**SAVEETHA SCHOOL OF ENGINEERING**

**SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES**

**INSTITUTE OF PLACEMENT AND TRAINING**

**CSA09 –JAVA PROGRAMMING**

**String**

1. Write a program to print the following pattern

Sample Input:

Enter the Character to be printed: %

Max Number of time printed: 3

%

% %

% % %

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the Character to be printed: ");

char ch = scanner.next().charAt(0);

System.out.print("Max Number of times printed: ");

int n = scanner.nextInt();

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

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

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

}

System.out.println();

}

}

}

1. Write a program to print hollow square symbol pattern?

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int side = scanner.nextInt();

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

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

if (i == 1 || i == side || j == 1 || j == side) {

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

} else {

System.out.print(" ");

}

}

System.out.println();

}

}

}

1. Write a program to print the below pattern

1

2 2

3 3 3

4 4 4 4

**public class Main {**

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

**int rows = 4;**

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

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

**System.out.print(i + " ");**

**}**

**System.out.println();**

**}**

**}**

**}**

1. Write a program to print the below pattern

1

4 9

16 25 36

49 64 81 100

public class Main {

public static void main(String[] args) {

int rows = 4;

int num = 1;

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

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

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

num++;

}

System.out.println();

}

}

}

1. Write a program to print the below pattern

1

2 2

3 3 3

4 4 4 4

3 3 3

2 2

1

public class Main {

public static void main(String[] args) {

int rows = 4;

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

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

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

}

System.out.println();

}

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

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

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

}

System.out.println();

}

}

}

1. Write a program to print hollow Square Dollar pattern?

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int side = scanner.nextInt();

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

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

if (i == 1 || i == side || j == 1 || j == side || i == j || j == side - i + 1) {

System.out.print("$ ");

} else {

System.out.print(" ");

}

}

System.out.println();

}

}

}

1. Write a program to print inverted pyramid pattern.

Input: no of rows: 3

Output

\*\*\*\*\*

\*\*\*

\*

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int rows = scanner.nextInt();

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

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

System.out.print(" ");

}

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

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

}

System.out.println();

}

}

}

**General:**

1. Write a program to reverse a number using loop?(Get the input from user)

Sample Input:

Number: 14567

Sample Output:

Reverse Number: 76541

Test cases:

1. -45721
2. 000
3. AD1947
4. !@#$%
5. 145\*999=144855

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Number: ");

int number = scanner.nextInt();

int reverse = 0;

while (number != 0) {

int digit = number % 10;

reverse = reverse \* 10 + digit;

number /= 10;

}

System.out.println("Reverse Number: " + reverse);

}

}

1. Write a program to convert the given decimal to binary and print the reverse of the binary decimal.

Input: 11

Output: 13

Explanation: (11)10 = (1011)2.

After reversing the bits we get:

(1101)2 = (13)10.

Test cases:

1. 25
2. Eighteen
3. 12
4. -18
5. 34.5

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Input decimal number: ");

int decimal = scanner.nextInt();

String binary = Integer.toBinaryString(decimal);

System.out.println("Binary representation: " + binary);

StringBuilder reversedBinary = new StringBuilder(binary);

reversedBinary.reverse();

int reversedDecimal = Integer.parseInt(reversedBinary.toString(), 2);

System.out.println("Reversed binary: " + reversedDecimal);

}

}

1. Write a program to find whether the person is eligible for vote or not. And if that particular person is not eligible, then print how many years are left to be eligible.

