# Program No: 1 Aim: A simple program in Java

public class FirstProgram {

public static void main(String[] args) { System.out.println("Hello Java");

}

}

**Output**

Test Case - 1 User Output Hello Java

# Program No: 2 Aim: Write a Java program to Display Default values of all Primitive data types.

package q10815; class PrimitiveTypes{

static byte a; static short b; static int c; static long d; static boolean e; static double f; static float g;

public static void main(String[] args){ System.out.println("byte default value = "+ a); System.out.println("short default value = "+ b); System.out.println("int default value = "+ c); System.out.println("long default value = "+ d); System.out.println("boolean default value = "+ e); System.out.println("double default value = "+f); System.out.println("float default value = "+g);

}

}

**Output**

Test Case - 1 User Output

byte default value = 0 short default value = 0 int default value = 0 long default value = 0

boolean default value = false double default value = 0.0 float default value = 0.0

# Program No: 3 Aim: Program to Understand Command line arguments.

package q10817;

import java.util.Scanner; public class TotalAndAvgMarks {

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

String name =args[0] ;

float marks1 = Float.parseFloat(args[1]); float marks2 =Float.parseFloat(args[2]) ; float marks3 =Float.parseFloat(args[3]) ;

float total =marks1+marks2+marks3 ; float avg =total/3.0f ; System.out.println("Name = " + name); System.out.println("Marks1 = " +marks1 ); System.out.println("Marks2 = " +marks2 );

System.out.println("Marks3 = " +marks3 ); System.out.println("Total Marks = " +total ); System.out.println("Average Marks = " +avg );

}

}

**Output**

Test Case - 1 User Output Name = Narmada Marks1 = 78.5

Marks2 = 67.75

Marks3 = 71.2

Total Marks = 217.45 Average Marks = 72.48333

Test Case - 2 User Output Name = Nile Marks1 = 67.0

Marks2 = 81.0

Marks3 = 50.0 Total Marks = 198.0

## Average Marks = 66.0

**Program No: 4 Aim: Understanding if-then-else statement.**

package q10850; class BalanceCheck{

public void checkBalance(double b){

if(b<1000){

System.out.println("Balance is low");

}

else{

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

}

}

}

**Output**

Test Case - 1 User Output Balance is low

Test Case - 2 User Output Sufficient balance

Test Case - 3 User Output Sufficient balance

# Program No: 5 Aim: Java Program to find the Factorial of a given number.

package q10886;

public class Factorial{

public static void main(String[] args){ int b = Integer.parseInt(args[0]); int a=b;

int j=1; while (a>0){

j=j\*a; a--;

}

System.out.println("Factorial of "+b+" is "+j );

}

}

**Output**

Test Case - 1 User Output

Factorial of 5 is 120

Test Case - 2 User Output

Factorial of 0 is 1

# Program No: 6 Aim: Java Program to check whether the given number is Palindrome or not.

package q10894;

public class NumberPalindrome {

public void isNumberPalindrome(int number){ int dig,rev=0;

int a=number; while(number>0){

dig=number%10; rev=rev\*10+dig; number=number/10;

}

if (rev==a){

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

}

else{

System.out.println(a+" is not a palindrome");

}

}

}

**Output**

Test Case - 1 User Output

333 is a palindrome

Test Case - 2 User Output

567 is not a palindrome

# Program No: 7 Aim: Java Program to Display the Fibonacci Series

|  |  |  |  |
| --- | --- | --- | --- |
| package q10896;  class FibonacciSeries{  static void fibbo(int num){ System.out.print("0 1"); int a=0;  int b=1;  while (b<num && a<num){ a+=b;  b+=a;  if (a<num){  System.out.print(" "+a);  }  if (b<num){  System.out.print(" "+b);  }  }  }  public static void main(String[] args){ int arg=Integer.parseInt(args[0]);  fibbo(arg); | | | |
| }  } |  |  |  |
| **Output** |  |  |  |
| Test Case - User Output 0 1 1 2 3 5  Test Case - | 1  2 |  |  |
| User Output  0 1 1 2 3 5 | 8 13 | 21 34 | 55 |
| Test Case - User Output  0 1 1 2 3 5 | 3  8 13 | 21 34 | 55 |

**Program No: 8 Aim: Write a JAVA program to implement a class mechanism**

package q116;

import java.util.Scanner; class Main{

public static void main(String[] args){ Scanner input= new Scanner(System.in); System.out.print("Enter a string: "); String str=input.nextLine();

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

}

}

**Output**

Test Case - 1 User Output

Enter a string: Hello

The entered string is: Hello

Test Case - 2 User Output

Enter a string: Hello CodeTantra

The entered string is: Hello CodeTantra

# Program No: 9 Aim: Write a Java program to illustrate the abstract class concept.

package q11287; class Shape{

public void numberofSides(String a ,int x){ System.out.println("Number of sides in a "+a+" are "+x);

}

}

public class AbstractExample {

public static void main(String[] args) { Shape s= new Shape();; s.numberofSides("trapezoid",4); Shape t=new Shape(); t.numberofSides("triangle",3); Shape p =new Shape(); p.numberofSides("hexagon",6);

