.repaint();

    }

}



**3.Write an applet to print the message implementing KeyListener Interface. Also show the status of key press and release in status window.**

import java.awt.\*;

import java.awt.event.\*;

import java.applet.\*; /\*   <applet code="KeyEvents" width=300 height=100>   </applet> \*/

public class Lab8 extends Applet implements KeyListener {

    String msg = "";

    int X = 10, Y = 20; // output coordinates

    public void init() {

        addKeyListener(this);

    }

    public void paint(Graphics g) {

        g.drawString(msg, X, Y);

    }

    public void keyPressed(KeyEvent ke) {

        showStatus("Key Down");

    }

    public void keyReleased(KeyEvent ke) {

        Frame f = new Frame("behind");

        f.setTitle("17103073");

        showStatus("Key Up");

    }

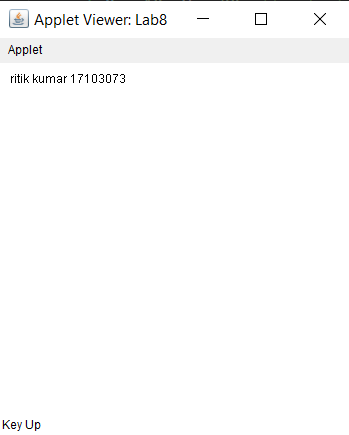
    public void keyTyped(KeyEvent ke) {

        msg += ke.getKeyChar();

        repaint();

    }

}



**4. Write an applet to print the message extending KeyAdapter class in Inner class and anonymous Inner class. Also show the status of key press and release in status window.**

import java.awt.\*;

import java.applet.\*;

import java.awt.event.\*;

public class Lab8 extends Applet {

    String msg = "";

    public void init() {

        addKeyListener(new KeyAdapter() {

            public void keyPressed(KeyEvent ke) {

                showStatus("Key pressed = " + ke.getKeyCode());

            }

            public void keyReleased(KeyEvent ke) {

                msg = msg + ke.getKeyChar();

                repaint();

            }

        });

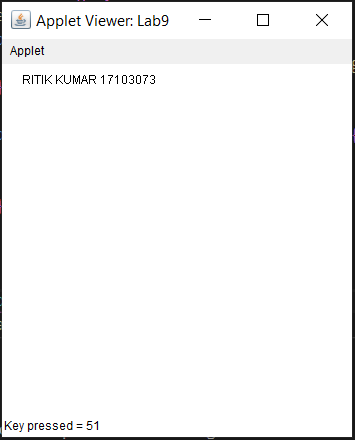
    }

    public void paint(Graphics g) {

        g.drawString(msg, 20, 20);

    }

}



**5. Write an applet demonstrating some virtual key codes i.e. Function keys, Page up, Page Down and arrow keys. To demonstrate only print the message of the key pressed. Also show the status of key press and release.**

import java.applet.\*;

import java.awt.\*;

import java.awt.event.\*;

public class Lab8 extends Applet implements KeyListener {

    String msg = "";

    int x = 20;

    int y = 20;

    public void init() {

        addKeyListener(this);

    }

    public void keyPressed(KeyEvent ke) {

        showStatus("key pressed");

        switch (ke.getKeyCode()) {

        case (KeyEvent.VK\_F1):

            msg += "<F1>";

            break;

        case (KeyEvent.VK\_F2):

            msg += "<F2>";

            break;

        case (KeyEvent.VK\_PAGE\_DOWN):

            msg += "Page Down";

            break;

        case (KeyEvent.VK\_PAGE\_UP):

            msg += "page up";

            break;

        case (KeyEvent.VK\_RIGHT):

            msg += "right arrow";

            break;

        }

        repaint();

    }

    public void keyReleased(KeyEvent ke) {

        showStatus("key up");

    }

  public void keyTyped(KeyEvent ke) {

        showStatus("keyTyped with code= " + ke.getKeyCode() + " and char " + ke.getKeyChar());