Sample Input:

Enter your age: 7

Sample output:

You are allowed to vote after 11 years

Test cases:

1. 25
2. Eighteen
3. 12
4. -18
5. 34.5

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int age = scanner.nextInt();

if (age < 0) {

System.out.println("Invalid age");

} else if (age < 18) {

int yearsLeft = 18 - age;

System.out.println("You are allowed to vote after " + yearsLeft + " years");

} else {

System.out.println("You are eligible to vote");

}

}

}

1. Find the LCM and GCD of n numbers?

Sample Input:

N value = 2

Number 1 = 16

Number 2 = 20

Sample Output:

LCM = 80

GCD = 4

Test cases:

1. N = 3, {12, 25, 30}
2. N = 2, {52, 25, 63}
3. N = 3, {17, 19, 11}
4. N = -2, {52, 60}
5. N = 2, {30, 45}

import java.util.Scanner;

public class Main {

// Function to calculate GCD of two numbers

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

if (b == 0)

return a;

return gcd(b, a % b);

}

// Function to calculate LCM of two numbers

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

return (a \* b) / gcd(a, b);

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("N value: ");

int n = scanner.nextInt();

int[] numbers = new int[n];

// Input numbers

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

System.out.print("Number " + (i + 1) + ": ");

numbers[i] = scanner.nextInt();

}

// Initialize lcm and gcd with first number

int lcmResult = numbers[0];

int gcdResult = numbers[0];

// Calculate LCM and GCD of all numbers

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

lcmResult = lcm(lcmResult, numbers[i]);

gcdResult = gcd(gcdResult, numbers[i]);

}

System.out.println("LCM = " + lcmResult);

System.out.println("GCD = " + gcdResult);

}

}

1. Write a program using function to calculate the simple interest. Suppose the customer is a senior citizen. He is being offered 12 percent rate of interest; for all other customers, the ROI is 10 percent.

Sample Input:

Enter the principal amount: 200000

Enter the no of years: 3

Is customer senior citizen (y/n): n

Sample Output:

Interest: 60000

Test Cases:

1. Principal: 2000 , Years: 0
2. Principal: 20000 , Years: -2
3. Principal: -2000 , Years: 2
4. Principal: 2 , Years: 2000
5. Principal: 0 , Years: 5

**import java.util.Scanner;**

**public class Main {**

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

**Scanner scanner = new Scanner(System.in);**

**System.out.print("Enter the principal amount: ");**

**double principal = scanner.nextDouble();**

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

**int years = scanner.nextInt();**

**System.out.print("Is the customer senior citizen (y/n): ");**

**char seniorCitizen = scanner.next().charAt(0);**

**double rateOfInterest = (seniorCitizen == 'y') ? 0.12 : 0.10;**

**double interest = principal \* rateOfInterest \* years;**

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

**}**

**}**

1. Write a program to print the Fibonacci series.

Sample Input:

Enter the n value: 6

**Sample Output:**

0 1 1 2 3 5

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the n value: ");

int n = scanner.nextInt();

int a = 0, b = 1;

System.out.print(a + "\t");

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

System.out.print(b + "\t");

int temp = b;

b = a + b;

a = temp;

}

}

}

1. [Java Program to Find Even Sum of Fibonacci Series Till number N](https://www.geeksforgeeks.org/java-program-to-find-sum-of-fibonacci-series-numbers-of-first-n-even-indexes/)?

Sample Input: n = 4

Sample Output: 33

(N = 4, So here the fibonacci series will be produced from 0th term till 8th term:0, 1, 1, 2, 3, 5, 8, 13, 21

Sum of numbers at even indexes = 0 + 1 + 3 + 8 + 21 = 33)

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter n: ");

int n = scanner.nextInt();

int a = 0, b = 1, sum = 0;

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

if (i % 2 == 0) {

sum += b;

}

int temp = b;

b = a + b;

a = temp;

}

System.out.println("Sum of even Fibonacci terms: " + sum);

}

}

1. Write a program to print the numbers from M to N by skipping K numbers in between?

Sample Input:

M = 50

N = 100

K = 7

Sample Output:

50, 58, 66, 74, …..

