OBJECT ORIENTED PROGRAMMING USING JAVA LAB

Program 1- Write a program to find the average and sum of the N numbers using Command line argument.

**INPUT-**

public class Sum Average {

public static void main (String [] args) {

if (args. length == 0) {

System.out.println("15");

return;

}

double sum = 0;

for (String arg: args) {

try {

double number = Double.parseDouble(arg);

sum += number;

} catch (NumberFormatException e) {

System.out.println("Invalid number: " + arg);

return;

}

}

double average = sum / args. length;

System.out.println("Sum: " + sum);

System.out.println("Average: " + average);

}

}

**OUTPUT:**

**A screen shot of a computer

AI-generated content may be incorrect.**

Program-2 Write a program to demonstrate type casting.

**INPUT-**

public class TypeCastingDemo {

public static void main (String [] args) {

int intVal = 100;

double doubleVal = intVal;

System.out.println("Implicit Type Casting:");

System.out.println("int value: " + intVal);

System.out.println("Converted to double: " + doubleVal);

System.out.println();

double originalDouble = 55.75;

int convertedInt = (int) originalDouble;

System.out.println("Explicit Type Casting:");

System.out.println("double value: " + originalDouble);

System.out.println("Converted to int: " + convertedInt);

}

}

**OUTPUT:A screenshot of a computer

AI-generated content may be incorrect.**

Program 3- Write a program to generate prime numbers between 1 & given numbers.

**INPUT-**

import java. util. Scanner;

public class Prime Generator {

public static void main (String [] args) {

Scanner scanner = new Scanner (System.in);

System.out.print("Enter the upper limit to generate prime numbers: ");

int limit = scanner. nextInt ();

System.out.println("Prime numbers between 1 and " + limit + " are:");

for (int num = 2; num <= limit; num++) {

if (isPrime(num)) {

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

}

}

scanner. Close ();

}

public static boolean isPrime (int number) {

if (number <= 1)

return false;

for (int i = 2; i <= Math.sqrt(number); i++) {

if (number % i == 0)

return false;

}

return true;

}

}

**OUTPUT:**

**A screenshot of a computer program

AI-generated content may be incorrect.**

Program 4-Write a program to generate pyramid of stars using nested for loops.

**INPUT-**

import java. util. Scanner;

public class Star Pyramid {

public static void main (String [] args) {

Scanner scanner = new Scanner (System.in);

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

int rows = scanner. nextInt ();

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

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

System.out.print(" ");

}

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

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

}

System.out.println();

}

scanner. Close ();

}

}

**OUTPUT:**

**A screenshot of a computer

AI-generated content may be incorrect.**

Program 5- Write a program to reversed pyramid using for loops and decrement operator.

**INPUT-**

import java. util. Scanner;

public class ReversedPyramid {

public static void main (String [] args) {

Scanner scanner = new Scanner (System.in);

System.out.print("Enter the number of rows for the reversed pyramid: ");

int rows = scanner.nextInt();

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

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

System.out.print(" ");

}

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

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

}

System.out.println();

}

scanner. Close();

}

}

**OUTPUT:**

**A screenshot of a computer

AI-generated content may be incorrect.**

Program 6- Write a program to demonstrate Nested Switch.

**INPUT-**

import java. util. Scanner;

public class Nested Switch Example {

public static void main (String [] args) {

Scanner scanner = new Scanner (System.in);

System.out.println("Main Menu:");

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

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

System.out.print("Enter your choice (1 or 2): ");

int main Choice = scanner. nextInt ();

switch (main Choice) {

case 1:

System.out.println("Fruits Menu:");

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

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

System.out.print("Enter your choice: ");

int fruit Choice = scanner. nextInt ();

switch (fruit Choice) {

case 1:

System.out.println("You selected Apple.");

break;

case 2:

System.out.println("You selected Banana.");

break;

default:

System.out.println("Invalid fruit choice.");

}

break;

case 2:

System.out.println("Vehicles Menu:");

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

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

System.out.print("Enter your choice: ");

int vehicle Choice = scanner. nextInt();

switch (vehicle Choice) {

case 1:

System.out.println("You selected Car.");

break;

case 2:

System.out.println("You selected Bike.");

break;

default:

System.out.println("Invalid vehicle choice.");

}

break;

default:

System.out.println("Invalid main menu choice.");

}

scanner. Close ();

}

}

**OUTPUT:**

A screen shot of a computer

AI-generated content may be incorrect.