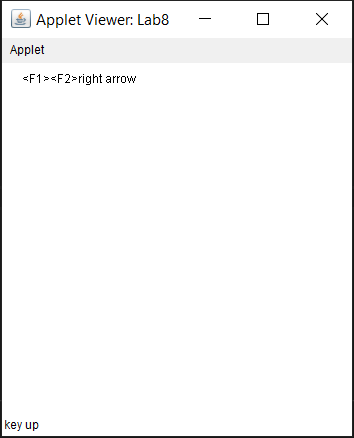
    }

    public void paint(Graphics g) {

        g.drawString(msg, x, y);

    }

}



**LAB - 10**

**Write an applet to design a calculator. Calculations should be atleast done with two numbers.**

import java.awt.\*;

import java.awt.event.\*;

import java.applet.\*;

public class Lab8 extends Applet implements ActionListener {

    String msg = " ";

    int v1, v2, result;

    TextField t1;

    Button b[] = new Button[10];

    Button add, sub, mul, div, EQ;

    char OP;

    public void init() {

        Color k = new Color(120, 89, 90);

        setBackground(k);

        t1 = new TextField(10);

        GridLayout gl = new GridLayout(4, 4);

        setLayout(gl);

        for (int i = 0; i < 10; i++) {

            b[i] = new Button("" + i);

        }

        add = new Button("add");

        sub = new Button("sub");

        mul = new Button("mul");

        div = new Button("div");

        EQ = new Button("EQ");

        t1.addActionListener(this);

        add(t1);

        for (int i = 0; i < 10; i++) {

            add(b[i]);

        }

        add(add);

        add(sub);

        add(mul);

        add(div);

        add(EQ);

        for (int i = 0; i < 10; i++) {

            b[i].addActionListener(this);

        }

        add.addActionListener(this);

        sub.addActionListener(this);

        mul.addActionListener(this);

        div.addActionListener(this);

        EQ.addActionListener(this);

    }

    public void actionPerformed(ActionEvent ae) {

        String str = ae.getActionCommand();

        char ch = str.charAt(0);

        if (Character.isDigit(ch))

            t1.setText(t1.getText() + str);

        else if (str.equals("add")) {

            v1 = Integer.parseInt(t1.getText());

            OP = '+';

            t1.setText("");

        } else if (str.equals("sub")) {

            v1 = Integer.parseInt(t1.getText());

            OP = '-';

            t1.setText("");

        } else if (str.equals("mul")) {

            v1 = Integer.parseInt(t1.getText());

            OP = '\*';

            t1.setText("");

        } else if (str.equals("div")) {

            v1 = Integer.parseInt(t1.getText());

            OP = '/';

            t1.setText("");

        }

        if (str.equals("EQ")) {

            v2 = Integer.parseInt(t1.getText());

            if (OP == '+')

                result = v1 + v2;

            else if (OP == '-')

                result = v1 - v2;

            else if (OP == '\*')

                result = v1 \* v2;

            else if (OP == '/')

                result = v1 / v2;

            else if (OP == '%')

                result = v1 % v2;

            t1.setText("" + result);

        }

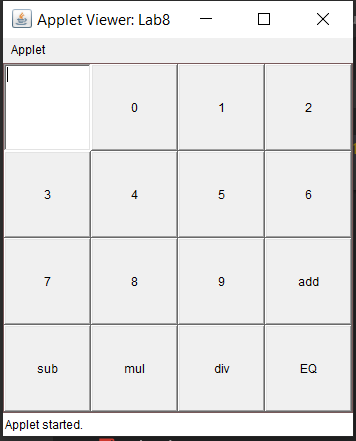
    }

    public void paint(Graphics g) {

        showStatus("Ritik\_17103073");

    }

}



**2.Write an applet displaying three buttons and three checkboxes labelled Rectangle ,Oval, Line Red, Green and Blue respectively. Fill the Rectangle or Oval or draw the line with respective colour when it is checked.**

import java.awt.\*;

import java.awt.event.\*;

public class Lab8 extends Frame implements ActionListener, ItemListener {

    Checkbox c1, c2, c3;

