1. **WAP in Java to print Hello**

public class HelloWorld {

public static void main(String[] args) {

System.out.println("Hello, World!");

}

}

1. **WAP in Java to understand the difference between print() and println().**

public class PrintVsPrintln {

public static void main(String[] args) {

System.out.print("This is using print().");

System.out.print(" It doesn't add a newline.");

System.out.println("This is using println().");

System.out.println(" It adds a newline after printing.");

}

}

1. **WAP in Java with two classes create an object of a class and call into another class (having main method)**

Class 1:Person

class Person {

String name;

int age;

public Person(String name, int age) {

this.name = name;

this.age = age;

}

public void displayInfo() {

System.out.println("Name:

" + name);

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

}

}

Class 2: MainClass

public class MainClass {

public static void main(String[] args) {

Person person1 = new Person("Alice", 30);

person1.displayInfo();

}

}

1. **Write a Java program to print the sum (addition), multiply, subtract, divide and remainder of two numbers.**

import java.util.Scanner;

public class ArithmeticOperations {

public static void main(String[] args) {

// Create a Scanner object to read input

Scanner scanner = new Scanner(System.in);

// Prompt user for two numbers

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

int num1 = scanner.nextInt();

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

int num2 = scanner.nextInt();

// Perform arithmetic operations

int sum = num1 + num2;

int difference = num1 - num2;

int product = num1 \* num2;

int quotient = num1 / num2;

int remainder = num1 % num2;

// Print the results

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

System.out.println("Difference: " + difference);

System.out.println("Product: " + product);

System.out.println("Quotient: " + quotient);

System.out.println("Remainder: " + remainder);

// Close the scanner

scanner.close();

}

}

1. **WAP in Java to find product of two numbers. (Input by the user)**

import java.util.Scanner;

public class ProductOfTwoNumbers {

public static void main(String[] args) {

// Create a Scanner object to read input

Scanner scanner = new Scanner(System.in);

// Prompt user for the first number

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

double num1 = scanner.nextDouble();

// Prompt user for the second number

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

double num2 = scanner.nextDouble();

// Calculate the product of the two numbers

double product = num1 \* num2;

// Display the product

System.out.println("The product of " + num1 + " and " + num2 + " is: " + product);

}

}

1. **WAP in Java to illustrate the concept of local, instance and static variable**

public class VariableTypes {

// Instance variable

int instanceVariable = 10;

// Static variable

static int staticVariable = 20;

public void instanceMethod() {

// Local variable

int localVar = 30;

System.out.println("Instance variable: " + instanceVariable);

System.out.println("Static variable: " + staticVariable);

System.out.println("Local variable: " + localVar);

}

public static void staticMethod() {

// Local variable

int localVar = 40;

System.out.println("Static variable: " + staticVariable);

System.out.println("Local variable: " + localVar);

}

public static void main(String[] args) {

// Create an instance of the class

VariableTypes obj = new VariableTypes();

// Call instance and static methods

obj.instanceMethod();

staticMethod();

}

}

1. **WAP in Java to check whether the given number is prime or not.**

import java.util.Scanner;

public class PrimeCheck {

public static void main(String[] args) {

// Create a Scanner object for input

Scanner scanner = new Scanner(System.in);

// Ask the user for input

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

int number = scanner.nextInt();

// Check if the number is prime

boolean isPrime = checkPrime(number);

// Output the result

if (isPrime) {

System.out.println(number + " is a prime number.");

} else {

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

}

// Close the scanner

scanner.close();

}

// Method to check if a number is prime

public static boolean checkPrime(int num) {

// Handle special cases for numbers less than 2

if (num <= 1) {

return false;

}

// Loop to check if the number has any divisors other than 1 and itself

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

if (num % i == 0) {

return false;

}

}

// If no divisors were found, the number is prime

return true;

}

}

1. **WAP in Java to implement implicit and explicit type casting**

public class TypeCastingExample {

public static void main(String[] args) {

// Implicit Type Casting (Widening)

int intVal = 100;

double doubleVal = intVal; // int is automatically converted to double (widening)