}

}

**Output**

Test Case - 1 User Output

Number of sides in a trapezoid are 4 Number of sides in a triangle are 3 Number of sides in a hexagon are 6

# Program No: 10 Aim: In a Java class, the fields which are marked as static are called static fields and those that are not marked as static are called as instance fields or simply fields.

package q11291;

import java.util.Scanner;

public class StaticFieldDemo{

public static void main(String [] args){ System.out.print("Enter a StaticField number"); Scanner input=new Scanner(System.in);

int a =input.nextInt(); int b =input.nextInt(); int c =input.nextInt();

System.out.println("a1 = A [instanceField = "+b+", aStaticField =

"+a+"]");

System.out.println("a2 = A [instanceField = "+c+", aStaticField =

"+a+"]");

System.out.println("A.aStaticField = "+a);

}

}

**Output**

Test Case – 1 User Output

Enter a StaticField number5 6 9

a1 = A [instanceField = 6, aStaticField = 5] a2 = A [instanceField = 9, aStaticField = 5] A.aStaticField = 5

Test Case - 2 User Output

Enter a StaticField number52 48 63 a1 = A [instanceField = 48, aStaticField = 52]

a2 = A [instanceField = 63, aStaticField = 52]

A.aStaticField = 52

# Program No: 11 Aim: Write the code.

package q11293;

import java.util.Scanner ; class StaticClassDemo{

public static void main(String[] args){ System.out.print("Enter a number:"); Scanner input=new Scanner(System.in);

int a =input.nextInt(); int b =input.nextInt();

System.out.println("a1 = A [value = "+a+"]"); System.out.println("a2 = A [value = "+b+"]");

}

}

**Output**

Test Case - 1 User Output

Enter a number:56 89 a1 = A [value = 56] a2 = A [value = 89]

Test Case - 2 User Output

Enter a number:9 4 a1 = A [value = 9] a2 = A [value = 4]

Test Case – 3 User Output

Enter a number:200 874 a1 = A [value = 200] a2 = A [value = 874]

# Program No: 12 Aim: Write a Java program to Access the Class members using super Keyword.

package q11274; class SuperClass {

int value1, value2;

// Write the code SuperClass(int a, int b){

value1 =a; value2 =b;

}

void show(){

System.out.println("This is super class show() method"); System.out.println("value1 = "+value1);

}

}

////SUB CLASS

class SubClass extends SuperClass { int value3, value4,value1,value2;

// Write the code

SubClass(int a,int b,int c,int d){ super(a,b);

//value1=a;

//value2=b; value3=c; value4=d;

}

public void show(){

System.out.println("This is sub class show() method");

//SuperClass obj1= new SuperClass(Integer.parseInt(args[0]),Integer.parseInt(args[1]));

super.show();

System.out.println("value2 from super class = "+super.value2); System.out.println("value3 = "+value3); System.out.println("value4 = "+value4);

}

}

public class AccessUsingSuper {

public static void main(String[] args) {

//SuperClass obj=new SuperClass(1,2);

SubClass obj = new SubClass(Integer.parseInt(args[0]), Integer.parseInt(args[1]), Integer.parseInt(args[2]), Integer.parseInt(args[3]));

obj.show();

}

}

**Output**

Test Case - 1 User Output

This is sub class show() method This is super class show() method value1 = 10

value2 from super class = 30 value3 = 40

value4 = 50

# Program No: 13 Aim: Program to implement Single Inheritance.

import java.util.Scanner; class FirstLevel{

Scanner input=new Scanner(System.in); void firstmethod(){

System.out.print("Enter the first class string: "); String var1=input.nextLine(); System.out.println("First class string is: "+var1);

}

}

class SecondLevel extends FirstLevel{ Scanner input=new Scanner(System.in); void secondmethod(){

super.firstmethod();

System.out.print("Enter the second class string: "); String var2=input.nextLine(); System.out.println("Second class string is: "+var2);

}

}

class SingleInheritance{

public static void main(String[] args){

//FirstLevel obj1=new FirstLevel(); SecondLevel obj2 =new SecondLevel();

//obj1.firstlevel(); obj2.secondmethod();

}

}

**Output**

Test Case - 1 User Output

Enter the first class string: Hello First class string is: HelloWorld! Enter the second class string: World! Second class string is: World!

Test Case - 2 User Output

Enter the first class string: Hey! Jack First class string is: Hey! JackHow are you?

Enter the second class string: How are you? Second class string is: How are you?