    Button b1, b2, b3;

    CheckboxGroup c;

    int a, b;

    Lab8() {

        setLayout(new FlowLayout());

        c = new CheckboxGroup();

        c1 = new Checkbox("Red", c, false);

        c2 = new Checkbox("Green", c, false);

        c3 = new Checkbox("Blue", c, false);

        b1 = new Button("Rectangle");

        b2 = new Button("Oval");

        b3 = new Button("Line");

        add(b1);

        add(b2);

        add(b3);

        add(c1);

        add(c2);

        add(c3);

        b1.addActionListener((ae) -> {

            b = 1;

            repaint();

        });

        b2.addActionListener((ae) -> {

            b = 2;

            repaint();

        });

        b3.addActionListener((ae) -> {

            b = 3;

            repaint();

        });

        c1.addItemListener(this);

        c2.addItemListener(this);

        c3.addItemListener(this);

        addWindowListener(new WindowAdapter() {

            public void windowClosing(WindowEvent we) {

                System.exit(0);

            }

        });

    }

    public void itemStateChanged(ItemEvent arg0) {

        repaint();

    }

    @Override

    public void actionPerformed(ActionEvent arg0) {

        // TODO Auto-generated method stub

    }

    public void paint(Graphics g) {

        String t;

        t = c.getSelectedCheckbox().getLabel();

        if (b == 1) { // rectangle

            if (t.equals("Red")) {

                setForeground(Color.RED);

                g.fillRect(100, 100, 20, 20);

            } else if (t.equals("Green")) {

                setForeground(Color.GREEN);

                g.fillRect(100, 100, 20, 20);

            } else if (t.equals("Blue")) {

                setForeground(Color.BLUE);

                g.fillRect(100, 100, 20, 20);

            }

        } else if (b == 2) { // oval

            if (t.equals("Red")) {

                setForeground(Color.RED);

                g.fillOval(100, 100, 20, 20);

            } else if (t.equals("Green")) {

                setForeground(Color.GREEN);

                g.fillOval(100, 100, 20, 20);

            } else if (t.equals("Blue")) {

                setForeground(Color.BLUE);

                g.fillOval(100, 100, 20, 20);

            }

        } else if (b == 3) { // line

            if (t.equals("Red")) {

                setForeground(Color.RED);

                g.drawLine(100, 100, 120, 120);

            } else if (t.equals("Green")) {

                setForeground(Color.GREEN);

                g.drawLine(100, 100, 120, 120);

            } else if (t.equals("Blue")) {

                setForeground(Color.BLUE);

                g.drawLine(100, 100, 120, 120);

            }

        }

    }

    public static void main(String[] args) {

        Lab8 d = new Lab8();

        d.setSize(new Dimension(340, 260));

        d.setTitle("Ritik");

        d.setVisible(true);

    }

}



**3.Write an applet to draw a square and two buttons labelled Enlarge and Shrink respectively. Enlarge and shrink the square when respective buttons are pressed.**

import java.awt.\*;

import java.awt.event.\*;

public class Lab8 extends Frame implements ActionListener {

    Button b1, b2;

    int s = 50;

    Lab8() {

        setLayout(new FlowLayout());

        b1 = new Button("Enlarge");

        b2 = new Button("Shrink");

        setForeground(Color.BLACK);

        add(b1);

        add(b2);

        b1.addActionListener((ae) -> {

            s = s + 10;

            repaint();

        });

        b2.addActionListener((ae) -> {

            s = s - 10;

            if (s < 20)

                s = 20;

            repaint();

        });

    }

    public void actionPerformed(ActionEvent arg0) {

    }

    public void paint(Graphics g) {

        g.drawRect(100, 100, s, s);

    }

    static public void main(String args[]) {

        Lab8 d = new Lab8();

