Program to count the letters, spaces, numbers, and other characters of an input string:

java

public class CharacterCounter {

public static void main(String[] args) {

String inputString = "Hello 123 World!";

int letters = 0, spaces = 0, numbers = 0, others = 0;

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

if (Character.isLetter(c)) {

letters++;

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

numbers++;

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

spaces++;

} else {

others++;

}

}

System.out.println("Letters: " + letters);

System.out.println("Spaces: " + spaces);

System.out.println("Numbers: " + numbers);

System.out.println("Other Characters: " + others);

}

}

Function to calculate the sum of digits for a given char array:

java

public class DigitSumCalculator {

public static void main(String[] args) {

char[] digits = {'1', '2', '3', '4', '5'};

long digitSum = calculateDigitSum(digits);

System.out.println("Digit Sum: " + digitSum);

}

public static long calculateDigitSum(char[] digits) {

long sum = 0;

for (char digit : digits) {

if (Character.isDigit(digit)) {

sum += digit - '0'; // Convert char to int

}

}

return sum;

}

}

Program to find the smallest and largest element from the array:

java

import java.util.Arrays;

public class MinMaxFinder {

public static void main(String[] args) {

int[] array = {5, 2, 7, 1, 9, 3};

int smallest = findSmallest(array);

int largest = findLargest(array);

System.out.println("Smallest element: " + smallest);

System.out.println("Largest element: " + largest);

}

public static int findSmallest(int[] array) {

Arrays.sort(array);

return array[0];

}

public static int findLargest(int[] array) {

Arrays.sort(array);

return array[array.length - 1];

}

}

Class SortData:

java

import java.util.Arrays;

public class SortData {

public void asec(int[] arr) {

Arrays.sort(arr);

System.out.println("Ascending order: " + Arrays.toString(arr));

}

public void desc(int[] arr) {

Arrays.sort(arr);

System.out.print("Descending order: ");

for (int i = arr.length - 1; i >= 0; i--) {

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

}

System.out.println();

}

}

Demonstration of Constructor and Destructor:

java

public class MyClass {

public MyClass() {

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

}

protected void finalize() {

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

}

public static void main(String[] args) {

MyClass obj = new MyClass();

obj = null; // Making the object eligible for garbage collection

System.gc(); // Request garbage collection

}

}

Bank Account Management System:

java

public class BankAccount {

protected String accountNumber;

protected double balance;

public BankAccount(String accountNumber) {

this.accountNumber = accountNumber;

this.balance = 0.0;

}

public void deposit(double amount) {

if (amount > 0) {

balance += amount;

System.out.println("Deposit successful. New balance: " + balance);

} else {

System.out.println("Invalid deposit amount.");

}

}

public void withdraw(double amount) {

if (balance >= amount && amount > 0) {

balance -= amount;

System.out.println("Withdrawal successful. New balance: " + balance);

} else {

System.out.println("Insufficient balance or invalid withdrawal amount.");

}

}

public void balanceInquiry() {

System.out.println("Current balance: " + balance);

}

}

class SavingsAccount extends BankAccount {

private double interestRate;

public SavingsAccount(String accountNumber, double interestRate) {

super(accountNumber);

this.interestRate = interestRate;

}

public void calculateInterest() {

double interest = balance \* (interestRate / 100);

System.out.println("Interest calculated: " + interest);

}

}

class CurrentAccount extends BankAccount {

private double overdraftLimit;

public CurrentAccount(String accountNumber, double overdraftLimit) {

super(accountNumber);

this.overdraftLimit = overdraftLimit;

}

@Override

public void withdraw(double amount) {

if (balance + overdraftLimit >= amount && amount > 0) {

balance -= amount;

System.out.println("Withdrawal successful. New balance: " + balance);

} else {

System.out.println("Insufficient balance or invalid withdrawal amount.");

}

}

}

class FixedDepositAccount extends BankAccount {

private int term;

private double interestRate;

public FixedDepositAccount(String accountNumber, int term, double interestRate) {

super(accountNumber);

this.term = term;

this.interestRate = interestRate;

}

public void closeAccount() {

double interest = balance \* (interestRate / 100) \* term;

balance += interest;

System.out.println("Account closed. Final balance: " + balance);

}

}

Hierarchy of Employees:

java

public class Employee {

protected String name;

protected int age;

protected double salary;

public Employee(String name, int age, double salary) {

this.name = name;

this.age = age;

this.salary = salary;

}

}

class Manager extends Employee {

protected String department;

public Manager(String name, int age, double salary, String department) {

super(name, age, salary);

this.department = department;

}

}

class Worker extends Manager {

protected String taskList;

public Worker(String name, int age, double salary, String department, String taskList) {

super(name, age, salary, department);

this.taskList = taskList;

}

}

**Shape Class Structure:**

java

public abstract class Shape {

public abstract double calculateArea();

public abstract double calculatePerimeter();

}

class Rectangle extends Shape {

private double length;

private double width;

public Rectangle(double length, double width) {

this.length = length;

this.width = width;

}

@Override

public double calculateArea() {

return length \* width;

}

@Override

public double calculatePerimeter() {

return 2 \* (length + width);

}

}

class Square extends Rectangle {

public Square(double side) {

super(side, side);

}

}

**Vehicle Class Hierarchy:**

java

public abstract class Vehicle {

public abstract void start();

public abstract void stop();

}

class Car extends Vehicle {

private String model;

private String color;

public Car(String model, String color) {

this.model = model;

this.color = color;

}

@Override

public void start() {

System.out.println("Car " + model + " started.");

}

@Override

public void stop() {

System.out.println("Car " + model + " stopped.");

}

}

class ElectricCar extends Car {

private double batteryCapacity;

public ElectricCar(String model, String color, double batteryCapacity) {

super(model, color);

this.batteryCapacity = batteryCapacity;

}

}

**Program to calculate and print the area of a circle**:

java

import java.util.Scanner;

public class CircleArea {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

double radius = scanner.nextDouble();

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

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

scanner.close();

}

}

**Program to calculate and print the perimeter of a rectangle:**

java

import java.util.Scanner;

public class RectanglePerimeter {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

double length = scanner.nextDouble();

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

double width = scanner.nextDouble();

double perimeter = 2 \* (length + width);

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

scanner.close();

}

}

**Program to calculate and print the area of a triangle:**

java

import java.util.Scanner;

public class TriangleArea {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

double base = scanner.nextDouble();

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

double height = scanner.nextDouble();

double area = 0.5 \* base \* height;

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

scanner.close();

}

}

**Program to find the area of a rectangle using an interface:**

java

interface FindArea {

double area();

}

class Rectangle implements FindArea {

private double length;

private double width;

public Rectangle(double length, double width) {

this.length = length;

this.width = width;

}

@Override

public double area() {

return length \* width;

}

}

public class RectangleArea {

public static void main(String[] args) {

double length = 5.0;

double width = 3.0;

Rectangle rectangle = new Rectangle(length, width);

double area = rectangle.area();

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

}

}