Program 7-Write a program to calculate area of a circle using radius.

**INPUT-**

import java. util.Scanner;

public class Circle Area {

public static void main (String[] args) {

Scanner scanner = new Scanner (System.in);

// Ask the user to enter the radius

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

double radius = scanner. next Double();

// Calculate area using the formula: Area = π \* r^2

double area = Math.PI \* radius \* radius;

// Display the result

System.out.println("The area of the circle with radius " + radius + " is: " + area);

scanner. Close();

}

}

**OUTPUT:**

A screenshot of a computer

AI-generated content may be incorrect.

Program 8-Write a program to find G.C.D of the number.

**INPUT-**

import java.util.Scanner;

public class GCDCalculator {

public static void main (String[] args) {

Scanner = new Scanner (System.in);

System.out.print("Enter the first number: ");

int num1 = scanner. nextInt ();

System.out.print("Enter the second number: ");

int num2 = scanner. nextInt();

int gcd = findGCD (num1, num2);

System.out.println("GCD of " + num1 + " and " + num2 + " is: " + gcd);

scanner. Close ();

}

public static int findGCD (int a, int b) {

while (b != 0) {

int temp = b;

b = a % b;

a = temp;

}

return a;

}

}

**OUTPUT:**

**A screen shot of a computer

AI-generated content may be incorrect.**

Program 9- Write a program to create a simple class to find out the area and perimeter of the rectangle using super and this keyboard.

**INPUT-**

// Base class

class Shape {

protected double length;

protected double width;

// Constructor of Shape class

public Shape(double length, double width) {

this. Length = length; // Using 'this' to refer to instance variables

this. Width = width;

}

// Method to calculate area (to be overridden by Rectangle class)

public double calculate Area() {

return length \* width;

}

// Method to calculate perimeter (to be overridden by Rectangle class)

public double calculate Perimeter() {

return 2 \* (length + width);

}

}

// Derived class

class Rectangle extends Shape {

// Constructor of Rectangle class

public Rectangle(double length, double width) {

super(length, width); // Using 'super' to call the parent class constructor

}

@Override

public double calculate Area() {

return super.calculateArea(); // Using 'super' to call the base class method

}

@Override

public double calculate Perimeter() {

return super. calculatePerimeter(); // Using 'super' to call the base class method

}

// Display the area and perimeter

public void display() {

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

System.out.println("Perimeter of Rectangle: " + calculate Perimeter());

}

}

public class Main {

public static void main (String[] args) {

// Create a Rectangle object

Rectangle= new Rectangle (5, 7); // Length = 5, Width = 7

// Display the area and perimeter

rectangle. Display ();

}

}

**OUTPUT:**

**A screenshot of a computer

AI-generated content may be incorrect.**

Program 10- Write a program to find the factorial of a given number using recursion.

**INPUT-**

import java.util.Scanner;

public class FactorialCalculator {

public static int factorial(int n) {

if (n == 0 || n == 1) {

return 1;

}

return n \* factorial (n - 1);

}

public static void main (String [] args) {

Scanner = new Scanner (System.in);

System.out.print("Enter a number to find its factorial: ");

int number = scanner. nextInt();

int result = factorial(number);

System.out.println("Factorial of " + number + " is: " + result);

scanner. Close();

}

}

**OUTPUT:**

**A screenshot of a computer

AI-generated content may be incorrect.**

Program 11- Write a program to design a class using abstract methods and abstract classes.