# Program No: 14 Aim: Write a Java program to implement Multilevel Inheritance.

package q11264; class Student{

int id; String name;

void setData(int id, String name){ this.id=id;

this.name=name;

}

void displayData(){

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

}

}

class Marks extends Student{

float javaMarks,cMarks,cppMarks;

void setMarks(float a,float b,float c){ javaMarks=a;

cMarks=b; cppMarks=c;

}

void displayMarks(){

System.out.println("Java marks : "+javaMarks); System.out.println("C marks : "+cMarks); System.out.println("Cpp marks : "+cppMarks);

}

}

class Result extends Marks{ float total,avg;

void compute(){

total=javaMarks+cMarks+cppMarks; avg=total/3;

}

void showResult(){

System.out.println("Total : "+total); System.out.println("Avg : "+avg);

}

}

public class MultilevelInheritanceDemo{ public static void main(String[] args){

int a=Integer.parseInt(args[0]); String b= args[1];

float c=Float.parseFloat(args[2]); float d=Float.parseFloat(args[3]); float e=Float.parseFloat(args[4]); Result obj=new Result(); obj.setData(a,b); obj.displayData(); obj.setMarks(c,d,e); obj.displayMarks(); obj.compute();

obj.showResult();

}

}

# Output

Test Case - 1 User Output Id : 99

Name : Geetha Java marks : 56.0 C marks : 75.5 Cpp marks : 66.6 Total : 198.1

Avg : 66.03333

Test Case - 2 User Output Id : 199

Name : Lakshmi Java marks : 55.5 C marks : 78.5 Cpp marks : 78.0 Total : 212.0

Avg : 70.666664

# Program No: 15 Aim: Write a Java program to implement Interface

package q11284; interface Car {

abstract String getName(); abstract int getMaxSpeed(); default void applyBreak(){

System.out.println("Applying break on "+getName());

}

public static String getFasterCar(Car obj1,Car obj2){ int s1=obj1.getMaxSpeed();

int s2=obj2.getMaxSpeed(); if (s1>s2){

return "Fastest car is : "+obj1.getName();

}else{

return "Fastest car is : "+obj2.getName();

}

}

}

class BMW implements Car { String name;

int MaxSpeed;

BMW(String name,int MaxSpeed){ this.name=name; this.MaxSpeed=MaxSpeed;

}

public String getName(){ return name;

}

public int getMaxSpeed(){ return MaxSpeed;

}

}

class Audi implements Car { String name;

int MaxSpeed;

Audi(String name,int MaxSpeed){ this.name=name; this.MaxSpeed=MaxSpeed;

}

public String getName(){ return name;

}

public int getMaxSpeed(){ return MaxSpeed;

}

}

public class MainApp {

public static void main(String args[]) { String a=args[0];

int b=Integer.parseInt(args[1]); String c=args[2];

int d=Integer.parseInt(args[3]); Car obj1=new BMW(a,b);

Car obj2=new Audi(c,d); System.out.println(Car.getFasterCar(obj1,obj2));

//System.out.println(obj2.getMaxSpeed());

}

}

# Output

Test Case - 1 User Output

Fastest car is : BMW

Test Case - 2 User Output

Fastest car is : Maruthi

# Program No: 16 Aim: Correct the error.

package q11161; public class Student {

private String id; private String name; private int age; private char gender;

public Student(String name, String rollNo, int age, char gender) { this.id = id;

this.name = name; this.age = age; this.gender = gender;

}

}

**Output**

Test Case - 1 User Output Good Job !

# Program No: 17 Aim: Program to implement method overriding and overloading.

//Overriding. class Vehicle{

void display(){

System.out.println("Vehicle is running");

}

}

class Bike1 extends Vehicle{ void display(){

System.out.println("Bike is running safely");

}

}

public class Bike2{

public static void main(String args[]){ Vehicle obj1=new Vehicle(); Vehicle obj2=new Bike1(); obj1.display();

obj2.display();

}

}

**Output**

Test Case - 1 User Output Vehicle is running

Bike is running safely

//Overloading

import java.util.Scanner; class Addition{

int a,b,c,sum;

void Add(int a,int b){ sum=a+b;

System.out.println("Addition of two numbers: "+sum);

}

void Add(int a,int b,int c){ sum=a+b+c;

System.out.println("Addition of three numbers: "+sum);

}

}

class TestOverloading1{

public static void main(String args[] ){ Scanner input=new Scanner(System.in); System.out.print("Enter three numbers: "); String s=input.nextLine();

String [] num=s.split(" ");

int n1=Integer.parseInt(num[0]); int n2=Integer.parseInt(num[1]); int n3=Integer.parseInt(num[2]); Addition obj=new Addition(); obj.Add(n1,n2);

obj.Add(n1,n2,n3);

}

}

# Output

Test Case - 1 User Output

Enter three numbers: 4 8 6 Addition of two numbers: 12 Addition of three numbers: 18

Test Case - 2 User Output

Enter three numbers: 101 301 501 Addition of two numbers: 402 Addition of three numbers: 903

# Program No: 18 Aim: Problem solving with Array.

package q11075;

import java.util.Scanner; public class CountOfTwoNumbers {

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

System.out.println("Enter no of elements in array:"); int n=input.nextInt();

System.out.println("Enter elements in the array seperated by

space:");

int[] nums=new int[n];

for(int i=0;i<nums.length;i++){ nums[i]=input.nextInt();

}

System.out.println("Enter the arg1 element:"); int arg1=input.nextInt(); System.out.println("Enter the arg2 element:"); int arg2=input.nextInt();

}

public boolean compareCountOf(int[] nums, int arg1, int arg2){ int i,carg1,carg2;

carg1=0; carg2=0;

for(i=0;i<nums.length;i++){ if (arg1==nums[i]){

carg1++;