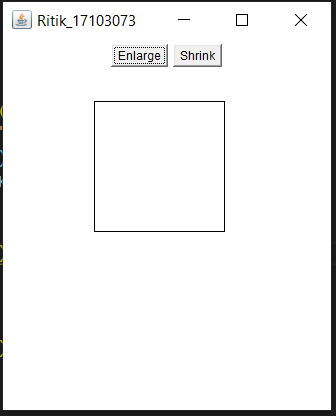
        d.setSize(new Dimension(340, 260));

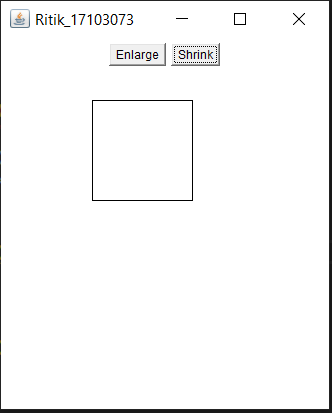
        d.setTitle("Ritik\_17103073");

        d.setVisible(true);

    }

}





**LAB 11**

**1.Write a program taking 16 buttons labelled 1 to 16 in a frame and represent them with FlowLayout and GridLayout. Print “Button <#> is pressed” message when a button is pressed.**

import java.awt.\*;

import java.awt.event.\*;

public class lab11\_1 extends Frame implements ActionListener{

String msg="";

Button []b;

lab11\_1()

{

setLayout(new FlowLayout());

int k;

b=new Button[16];

for( int i=0; i<16;++i)

{ k=i+1;

add(b[i]=new Button(""+k));

}

for( int i=0;i<16;++i)

{

b[i].addActionListener(this);

}

addWindowListener(new WindowAdapter() {

public void windowClosing(WindowEvent we)

{

System.*exit*(0);

}

});

}

public void actionPerformed(ActionEvent ae) {

String s=ae.getActionCommand();

msg="Button "+ s+ " is pressed ";

repaint();

}

public void paint(Graphics g)

{

g.drawString(msg, 200, 200);

}

public static void main(String args[])

{

lab11\_1 d= new lab11\_1();

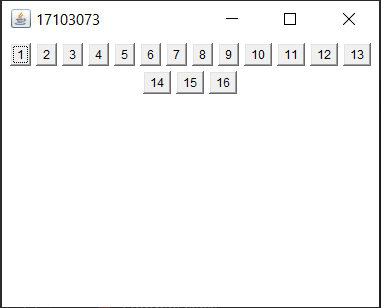
d.setSize(new Dimension(340,260));

d.setTitle("17103073");

d.setVisible(true);

}

}



import java.awt.\*;

import java.awt.event.\*;

public class lab11\_1b extends Frame implements ActionListener{

Label l;

String msg="NONE";

Button []b;

lab11\_1b()

{ l=new Label("NoNe");

setLayout(new GridLayout(5,4));

int k;

b=new Button[16];

for( int i=0; i<16;++i)

{ k=i+1;

add(b[i]=new Button(""+k));

}

add(l);

for( int i=0;i<16;++i)

{

b[i].addActionListener(this);

}

addWindowListener(new WindowAdapter() {

public void windowClosing(WindowEvent we)

{

System.exit(0);

}

});

}

public void actionPerformed(ActionEvent ae) {

String s=ae.getActionCommand();

msg="Button "+ s+ "\n is pressed ";

repaint();

}

public void paint(Graphics g)

{

l.setText(msg);

}

public static void main(String[] args) {

// TODO Auto-generated method stub

lab11\_1b d= new lab11\_1b();

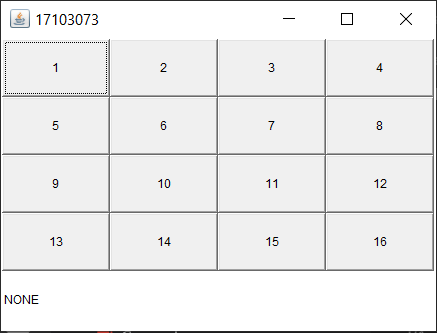
d.setSize(new Dimension(340,260));

d.setTitle("17103073");

d.setVisible(true);