**INPUT-**

abstract class Shape {

// Abstract method (no body)

abstract double calculate Area();

// Non-abstract method

public void display Shape() {

System.out.println("Calculating area of a shape...");

}

}

// Concrete subclass

class Rectangle extends Shape {

private double length;

private double width;

// Constructor

public Rectangle(double length, double width) {

this. Length = length;

this. Width = width;

}

// Implement the abstract method

@Override

double calculate Area() {

return length \* width;

}

}

// Another concrete subclass

class Circle extends Shape {

private double radius;

// Constructor

public Circle(double radius) {

this. Radius = radius;

}

// Implement the abstract method

@Override

double calculate Area() {

return Math.PI \* radius \* radius;

}

}

// Main class

public class Abstract Demo {

public static void main(String[] args) {

// Create objects of Rectangle and Circle

Shape rect = new Rectangle(5, 4);

Shape circle = new Circle(3);

// Display shape details and area

rect. display Shape();

System.out.println("Rectangle Area: " + rect. calculate Area());

circle. display Shape();

System.out.println("Circle Area: " + circle. calculate Area());

}

}

**OUTPUT:**

A screenshot of a computer

AI-generated content may be incorrect.

Program 12- Write a program to count the number of object created for a class using static member function.

**INPUT-**

public class ObjectCounter {

// Static variable to count objects

private static int count = 0;

// Constructor

public ObjectCounter() {

count++;

}

// Static method to get object count

public static int getObjectCount() {

return count;

}

// Main method

public static void main (String[] args) {

// Create some objects

ObjectCounter obj1 = new ObjectCounter ();

ObjectCounter obj2 = new ObjectCounter ();

ObjectCounter obj3 = new ObjectCounter ();

// Display count

System.out.println("Total objects created: " + ObjectCounter.getObjectCount());

}

}

**OUTPUT:**

**A black rectangular object with a dark background

AI-generated content may be incorrect.**

Program 13- Write a program to demonstrate the use of function overloading.

**INPUT-**

public class FunctionOverloading {

public void show() {

System.out.println("Show method with no parameters.");

}

public void show(int a) {

System.out.println("Show method with one integer: " + a);

}

public void show(String message) {

System.out.println("Show method with one string: " + message);

}

public void show(int a, String message) {

System.out.println("Show method with int and string: " + a + ", " + message);

}

public static void main(String[] args) {

Function Overloading obj = new Function Overloading();

obj. show();

obj. show(10);

obj. show("Hello");

obj.show(20, "Java");

}

}

**OUTPUT:**

**A screenshot of a computer

AI-generated content may be incorrect.**

Program14- Write a program to demonstrate the use of inheritance.

**INPUT-**

// Superclass (Parent class)

class Animal {

// Field in the parent class

String name;

// Constructor of the parent class

public Animal(String name) {

this.name = name;

}

// Method in the parent class

public void eat() {

System.out.println(name + " is eating.");

}

public void sleep() {

System.out.println(name + " is sleeping.");

}

}

// Subclass (Child class) inheriting from Animal

class Dog extends Animal {

// Constructor of the child class

public Dog(String name) {

super(name); // Call the parent class constructor

}

// Method in the child class public void bark() {

System.out.println(name + " is barking.");

}

}

// Main class to test inheritance

public class Inheritance Demo {

public static void main (String[] args) {

// Creating an object of Dog class

Dog = new Dog("Buddy");

// Calling methods inherited from Animal class

dog. Eat(); // Inherited method

dog.sleep(); // Inherited method

// Calling method defined in Dog class

dog.bark(); // Child class specific method

}

}

**OUTPUT:**

A black rectangular object with a black stripe

AI-generated content may be incorrect.

Program 15- Write a program that shows the partial implementation of Interface.

**INPUT**-

interface Animal {

void sound();

default void sleep() {

System.out.println("The animal is sleeping.");

}

default void eat() {

System.out.println("The animal is eating.");

}

}

class Dog implements Animal {

@Override

public void sound() {

System.out.println("The dog barks.");

}

}

class Cat implements Animal {

@Override

public void sound() {

System.out.println("The cat meows.");

}

@Override

public void sleep() {

System.out.println("The cat is sleeping peacefully.");

}

}

public class InterfacePartialImplementationDemo {

public static void main(String[] args) {

Animal dog = new Dog();

Animal cat = new Cat();

dog.sound();

cat.sound();

dog.sleep();

cat.sleep();

dog.eat();

cat.eat();

}

}

**OUTPUT**:

A screenshot of a computer

AI-generated content may be incorrect.