}

else if (arg2==nums[i]){ carg2++;

}

}

if (arg1==arg2){

return false;

}

else if (carg1>carg2){ return true;

}

else{

return false;

}

}

}

**Output**

Test Case - 1 User Output

Enter no of elements in the array:6

Enter elements in the array seperated by space:1 2 2 3 5 2 Enter the arg1 element:2

Enter the arg2 element:5 True

Test Case - 2 User Output

Enter no of elements in the array:3

Enter elements in the array seperated by space:80 56 56 Enter the arg1 element:80

Enter the arg2 element:56 false

# Program No: 19 Aim: Write a program prints a multidimensional array of integers.

package q10946;

import java.util.Scanner; class MultiDimArrayPrinter{

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

System.out.print("Enter Number of rows: "); int row=sc.nextInt();

System.out.print("Enter Number of columns: "); int col=sc.nextInt();

int [][] mat=new int[row][col]; for (int i=0;i<row;i++){

System.out.print("Enter row "+(i+1)+": "); if (i==0){

sc.nextLine();

}

String[] var=sc.nextLine().split(" "); for (int j=0;j<col;j++){

mat[i][j]=Integer.parseInt(var[j]);

}

}

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

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

}

System.out.println( );

}

}

}

**Output**

Test Case - 1 User Output

Enter Number of rows: 3 Enter Number of columns: 3 Enter row 1: 1 2 3

Enter row 2: 4 5 6

Enter row 3: 7 8 9

1 2 3

4 5 6

7 8 9

Test Case - 2 User Output

Enter Number of rows: 3 Enter Number of columns: 4 Enter row 1: 4 5 6 1

Enter row 2: 9 4 2 5

Enter row 3: 76 3 7 69

4 5 6 1

9 4 2 5

76 3 7 69

# Program No: 20 Aim: Program to find Multiplication of Two matrices

package q11106;

import java.util.Scanner;

public class MultiplicationOfMatrix{ Scanner input=new Scanner(System.in); public int[][] readmatrix(){

System.out.print("Enter numbers of row: "); int row=input.nextInt(); System.out.print("Enter numbers of columns: "); int col=input.nextInt();

int[][] mat=new int[row][col];

System.out.println("Enter "+col+" numbers seperated by space"); for (int i=0;i<row;i++){

System.out.print("Enter row "+(i+1)+" "); if (i==0){

input.nextLine();

}

String[] var=input.nextLine().split(" "); for (int j=0;j<col;j++){

mat[i][j]=Integer.parseInt(var[j]);

}

}

return mat;

}

public int[][] multiplication(int[][] matrix1, int[][] matrix2) { int r1=matrix1.length;

int c1=matrix1[0].length; int r2=matrix2.length; int c2=matrix2[0].length; if (c1!=r2){

return null;

}

int[][] matrix=new int[r1][c2]; for (int i=0;i<r1;i++){

for (int j=0;j<c2;j++){ int sum=0;

for (int k=0;k<r2;k++){ sum+=matrix1[i][k]\*matrix2[k][j];

}

matrix[i][j]=sum;

}

}return matrix;

}

public static void main(String args[]){ MultiplicationOfMatrix obj=new MultiplicationOfMatrix(); System.out.println("Matrix 1:");

int [][] matrix1=obj.readmatrix(); System.out.println("Matrix 2: "); int [][] matrix2=obj.readmatrix();

System.out.println("Multiplication of the two given matrices is: "); int [][] result =obj.multiplication(matrix1,matrix2);

int row1=result.length; int col1=result[0].length; for (int i=0;i<row1;i++){

for (int j=0;j<col1;j++){ System.out.print(result[i][j]+" ");

}

System.out.println();

}

}

}

# Output

Test Case - 1 User Output Matrix 1:2

Enter number of rows: 2 Enter number of columns: 3

Enter 3 numbers separated by space1 2 3

Enter row 1: 1 2 3

Enter row 2: 4 5 6

Matrix 2:3

Enter number of rows: 3 Enter number of columns: 2

Enter 2 numbers separated by space1 2 Enter row 1: 1 2

Enter row 2: 3 4

Enter row 3: 5 6

Multiplication of the two given matrices is:

22 28

49 64

Test Case - 2 User Output Matrix 1:2

Enter number of rows: 2 Enter number of columns: 2

Enter 2 numbers separated by space1 2 Enter row 1: 1 2

Enter row 2: 3 4

Matrix 2:2

Enter number of rows: 2 Enter number of columns: 2

Enter 2 numbers separated by space5 6 Enter row 1: 5 6

Enter row 2: 7 8

Multiplication of the two given matrices is:

19 22

43 50

# Program No: 21 Aim: Program to Search an element using Linear Search.

package q11044; class LinearSearch{

public static void main(String args[]){ int[] array=new int[args.length]; int n= array.length;

for (int i=0;i<n;i++){ array[i]=Integer.parseInt(args[i]);

}

for (int i=0;i<n;i++){ System.out.println(array[i]);

