1. public class PrimitiveDataTypesDemo {

public static void main(String[] args) {

// Integer data types

byte myByte = 127;

short myShort = 32767;

int myInt = 2147483647;

long myLong = 9223372036854775807L;

// Floating-point data types

float myFloat = 3.14f;

double myDouble = 3.14159;

// Character data type

char myChar = 'A';

// Boolean data type

boolean myBoolean = true;

// Printing values

System.out.println("Byte: " + myByte);

System.out.println("Short: " + myShort);

System.out.println("Int: " + myInt);

System.out.println("Long: " + myLong);

System.out.println("Float: " + myFloat);

System.out.println("Double: " + myDouble);

System.out.println("Char: " + myChar);

System.out.println("Boolean: " + myBoolean);

}

}

2. public class DataTypeConversionDemo {

public static void main(String[] args) {

// Implicit conversion (widening)

int intValue = 100;

long longValue = intValue;

float floatValue = longValue;

double doubleValue = floatValue;

System.out.println("Implicit conversions:");

System.out.println("int to long: " + longValue);

System.out.println("long to float: " + floatValue);

System.out.println("float to double: " + doubleValue);

// Explicit conversion (narrowing)

double doubleVal = 1234.567;

float floatVal = (float) doubleVal;

long longVal = (long) floatVal;

int intVal = (int) longVal;

System.out.println("\nExplicit conversions:");

System.out.println("double to float: " + floatVal);

System.out.println("float to long: " + longVal);

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

// Overflow and underflow scenarios

int maxValue = Integer.MAX\_VALUE;

byte overflowValue = (byte) maxValue;

int underflowValue = (int) 1.5;

System.out.println("\nOverflow and underflow scenarios:");

System.out.println("Overflow value: " + overflowValue);

System.out.println("Underflow value: " + underflowValue);

}

}

3. public class StringMethodsDemo {

public static void main(String[] args) {

// String constructors

String str1 = new String("Hello");

String str2 = "World";

// String methods

System.out.println("Length of str1: " + str1.length());

System.out.println("Character at index 2 in str1: " + str1.charAt(2));

System.out.println("Concatenation of str1 and str2: " + str1.concat(str2));

System.out.println("Index of 'l' in str1: " + str1.indexOf('l'));

System.out.println("Substring of str1: " + str1.substring(2));

System.out.println("Uppercase of str2: " + str2.toUpperCase());

System.out.println("Lowercase of str2: " + str2.toLowerCase());

System.out.println("Comparison of str1 and str2: " + str1.equals(str2));

}

}

4. public class StringBuilderVsStringBuffer {

public static void main(String[] args) {

int iterations = 1000000; // Number of iterations

// StringBuilder example

long startTime = System.currentTimeMillis();

StringBuilder stringBuilder = new StringBuilder();

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

stringBuilder.append("Hello");

}

long endTime = System.currentTimeMillis();

System.out.println("Time taken by StringBuilder: " + (endTime - startTime) + " milliseconds");

// StringBuffer example

startTime = System.currentTimeMillis();

StringBuffer stringBuffer = new StringBuffer();

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

stringBuffer.append("Hello");

}

endTime = System.currentTimeMillis();

System.out.println("Time taken by StringBuffer: " + (endTime - startTime) + " milliseconds");

}

}

5. public class MatrixCreation {

public static void main(String[] args) {

int[][] matrixA = new int[9][1];

int[][] matrixB = new int[9][1];

for (int i = 0; i < 9; i++) {

matrixA[i][0] = i + 1;

matrixB[i][0] = 9 - i;

}

System.out.println("Matrix A:");

printMatrix(matrixA);

System.out.println("Matrix B:");

printMatrix(matrixB);

}

public static void printMatrix(int[][] matrix) {

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

for (int j = 0; j < matrix[i].length; j++) {

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

}

System.out.println();

}

}

}

6. public class ConfusionMatrix {

public static void main(String[] args) {

int TP = 10, TN = 20, FP = 5, FN = 3;

double precision = (double) TP / (TP + FP);

double recall = (double) TP / (TP + FN);

double f1Score = 2 \* (precision \* recall) / (precision + recall);

System.out.println("True Positive (TP): " + TP);

System.out.println("True Negative (TN): " + TN);

System.out.println("False Positive (FP): " + FP);

System.out.println("False Negative (FN): " + FN);

System.out.println("F1-score: " + f1Score);

}

}

7. import java.util.Arrays;

public class MatrixCreation {

public static void main(String[] args) {

int[][] matrix = new int[3][3];

for (int[] row : matrix) {

Arrays.fill(row, 1);

}

System.out.println(Arrays.deepToString(matrix));

}

}

8. import java.util.Arrays;

public class EqualLengthArrays {

public static void main(String[] args) {

int[] array1 = {1, 2, 3, 4, 5};

int[] array2 = {1, 2, 3};

int minLength = Math.min(array1.length, array2.length);

array1 = Arrays.copyOf(array1, minLength);

array2 = Arrays.copyOf(array2, minLength);

System.out.println("Array 1: " + Arrays.toString(array1));

System.out.println("Array 2: " + Arrays.toString(array2));

}

}

9. public class GreatestOfThree {

public static void main(String[] args) {

int a = 10, b = 20, c = 15;

int greatest = (a > b) ? ((a > c) ? a : c) : ((b > c) ? b : c);

System.out.println("The greatest number is: " + greatest);

}

}

10. public class NestedIfDemo {

public static void main(String[] args) {

int a = 10, b = 20, c = 15;

if (a > b) {

if (a > c) {

System.out.println("The greatest value is in the first if statement (a > b and a > c)");