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

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

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

// Explicit Type Casting (Narrowing)

double largeDoubleVal = 99.99;

int narrowedIntVal = (int) largeDoubleVal; // double is explicitly cast to int (narrowing)

System.out.println("\nExplicit Type Casting:");

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

System.out.println("Converted to int: " + narrowedIntVal); // decimal part is truncated

}

}

1. **WAP in Java to implement various operators in java**

public class OperatorsDemo {

public static void main(String[] args) {

// 1. Arithmetic Operators

int a = 10;

int b = 5;

System.out.println("Arithmetic Operators:");

System.out.println("Addition (a + b): " + (a + b));

System.out.println("Subtraction (a - b): " + (a - b));

System.out.println("Multiplication (a \* b): " + (a \* b));

System.out.println("Division (a / b): " + (a / b));

System.out.println("Modulus (a % b): " + (a % b));

// 2. Relational Operators

System.out.println("\nRelational Operators:");

System.out.println("a is equal to b: " + (a == b));

System.out.println("a is not equal to b: " + (a != b));

System.out.println("a is greater than b: " + (a > b));

System.out.println("a is less than b: " + (a < b));

System.out.println("a is greater than or equal to b: " + (a >= b));

System.out.println("a is less than or equal to b: " + (a <= b));

// 3. Logical Operators

boolean x = true;

boolean y = false;

System.out.println("\nLogical Operators:");

System.out.println("x AND y: " + (x && y));

System.out.println("x OR y: " + (x || y));

System.out.println("NOT x: " + (!x));

// 4. Bitwise Operators

int p = 6; // In binary: 110

int q = 4; // In binary: 100

System.out.println("\nBitwise Operators:");

System.out.println("p AND q (p & q): " + (p & q)); // Binary AND

System.out.println("p OR q (p | q): " + (p | q)); // Binary OR

System.out.println("p XOR q (p ^ q): " + (p ^ q)); // Binary XOR

System.out.println("Bitwise NOT (~p): " + (~p)); // Binary NOT

System.out.println("Left Shift (p << 1): " + (p << 1)); // Left shift

System.out.println("Right Shift (p >> 1): " + (p >> 1)); // Right shift

// 5. Assignment Operators

int c = 20;

System.out.println("\nAssignment Operators:");

System.out.println("Initial value of c: " + c);

c += 10; // Equivalent to c = c + 10;

System.out.println("After c += 10: " + c);

c -= 5; // Equivalent to c = c - 5;

System.out.println("After c -= 5: " + c);

c \*= 2; // Equivalent to c = c \* 2;

System.out.println("After c \*= 2: " + c);

c /= 5; // Equivalent to c = c / 5;

System.out.println("After c /= 5: " + c);

c %= 3; // Equivalent to c = c % 3;

System.out.println("After c %= 3: " + c);

}

}

1. **WAP in Java to implement the single dimension array**

import java.util.Scanner;

public class SingleDimensionArray {

public static void main(String[] args) {

// Create a Scanner object to take input from the user

Scanner scanner = new Scanner(System.in);

// Ask the user for the size of the array

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

int size = scanner.nextInt();

// Declare and initialize the array

int[] array = new int[size];

// Prompt the user to input values into the array

System.out.println("Enter " + size + " elements for the array:");

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

array[i] = scanner.nextInt();

}

// Display the elements of the array

System.out.println("The elements of the array are:");

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

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

}

// Close the scanner to avoid resource leak

scanner.close();

}

}

1. **WAP in Java to copy the elements from one array to another array**

public class ArrayCopy {

public static void main(String[] args) {

// Original array

int[] originalArray = {1, 2, 3, 4, 5};

// Create a new array of the same length

int[] newArray = new int[originalArray.length];

// Copying elements from originalArray to newArray

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

newArray[i] = originalArray[i];

}

// Display the new array

System.out.println("Elements in the new array:");

for (int element : newArray) {

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

}

}

}

1. **WAP in Java to perform the addition and multiplication in 2-D array**

import java.util.Scanner;

public class ArrayOperations {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the

number of rows: ");

int rows = scanner.nextInt();