}

LinearSearch obj=new LinearSearch(); int key=3; obj.linearSearch(array,key);

}

public void linearSearch(int[] array,int key){ int c=0;

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

c=c+1;

if (key==array[i]){

System.out.println("Search element "+key+" is found at

position : "+i);

break;

}

}

if (c==n){

System.out.println("Search element "+key+" is not found");

}

}

}

**Output**

Test Case - 1 User Output

Search element 20 is found at position : 1 Test Case - 2

User Output

Search element 21 is found at position : 2 Test Case - 3

User Output

Search element 9 is not found

# Program No: 22 Aim: Write a Java program to Search an element using Binary Search

package q11045;

import java.util.Arrays; public class BinarySearch{

public static void main(String args[]){ int[] array=new int[args.length]; for (int i=0;i<array.length;i++){

array[i]=Integer.parseInt(args[i]);

}

BinarySearch biSearch=new BinarySearch(); int key=4; biSearch.binarySearch(array,key);

}

public void binarySearch(int[] array,int key){ Arrays.sort(array);

int lb=0, ub=array.length-1; while (lb<=ub){

int mid=(lb+ub)/2; if (key==array[mid]){

System.out.println("Search element "+key+" is found at

position : "+(mid));

break;

}

else if(key<array[mid]){ ub=mid-1;

}

else if (key>array[mid]){ lb=mid+1;

}

}

if (lb>ub){

System.out.println("Search element "+key+" is not found");

}

}

}

**Output**

Test Case - 1 User Output

Search element 4 is found at position : 4

Test Case - 2 User Output

Search element 9 is not found

# Program No: 23 Aim: Program to Sort elements using Insertion Sort

package q11040;

import java.util.Arrays; class InsertionSorting{

public static void main(String args[]){ int[] array=new int[args.length]; for (int i=0;i<array.length;i++){

array[i]=Integer.parseInt(args[i]);

}

InsertionSorting inSorting=new InsertionSorting(); inSorting.insertionSort(array);

}

public void insertionSort(int[] array){ int l=array.length;

//Arrays.sort(array); for (int i=1;i<l;i++){

int temp=array[i]; int j=i;

while(j>0&&temp<array[j-1]){ array[j]=array[j-1];

j--;

}

array[j]=temp;

}

for (int i=0;i<l;i++){ System.out.println(array[i]);

}

}

}

**Output**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case - 1 User Output | Test Case - 2 User Output | Test User | Case - Output | 3 |
| 1 | 1 | 5 |  |  |
| 3 | 2 | 8 |  |  |
| 4 | 3 | 10 |  |  |
| 5 | 4 | 15 |  |  |
| 6 |  | 23 |  |  |

**Program No: 24 Aim: Program to Sort elements using Selection Sort - Largest element method.**

package q11041;

import java.util.Arrays;

class SelectionSortingLargestElement{

public static void main(String args[]){ int[] array=new int[args.length]; for (int i=0;i<array.length;i++){

array[i]=Integer.parseInt(args[i]);

}

SelectionSortingLargestElement sLarge =new SelectionSortingLargestElement();

sLarge.selectionSortLargestEle(array);

}

public void selectionSortLargestEle(int[] array){

//Arrays.sort(array); int l=array.length; for (int i=0;i<l;i++){

int temp,min=i;

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

if (array[j]<array[min]){ min=j;

}

}

temp=array[i]; array[i]=array[min]; array[min]=temp;

}

for (int i=0;i<array.length;i++){ System.out.println(array[i]);

}

}

}

**Output**

Test Case - 1 User Output 33

53

63

83

Test Case - 2 User Output 13

15

18

20

30

# Program No: 25 Aim: Write a Java program to Sort elements using Bubble Sort.

package q11039;

import java.util.Arrays; class BubbleSorting{

public static void main(String args[]){ int[] array=new int[args.length]; for (int i=0;i<array.length;i++){

array[i]=Integer.parseInt(args[i]);

}

//ubblting bSorting=new BubbleSorting(); BubbleSorting bSorting=new BubbleSorting(); bSorting.bubbleSort(array);

}

public void bubbleSort(int[] array){

//Arrays.sort(array); int l=array.length-1; for (int i=0;i<l;i++){

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

if (array[j]>array[j+1]){ int temp=array[j]; array[j]=array[j+1]; array[j+1]=temp;

}

}

}

for (int i=0;i<array.length;i++){ System.out.println(array[i]);

}

}

}

**Output**

Test Case - 1 User Output 10

20

30

50

Test Case - 2 User Output 9

17

18

19

20

21

# Program No: 26 Aim: Write a Java program to handle an ArithmeticException - divided by zero.

package q11329; class Division{

public static void main(String args[]){ int a=Integer.parseInt(args[0]); int b=Integer.parseInt(args[1]); try {

int c=a/b; System.out.println("Result = "+c);

}

catch(ArithmeticException e){

System.out.println("Exception caught : divide by zero

occurred");

}

}

}

**Output**

Test Case - 1 User Output Result = 4

Test Case - 2 User Output

Exception caught : divide by zero occurred

# Program No: 27 Aim: Program to implement User Defined Exception in Java.

class MyException extends Exception{ String str1; MyException(String str2){

str1=str2;