Program 16- Write a program to design a string class that perform string method (Equal, Reverse the String Change case).

**INPUT:**

class MyString {

private String str;

public My String(String str) {

this.str = str;

}

public boolean equals(String other) {

return St. Equals(other);

}

public String reverse() {

StringBuilder reversed = new StringBuilder(str);

return reversed. Reverse().to String();

}

public String change Case() {

StringBuilder changed = new StringBuilder();

for (char c : str.toCharArray()) {

if (Character.isUpperCase(c)) {

changed. Append(Character.toLowerCase(c));

} else if (Character.isLowerCase(c)) {

changed. Append(Character.toUpperCase(c));

} else {

changed. Append(c);

}

}

return changed.toString();

}

public String getStr() {

return str;

}

}

public class String Method {

public static void main(String[] args) {

My String myStr = new My String("Hello World");

System.out.println("Is equal to 'Hello World'? " + myStr.equals("Hello World"));

System.out.println("Is equal to 'hello world'? " + myStr.equals("hello world"));

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

System.out.println("Changed Case: " + myStr.changeCase());

}

}

Program 17- Write a program to handle the exception using try and multiple catch block.

**INPUT-**

public class ExceptionHandlingExample {

public static void main(String[] args) {

try {

// Code that may cause exceptions

int [] numbers = {1, 2, 3};

System.out.println("Accessing 4th element: " + numbers[3]); // ArrayIndexOutOfBoundsException

int result = 10 / 0; // ArithmeticException

String str = null;

System.out.println(str.length()); // NullPointerException

} catch (ArrayIndexOutOfBoundsException e) {

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

} catch (Arithmetic Exception e) {

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

} catch (NullPointerException e) {

System.out.println("Caught NullPointerException: " + e. get Message ());

} catch (Exception e) {

// This will catch any other exceptions

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

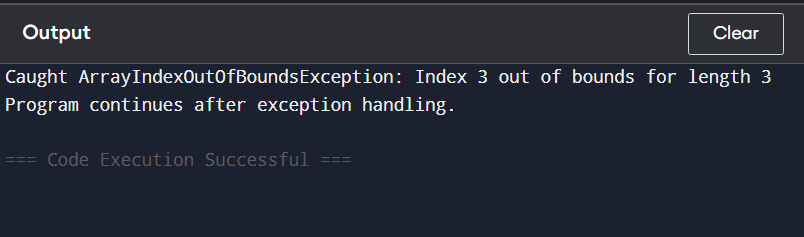
}

System.out.println("Program continues after exception handling.");

}

}

**OUTPUT :**



Program 18- Write a program that implement the Nested try statement.

**INPUT –**

import java.util.Scanner;

public class NestedTryExample {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

try {

// Outer try block

System.out.print("Enter a number to divide 100: ");

int number = scanner.nextInt();

int result = 100 / number;

System.out.println("Result: " + result);

try {

// Inner try block

int[] arr = new int[5];

System.out.print("Enter index to access (0 to 4): ");

int index = scanner.nextInt();

System.out.println("Element at index " + index + ": " + arr[index]);

} catch (ArrayIndexOutOfBoundsException e) {

System.out.println("Inner Catch: Array index out of bounds!");

}

} catch (ArithmeticException e) {

System.out.println("Outer Catch: Cannot divide by zero!");

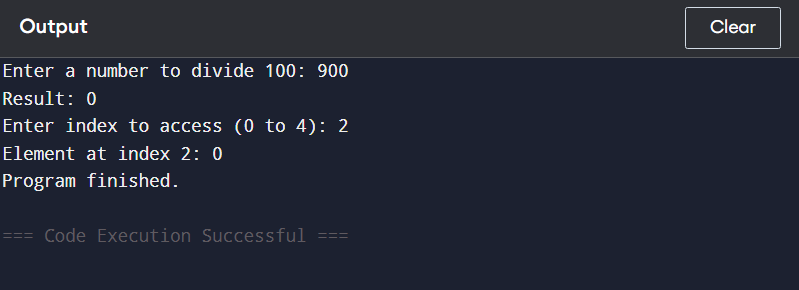
}

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

}

}

**OUTPUT:**



Program 19- Write a program to create a package that access the members of external class as well as same package.