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

int columns = scanner.nextInt();

int[][] array1

= new int[rows][columns];

int[][] array2 = new int[rows][columns];

System.out.println("Enter elements of array 1:");

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

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

array1[i][j] = scanner.nextInt();

}

}

System.out.println("Enter elements of array 2:");

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

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

array2[i][j] = scanner.nextInt();

}

}

int[][] sum = new int[rows][columns];

int[][] product = new int[rows][columns];

// Addition

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

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

sum[i][j] = array1[i][j] + array2[i][j];

}

}

// Multiplication

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

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

product[i][j] = 0;

for (int k = 0; k < columns; k++) {

product[i][j] += array1[i][k] \* array2[k][j];

}

}

}

System.out.println("Sum of the arrays:");

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

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

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

}

System.out.println();

}

System.out.println("Product of the arrays:");

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

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

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

}

System.out.println();

}

}

}

1. **WAP in Java to print the duplicate elements of an array**.

import java.util.HashSet;

public class DuplicateElements {

public static void main(String[] args) {

// Sample array

int[] array = {4, 3, 2, 7, 8, 2, 3, 1};

// HashSet to store already seen elements

HashSet<Integer> seen = new HashSet<>();

System.out.println("Duplicate elements in the array:");

// Iterate through the array

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

// Check if element has already been seen

if (!seen.add(array[i])) {

// Print the element if it's a duplicate

System.out.println(array[i]);

}

}

}

}

1. **WAP in Java for constructor overloading**

class MyClass {

int x, y;

// Default constructor

MyClass() {

x = 0;

y = 0;

}

// Parameterized constructor

MyClass(int a, int b) {

x = a;

y = b;

}

void display() {

System.out.println("x = " + x + ", y = " + y);

}

}

public class ConstructorOverloadingExample {

public static void main(String[] args) {

MyClass obj1 = new MyClass();

MyClass obj2 = new MyClass(10, 20);

obj1.display();

obj2.display();

}

}

1. **WAP in Java for method overloading**

public class MethodOverloadingExample {

public static void main(String[] args) {

int num1 = 10, num2 = 20;

double num3 = 30.5;

// Calling the overloaded methods

int sum1 = add(num1, num2);

System.out.println("Sum of integers: " + sum1);

double sum2 = add(num1, num3);

System.out.println("Sum of integer and double: " + sum2);

double sum3 = add(num2, num3);

System.out.println("Sum of doubles: " + sum3);

}

// Overloaded methods with different parameter types

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

return a + b;

}

public static double add(int a, double b) {

return a + b;

}

public static double add(double a, double b) {

return a

+ b;

}

}

1. **WAP in Java to count the total number of vowels and consonants in a string.**

import java.util.Scanner;

public class VowelConsonantCount {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter

a string: ");

String inputString = scanner.nextLine().toLowerCase();

int vowelCount = 0;

int consonantCount = 0;

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

char ch = inputString.charAt(i);

if (isVowel(ch)) {

vowelCount++;

} else if (Character.isLetter(ch)) {

consonantCount++;

}

}

System.out.println("Vowels: " + vowelCount);

System.out.println("Consonants: " + consonantCount);

}

private static boolean isVowel(char ch) {

return "aeiou".indexOf(ch) != -1;

}

}

1. **WAP in Java input a string and check whether it is palindrome or not.**

import java.util.Scanner;

public class PalindromeChecker {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Input string from user

System.out.print("Enter a string: ");

String input = scanner.nextLine();

// Clean the string: remove non-alphanumeric characters and convert to lowercase

String cleanedString = input.replaceAll("[^a-zA-Z0-9]", "").toLowerCase();

// Check if the cleaned string is a palindrome

if (isPalindrome(cleanedString)) {

System.out.println("The string is a palindrome.");

} else {

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

}

}

// Method to check if a string is a palindrome

public static boolean isPalindrome(String str) {

int left = 0;

int right = str.length() - 1;

while (left < right) {

if (str.charAt(left) != str.charAt(right)) {

return false;

}

left++;

right--;

}

return true;

}

}