}

}



**2.Write a program showing use of BorderLayout in a frame showing the directions i.e. North, East, South and West on buttons.**

import java.awt.\*;

import java.awt.event.\*;

public class lab11\_2 extends Frame implements ActionListener {

Button a,b,c,d,e;

lab11\_2()

{

setLayout(new BorderLayout());

a=new Button("North");

b=new Button("South");

d=new Button("West");

c=new Button("East");

e=new Button("Center");

add(a,BorderLayout.NORTH);

add(b,BorderLayout.SOUTH);

add(c,BorderLayout.EAST);

add(d,BorderLayout.WEST);

add(e,BorderLayout.CENTER);

addWindowListener(new WindowAdapter() {

public void windowClosing(WindowEvent we)

{

System.exit(0);

}

});

}

public void actionPerformed(ActionEvent e) {

}

public static void main(String args[])

{

lab11\_2 d= new lab11\_2();

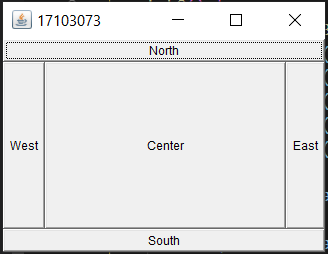
d.setSize(new Dimension(340,260));

d.setTitle("17103073");

d.setVisible(true);

}

}



**3.Write a program for menu demonstration same as in notepad extending Frame class. Menu bar should contain File,Edit,View and its submenus as in notepad. Show the message when clicked on the menu.**

import java.awt.\*;

import java.awt.event.\*;

public class lab11\_3 extends Frame implements ActionListener {

TextArea t;

lab11\_3()

{

MenuBar mbar=new MenuBar();

setMenuBar(mbar);

MenuItem i1,i2,i3,i4;

t=new TextArea("",100,100);

/\*

\* textArea.setLineWrap(true);

textArea.setWrapStyleWord(true)

\*/

add(t);

Menu file=new Menu("File");

file.add(i1=new MenuItem("New"));

file.add(i2=new MenuItem("Open..."));

file.add(i3=new MenuItem("Save"));

file.add(i4=new MenuItem("Save As..."));

mbar.add(file);

Menu edit=new Menu("Edit");

MenuItem e1,e2,e3,e4,e5;

edit.add(e1=new MenuItem("Undo"));

edit.add(e2=new MenuItem("Cut"));

edit.add(e3=new MenuItem("Copy"));

edit.add(e4=new MenuItem("Paste"));

edit.add(e5=new MenuItem("Delete"));

mbar.add(edit);

MenuItem f1,f2;

Menu format=new Menu("Format");

format.add(f1=new MenuItem("Word Wrap"));

format.add(f2=new MenuItem("Font..."));

mbar.add(format);

addWindowListener(new WindowAdapter() {

public void windowClosing(WindowEvent we)

{

System.exit(0);

}

});

Myhand hand=new Myhand();

i1.addActionListener(hand);

i2.addActionListener(hand);

i3.addActionListener(hand);

i4.addActionListener(hand);

e1.addActionListener(hand);

e2.addActionListener(hand);

e3.addActionListener(hand);

e4.addActionListener(hand);

e5.addActionListener(hand);

f1.addActionListener(hand);

f2.addActionListener(hand);

}

class Myhand implements ActionListener

{

@Override

public void actionPerformed(ActionEvent e) {

String k=e.getActionCommand();

t.setText("Pressed "+ k);

}

}

public static void main(String[] args) {

// TODO Auto-generated method stub

lab11\_3 d= new lab11\_3();