} else {

System.out.println("The greatest value is in the second if statement (c > a > b)");

}

} else {

if (b > c) {

System.out.println("The greatest value is in the third if statement (b > a and b > c)");

} else {

System.out.println("The greatest value is in the fourth if statement (c > a > b)");

}

}

}

}

11. public class AgeDifference {

public static void main(String[] args) {

int ageWhenYoungerBrotherIs24 = 28;

int ageDifference = 28 - 24;

int ageWhenOlderBrotherIs56 = ageWhenYoungerBrotherIs24 + ageDifference;

if (ageDifference > 3) {

int ageDifferenceBetweenOlderBrotherAndYoungerBrother = 56 - 5;

int ageWhenYoungerBrotherWas5 = ageWhenOlderBrotherIs56 - ageDifferenceBetweenOlderBrotherAndYoungerBrother;

System.out.println("Age of younger brother when older brother was 5: " + ageWhenYoungerBrotherWas5);

} else {

System.out.println("Age of the person when older brother is 56: " + ageWhenOlderBrotherIs56);

}

}

}

12. public class BirthYearCalculation {

public static void main(String[] args) {

int currentYear = 2073;

int eligibilityAge = 21;

int birthYear = currentYear - eligibilityAge;

if (birthYear > 2060) {

int voteEligibleYear = currentYear - eligibilityAge;

int yearsBetween = voteEligibleYear - 2023;

System.out.println("Years between vote-eligible year and 2023: " + yearsBetween);

} else {

int medianYear = (currentYear + 2023) / 2;

System.out.println("Median birth year: " + medianYear);

}

}

}

13. import java.util.Scanner;

public class GreatestValue {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter the value for loop 1:");

int value1 = scanner.nextInt();

System.out.println("Enter the value for loop 2:");

int value2 = scanner.nextInt();

if (value1 > value2) {

System.out.println("Loop 1 contains the greatest value.");

} else if (value2 > value1) {

System.out.println("Loop 2 contains the greatest value.");

} else {

System.out.println("Both loops have the same value.");

}

}

}

14. public class MissingNumbers {

public static void main(String[] args) {

int[] series = {1, 5, 11, 19};

for (int i = 0; i < series.length - 1; i++) {

int diff = series[i + 1] - series[i];

if (diff > 1) {

System.out.println("Missing numbers between " + series[i] + " and " + series[i + 1] + ":");

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

System.out.println(series[i] + j);

}

}

}

}

}

15. public class PrimeNumbers {

public static void main(String[] args) {

int[] array = {2, 3, 5, 7, 11, 13, 17, 19, 23, 29};

int primeCount = 0;

for (int num : array) {

boolean isPrime = true;

if (num < 2) {

continue;

}

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

if (num % i == 0) {

isPrime = false;

break;

}

}

if (isPrime) {

primeCount++;

}

}

System.out.println("Number of prime numbers in the array: " + primeCount);

}

}

16. public class MedianMeetingPoint {

public static void main(String[] args) {

int[] array1 = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};

int[] array2 = {10, 9, 8, 7, 6, 5, 4, 3, 2, 1};

int length = array1.length;

int[] combinedArray = new int[length \* 2];

System.arraycopy(array1, 0, combinedArray, 0, length);

System.arraycopy(array2, 0, combinedArray, length, length);

// Sort the combined array

java.util.Arrays.sort(combinedArray);

// Find the median

double median;

if (length % 2 == 0) {

median = (combinedArray[length - 1] + combinedArray[length]) / 2.0;

} else {

median = combinedArray[length];

}

System.out.println("Median meeting point: " + median);

}

}

17. import java.util.Scanner;

public class LuckGuess {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter your guess (1-3):");

int guess = scanner.nextInt();

switch (guess) {

case 1:

System.out.println("You guessed option 1 - It's your lucky guess!");

break;

case 2:

System.out.println("You guessed option 2 - Better luck next time!");

break;

case 3:

System.out.println("You guessed option 3 - Try again later!");

break;

default:

System.out.println("Invalid input! Please enter a number between 1 and 3.");

}

}

}

18. import java.util.Scanner;

public class ShortestPath {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter the number of iterations for loop 1:");

int iterations1 = scanner.nextInt();

System.out.println("Enter the number of iterations for loop 2:");

int iterations2 = scanner.nextInt();

switch (Integer.compare(iterations1, iterations2)) {

case -1:

System.out.println("Loop 1 is the shortest path.");

break;

case 0:

System.out.println("Both loops have the same number of iterations.");

break;

case 1:

System.out.println("Loop 2 is the shortest path.");

break;

}

}

}

19. public class LogicGates {

public static void main(String[] args) {

boolean input1 = true;

boolean input2 = false;

// OR gate

boolean orResult = input1 || input2;

System.out.println("OR gate result: " + orResult);

// AND gate

boolean andResult = input1 && input2;

System.out.println("AND gate result: " + andResult);

}

}

20. public class ArrayShift {

public static void main(String[] args) {

int[] A = {3, 6, 9, 12, 15};

int[] B = {5, 10, 15, 20, 25};

if (A.length == B.length) {

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

if (A[i] > B[i]) {

// Shift values from left to right

int temp = A[i];

A[i] = B[i];

B[i] = temp;

} else {

// Shift values from right to left

int temp = B[i];

B[i] = A[i];

A[i] = temp;

}

}

System.out.println("A array after shifting: ");

for (int num : A) {

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

}

System.out.println("\nB array after shifting: ");

for (int num : B) {

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

}

} else {

System.out.println("Arrays A and B must have the same length.");

}

}

}