}

public String toString(){

return ("MyException Occurred: "+str1);

}

}

class Example1{

public static void main(String args[]){ try{

System.out.println("Starting of try block" ); throw new MyException("This is My error Message");

}

catch(MyException exp){ System.out.println("Catch Block"); System.out.println(exp);

}

}

}

**Output**

Test Case - 1 User Output

Starting of try block Catch Block

MyException Occurred: This is My error Message

# Program No: 28 Aim: Write a Java program to illustrate Finally block.

package q11330;

public class MyFinallyBlock {

public static void main(String args[]){

|  |  |  |  |
| --- | --- | --- | --- |
| int a=Integer.parseInt(args[0]); | | |  |
| int b=Integer.parseInt(args[1]); | | |
| float c=Float.parseFloat(args[2]); | | | |
| float d=Float.parseFloat(args[3]); | | | |
| try{ |  | | |
| int s=a/b; | |  | |

System.out.println("Result of integer values division : "+s);

|  |  |
| --- | --- |
| } |  |
| catch(ArithmeticException e){ | |

System.out.println("Inside the 1st catch block");

|  |  |  |
| --- | --- | --- |
|  | |  |
| } |  | |
| finally{ | | |

System.out.println("Inside the 1st finally block");

|  |  |  |
| --- | --- | --- |
| } |  | |
| try{ | |  |
| float s1=c/d; | | |

System.out.println("Result of float values division : "+s1);

}

catch(ArithmeticException e){

System.out.println("Inside the 2nd catch block");

}

finally{

System.out.println("Inside the 2nd finally block");

}

}

}

# Output

Test Case - 1 User Output

Result of integer values division : 2 Inside the 1st finally block

Result of float values division : 0.8333333 Inside the 2nd finally block

Test Case – 2

## User Output

Inside the 1st catch block Inside the 1st finally block

Result of float values division : 2.8666668

Inside the 2nd finally block

**Program No: 29 Aim: Write a Java program to illustrate Multiple catch blocks**.

package q11331;

import java.util.Scanner; public class MultiCatchBlocks {

void multiCatch(int arr[],int x){

|  |  |  |
| --- | --- | --- |
| //int[] array=arr;int x=x; | | |
| try{ |  | |
| int a=arr[x]; | |  |

System.out.println(a);

int b=a/x;

System.out.println(b);

|  |  |
| --- | --- |
| } |  |
| catch(ArithmeticException e){ | |

System.out.println("Division by zero exception occurred");

|  |  |
| --- | --- |
| } |  |
| catch(ArrayIndexOutOfBoundsException e){ | |

System.out.println("Array index out of bounds exception occurred");

|  |  |  |
| --- | --- | --- |
| } |  | |
|  | |  |
| catch(Exception e){ | | |

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

|  |  |  |  |
| --- | --- | --- | --- |
| } | |  | |
| } |  | | |
| public static void main(String[] args){ | | |  |
| Scanner input=new Scanner(System.in); | | | |

System.out.println("Enter no of elements in the array:");

|  |  |
| --- | --- |
| int n=input.nextInt(); | |
| int[] arr=new int[n]; |  |

System.out.println("Enter elements in the array seperated by space");

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

arr[i]=input.nextInt();

}

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

int x=input.nextInt();

MultiCatchBlocks obj=new MultiCatchBlocks();

|  |  |  |
| --- | --- | --- |
| obj.multiCatch(arr,x); | | |
|  | |  |
| } |  | |

}

# Output

Test Case - 1 User Output

Enter no of elements in the array:3

Enter elements in the array seperated by space:1 2 3 Enter the index element:3

Array index out of bounds exception occurred

Test Case - 2 User Output

Enter no of elements in the array:3

Enter elements in the array seperated by space:6 5 4 Enter the index element:1

5

5

Test Case - 3 User Output

Enter no of elements in the array:3

Enter elements in the array seperated by space:6 8 4 Enter the index element:0

6

Division by zero exception occurred

# Program No: 30 Aim: Write a Java program for creation of illustrating throw.

package q11335;

public class ThrowExample {

public static void main(String args[]) {

System.out.println("Welcome to the Registration process!!");

|  |  |
| --- | --- |
| try { |  |
| int a=Integer.parseInt(args[0]); | |
| int b=Integer.parseInt(args[1]); | |

checkEligibilty(a,b); // Fill the missing code

System.out.println("Have a nice day");

}

catch(ArithmeticException e ) { // Fill the missing code

System.out.println(e); // Fill the missing code

}

}

static void checkEligibilty(int age, int weight) {

if((age<12)&(weight<40)){ // Write the condition

throw new ArithmeticException("Student is not eligible for registration"); // Fill the missing code

}

else {

System.out.println("Student Entry is Valid!!");

}

}

}

# Output

Test Case - 1 User Output

Welcome to the Registration process!!

java.lang.ArithmeticException: Student is not eligible for registration

Test Case - 2 User Output

Welcome to the Registration process!! Student Entry is Valid!!