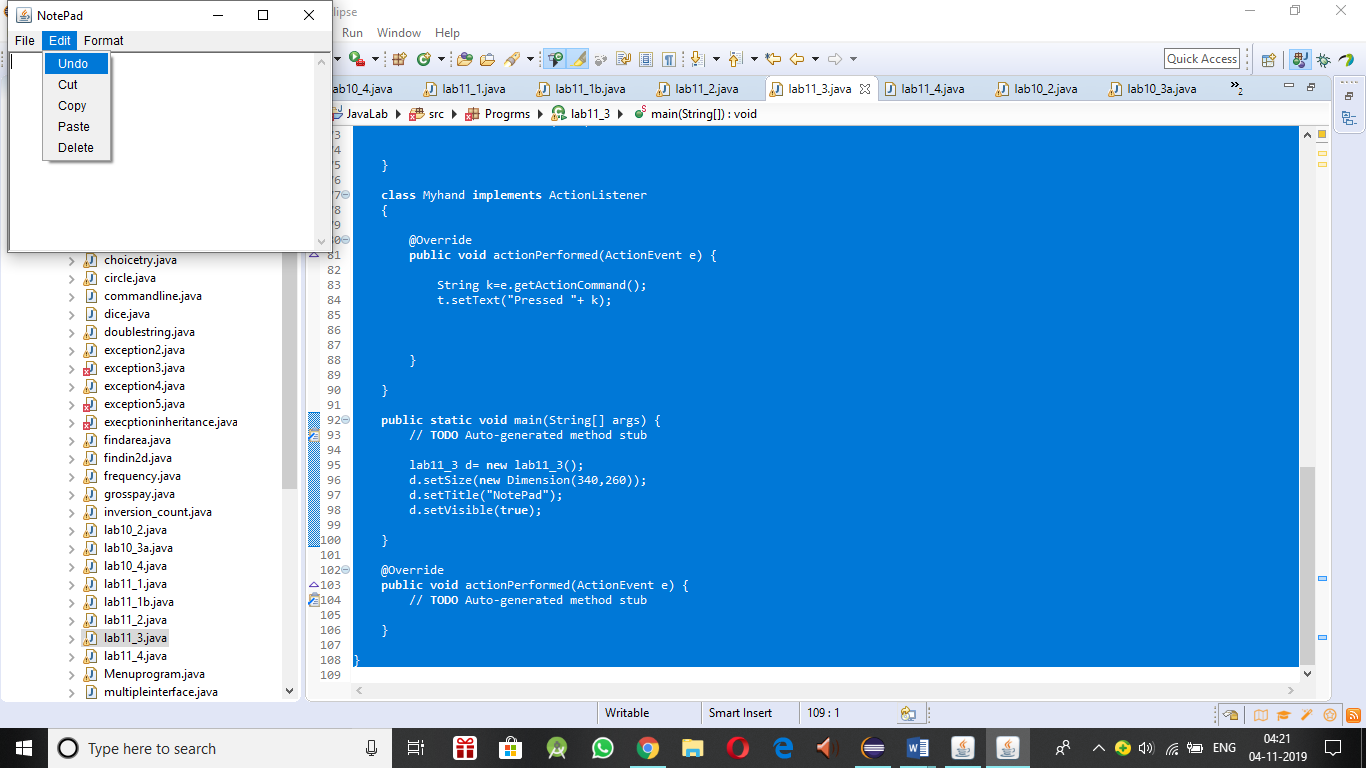
d.setSize(new Dimension(340,260));

d.setTitle("NotePad");

d.setVisible(true);

}

}



**4.FileDialog control represents a dialog window from which the user can select a file. Write a program to demonstrate simple FileDialog using Frame class.**

import java.awt.\*;

import java.awt.event.\*;

class SampleFrame extends Frame {

    SampleFrame(String title) {

        super(title);

            addWindowListener(new WindowAdapter() {

            public void windowClosing(WindowEvent we) {

                System.exit(0);}

        });

    }

}

class Lab8 {

    public static void main(String args[]) {

        // create a frame that owns the dialog

        Frame f = new SampleFrame("File Dialog Demo");

        f.setVisible(true);

        f.setSize(100, 100);

        FileDialog fd = new FileDialog(f, "File Dialog");

        fd.setVisible(true);

    }

}