**INPUT-**

package mypackage;

public class External Class {

public void show Message() {

System.out.println("Hello from External Class in 'mypackage'!");

}

}

import mypackage. External Class; // Importing external class

class SamePackageClass {

public void display () {

System.out.println("Hello from SamePackageClass in the same file/package!");

}

}

public class Main Program {

public static void main (String[] args) {

// Creating object of external class

External Class external = new External Class();

external. showMessage();

// Creating object of same package class

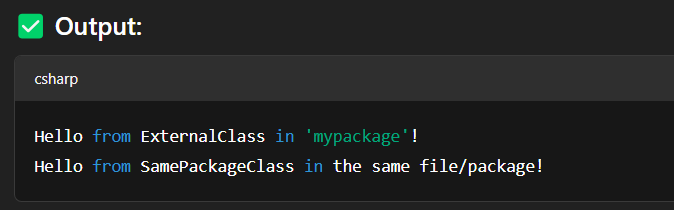
SamePackageClass local = new SamePackageClass ();

local. Display();

}

}

**OUTPUT:**



Program 20- Write a program that import the user define package ands access the member variable of classes that contained by package.

**INPUT-**

package user package;

public class Student {

// Member variables

public String name = "Alice";

public int age = 20;

public String course = "Computer Science";

}

import userpackage.Student; // Importing user-defined package

public class Test Student {

public static void main (String[] args) {

// Create object of Student class

Student = new Student ();

// Access and print member variables

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

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

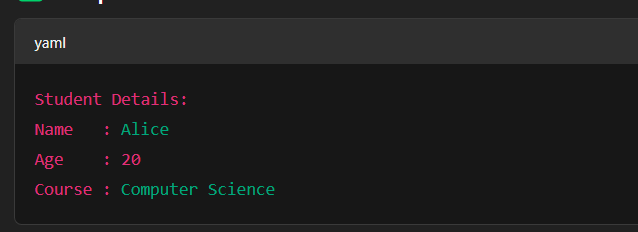
System.out.println("Age : " + student. Age);

System.out.println("Course: " + student. Course);

}

}

**OUTPUT:**

****

Program 21- Write a program to handle the user defined exception using throw keyboards.

**INPUT –**

// Step 1: Create a user-defined exception

class MyCustomException extends Exception {

public MyCustomException (String message) {

super(message);

}

}

public class Throw Example {

// Method that throws user-defined exception

static void check Number(int num) throws MyCustomException {

if (num < 0) {

// Step 2: Throw the exception using 'throw'

throw new MyCustomException ("Number is negative. Custom Exception triggered!");

} else {

System.out.println("Number is positive: " + num);

}

}

public static void main(String[] args) {

try {

// Step 3: Use try-catch to handle the exception

check Number(-10); // This will trigger the exception

} catch (MyCustomException e) {

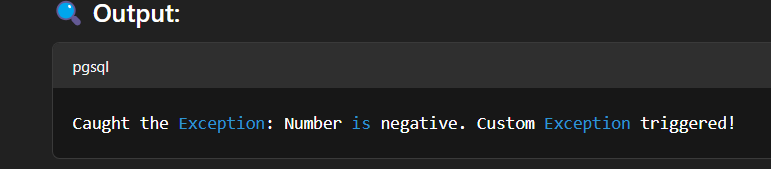
System.out.println("Caught the Exception: " + e. getMessage());

}

}

}

**OUTPUT:**

****

Program 22- Write a program to create a class component that shows controls ands event handling on that controls.

**INPUT** –

-