Have a nice day

# Program No: 31 Aim: Implement the concept of Assertions in JAVA programming language

package q122;

import java.util.Scanner; class AssertionExample{

|  |  |  |  |
| --- | --- | --- | --- |
| public static void main(String[] args){ | | |  |
| Scanner input=new Scanner(System.in); | | | |
| System.out.print("Enter your age: "); | | | |
| int value=input.nextInt(); |  | | |
| assert value>=18:"Not valid"; | |  | |

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

}

}

# Output

Test Case - 1 User Output

Enter your age: 23 value is 23

Test Case - 2 User Output Enter your age: 5 value is 5

# Program No: 32 Aim: Write a java program to implement the concept of localization.

package q123; import java.util.\*;

import java.util.Scanner; public class LocaleExample{

public static void main (String[] args){ Scanner myObj=new Scanner(System.in); System.out.println("Enter name"); String name =myObj.nextLine();

Locale locale = new Locale(name,name); System.out.println(locale.getDisplayCountry()); System.out.println(locale.getDisplayLanguage()); System.out.println(locale.getDisplayName()); System.out.println(locale.getISO3Country()); System.out.println(locale.getISO3Language()); System.out.println(locale.getLanguage()); System.out.println(locale.getCountry());

}

}

**Output**

Test Case - 1 User Output Enter nameit Italy Italian

Italian (Italy) ITA

ita it IT

Test Case - 2 User Output Enter namefr France French

French (France) FRA

fra fr FR

# Program No: 33 Aim : Appending all the small letters.

package q24212; class Example{

public static void main(String[] args){

String isUp=""; int z=args[0].length(); for (int y=0;y<z;y++){

if(Character.isUpperCase(args[0].charAt(y))){ char w=args[0].charAt(y);

isUp=isUp+w;

}

}

System.out.println("The result is: "+isUp);

}

}

**Output**

Test Case - 1 User Output

The result is: HYB Test Case – 2

User Output

The result is: CT

# Program No: 34 Aim: To show the usage of String Builder.

package q24216;

public class StringBuilderDemo {

public static void main(String args[]) { String s = new String("AB");

System.out.print("In Strings before concatenation Hash Code is: "); System.out.println(s.hashCode());

s += "C";

// print hash code after concatenating

System.out.println("In Strings after concatenation Hash Code: "+s.hashCode());

StringBuilder sb = new StringBuilder("AB");

// print hash code before concatenating

System.out.println("In StringBuilder before concatenation Hash Code is: "+sb.hashCode());

// add string C to AB

StringBuilder c=new StringBuilder("C"); sb=sb.append(c);

// print hash code after concatenating

System.out.println("In StringBuilder after concatenation Hash Code is: "+sb.hashCode());

// and observe the output

}

}

**Output**

Test Case - 1 User Output

In Strings before concatenation Hash Code is: 2081 In Strings after concatenation Hash Code is: 64578

In StringBuilder before concatenation Hash Code is: 1338823963 In StringBuilder after concatenation Hash Code is: 1338823963

# Program No: 35 Aim: To show the usage of String Buffer.

package q24215;

public class StringbufferExample {

public static void main (String args[]) { StringBuffer s=new StringBuffer();

System.out.println("Initial capacity is: "+s.capacity()); s=new StringBuffer(args[0]); System.out.println("Capacity after passing parameter is:

"+s.capacity());

StringBuffer s1=new StringBuffer(50);

System.out.println("Creating a StringBuffer object with the given capacity: "+s1.capacity());

// create instance of StringBuffer

// find the initial capacity

//find the capactiy after passing a parameter args[0] using command line

argument

// find the capatity by intializing capatity to 50

}

}

**Output**

Test Case - 1 User Output

Initial capacity is: 16

Capacity after passing parameter is: 27

Creating a StringBuffer object with the given capacity: 50

Test Case - 2 User Output

Initial capacity is: 16

Capacity after passing parameter is: 28

Creating a StringBuffer object with the given capacity: 50

# Program No: 36 Aim: Java program to implement even and odd threads by using Thread class and Runnable interface

package q124;

import java.util.Scanner; public class OddEvenPrintMain {

boolean odd; int count = 1; static int MAX;

//int MAX = 20;

public void printOdd() { synchronized (this) {

while (count < MAX) { System.out.println("Checking odd loop"); while (!odd) {

try {

System.out.println("Odd waiting : " + count); wait();

System.out.println("Notified odd :" + count);

} catch (InterruptedException e) {

// TODO Auto-generated catch block e.printStackTrace();

}

}

System.out.println("Odd Thread :" + count); count++;

odd = false; notify();

}

}

}

public void printEven() { try {

Thread.sleep(20);

} catch (InterruptedException e1) { e1.printStackTrace();

}

synchronized (this) {

while (count < MAX) { System.out.println("Checking even loop"); while (odd) {

try {

count); count);

System.out.println("Even waiting: " +

wait();

System.out.println("Notified even:" +

} catch (InterruptedException e) { e.printStackTrace();

}

}

System.out.println("Even thread :" + count); count++;

odd = true; notify();

}

}

}

public static void main(String[] args) { Scanner sc = new Scanner(System.in); System.out.print("Enter MAX value: "); MAX = sc.nextInt();

OddEvenPrintMain oep = new OddEvenPrintMain(); oep.odd = true;

Thread t1 = new Thread(new Runnable() { @Override

public void run() {

oep.printEven();

}

});

Thread t2 = new Thread(new Runnable() { @Override

public void run() {

oep.printOdd();

}

});

t1.start();

t2.start(); try {

t1.join();

t2.join();

} catch (InterruptedException e) { e.printStackTrace();

}

}

}

# Output

Test Case - 1 User Output

Enter MAX value: 10 Checking odd loop Odd Thread :1 Checking odd loop Odd waiting : 2 Checking even loop Even thread :2 Checking even loop Even waiting: 3

Notified odd :3

Odd Thread :3 Checking odd loop Odd waiting : 4 Notified even:4 Even thread :4 Checking even loop Even waiting: 5

Notified odd :5

Odd Thread :5 Checking odd loop Odd waiting : 6 Notified even:6 Even thread :6 Checking even loop Even waiting: 7

Notified odd :7

Odd Thread :7 Checking odd loop Odd waiting : 8 Notified even:8 Even thread :8 Checking even loop Even waiting: 9

Notified odd :9

Odd Thread :9 Notified even:10 Even thread :10

# Program No: 37 Aim: JAVA program to synchronize the threads by using Synchronize statements and Synchronize block

package q125;

import java.util.Scanner; class Table{

void printTable(){

synchronized(this){

System.out.println(" Current

Thread:"+Thread.currentThread().getName()+" ");

System.out.print("enter number to print its table:"); Scanner s=new Scanner(System.in);

int n=s.nextInt(); for(int i=1;i<=5;i++){

System.out.println(n\*i); try{

Thread.sleep(100);

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

}

}

}

}

class MyThread1 extends Thread{ Table t;

MyThread1(Table t){

this.t=t;

}

public void run(){

t.printTable();

}

}

class MyThread2 extends Thread{ Table t;

MyThread2(Table t){

this.t=t;

}

public void run(){

t.printTable();

}

}

class TestSynchronizedBlock1{

public static void main(String[] args){ Table obj=new Table();

MyThread1 t1=new MyThread1(obj);

MyThread2 t2=new MyThread2(obj); t1.start();

t2.start();

}

}

**Output**

Test Case - 1 User Output

-----Current Thread:Thread-0 5

enter number to print its table:5 57

107

157

207

257

-----Current Thread:Thread-1 7

enter number to print its table:7 7

14

21

28

35

Test Case - 2 User Output

-----Current Thread:Thread-0 8

enter number to print its table:8 823

1623

2423

3223

4023

-----Current Thread:Thread-1 23

enter number to print its table:23 23

46

69

92

115

# Program No: 38 Aim: Demonstrate the concept of type annotations in JAVA programming language.

package q128;

import java.util.Scanner; import java.lang.annotation.\*;

@Target(ElementType.TYPE\_USE)@interface TypeAnnoDemo{} public class MyClass{

public static void main(String[] args){ Scanner sc=new Scanner(System.in); System.out.print("Enter String : "); String x=sc.nextLine(); @TypeAnnoDemo String s=x; System.out.println(s);

myMethod();

}

static@TypeAnnoDemo int myMethod(){

System.out.println("There is a use of annotation with the return type of the function");

return 0;

}

}

**Output**

Test Case - 1 User Output

Enter String : hii hii

There is a use of annotation with the return type of the function

Test Case - 2 User Output

Enter String : hello! Good Morning hello! Good Morning

There is a use of annotation with the return type of the function

# Program No: 39 Aim: Demonstrate the concept of user-defined annotations in the JAVA programming language

import java.lang.annotation.\*; import java.lang.reflect.\*; @Retention(RetentionPolicy.RUNTIME) @Target(ElementType.METHOD) @interface MyAnnotation{

int value();

}

class Hello{ @MyAnnotation(value=10)

public void sayHello(){System.out.println("hello annotation");}

}

//Create class and access the defined annotation class TestCustomAnnotation1{

public static void main(String[] args)throws Exception{ Hello h=new Hello();

Method m=h.getClass().getMethod("sayHello"); MyAnnotation manno=m.getAnnotation(MyAnnotation.class);

System.out.println("value is: "+manno.value());

}

}

**Output**

Test Case - 1 User Output value is: 10

# Program No: 40 Aim: Write a JAVA program to implement the concept of Generic and collection classes.

package q132;

import java.util.Scanner; import java.util.ArrayList; import java.util.Collections; import java.util.List;

class AddingElements{

public static void main(String[] args){ Scanner sc=new Scanner(System.in); List<String>items =new ArrayList<>();

System.out.print("Enter any three collections: "); String a=sc.nextLine();

String b=sc.nextLine(); String c=sc.nextLine();

Collections.addAll(items,a,b,c); for(int i=0;i<items.size();i++){

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

}

}

}

**Output**

Test Case - 1 User Output

Enter any three collections: apple ball

cat

apple ball cat