Test cases:

1. M = 15, N = 05, K = 02
2. .M = 25, N = 50, K = 04
3. M = 15, N = 100, K = -02
4. M = 0 , N = 0 , K = 2
5. M = 200 , N = 200 , K = 50

**import java.util.Scanner;**

**public class Main {**

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

**Scanner scanner = new Scanner(System.in);**

**System.out.print("M = ");**

**int m = scanner.nextInt();**

**System.out.print("N = ");**

**int n = scanner.nextInt();**

**System.out.print("K = ");**

**int k = scanner.nextInt();**

**for (int i = m; i <= n; i += k) {**

**System.out.print(i + ", ");**

**}**

**}**

**}**

1. Write a program to print all the composite numbers between a and b?

Sample Input:

A = 12

B = 19

Sample Output

14, 15, 16, 18

Test cases:

1. A = 11, B = 11
2. A = 20, B = 10
3. A = 0, B = 0
4. A = -5, B = 5
5. A = 7, B = -12

import java.util.Scanner;

public class Main {

public static boolean isComposite(int num) {

if (num <= 1) return false;

for (int i = 2; i <= num / 2; i++) {

if (num % i == 0) return true;

}

return false;

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("A = ");

int a = scanner.nextInt();

System.out.print("B = ");

int b = scanner.nextInt();

for (int i = a; i <= b; i++) {

if (isComposite(i)) {

System.out.print(i + ", ");

}

}

}

}

1. Find the factorial of n?

Sample Input:

N = 4

Sample Output:

4 Factorial = 24

Test cases:

1. N = 0
2. N = -5
3. N = 1
4. N = Q
5. N = 3A

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("N = ");

int n = scanner.nextInt();

int factorial = 1;

for (int i = 2; i <= n; i++) {

factorial \*= i;

}

System.out.println(n + " Factorial = " + factorial);

}

}

1. Find the year of the given date is leap year or not

Sample Input:

Enter Date: 04/11/1947

Sample Output:

Given year is Non Leap Year

Test cases:

1. 04/11/19.47
2. 11/15/1936
3. 31/45/1996
4. 64/09/1947
5. 00/00/2000

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter Date (MM/DD/YYYY): ");

String date = scanner.nextLine();

int year = Integer.parseInt(date.split("/")[2]);

boolean isLeapYear = (year % 4 == 0 && year % 100 != 0) || (year % 400 == 0);

if (isLeapYear) {

System.out.println("Given year is a Leap Year");

} else {

System.out.println("Given year is not a Leap Year");

}

}

}

1. Find the number of factors for the given number

Sample Input:

Given number: 100

Sample Output:

Number of factors = 9

Test cases:

1. 343
2. 1080
3. -243
4. 101010
5. 0

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Given number: ");

int number = scanner.nextInt();

int count = 0;

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

if (number % i == 0) {

count++;

}

}

System.out.println("Number of factors = " + count);

}

}

1. Write a program to print the given number is Perfect number or not?

Sample Input:

Given Number: 6

Sample Output:

It’s a Perfect Number

Test cases:

1. 17
2. 26!
3. 143
4. 84.1
5. -963

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Given Number: ");

int number = scanner.nextInt();

int sum = 0;

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

if (number % i == 0) {

sum += i;

}

}

if (sum == number) {

System.out.println("It’s a Perfect Number");

} else {

System.out.println("It’s not a Perfect Number");

}

}

}

1. Write a program to find the square, cube of the given decimal number

Sample Input:

Given Number: 0.6

Sample Output:

Square Number: 0.36

Cube Number:0.216

Test cases:

1. 12
2. 0
3. -0.5
4. 14.25
5. -296

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Given Number: ");

double number = scanner.nextDouble();

double square = number \* number;

double cube = square \* number;

System.out.println("Square Number: " + square);

System.out.println("Cube Number: " + cube);

}

}

1. Find the nth odd number after n odd number

Sample Input: N : 7

Sample Output:

Hence the values printed for i are 1 , 3 , 5.

Test cases:

1. N = 0
2. N = -6
3. N = 2021
4. N = -14.5
5. N = -196

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("N : ");

int n = scanner.nextInt();

int count = 0;

for (int i = 1; ; i += 2) {

if (i % 2 != 0) {

count++;

if (count == n) {

System.out.println("Nth Odd Number after Mth Odd Number: " + i);

break;

}

}

}

}

}

1. Program to find whether the given number is Armstrong number or not

Sample Input:

Enter number: 153

Sample Output:

Given number is Armstrong number

Test cases:

1. 370
2. 1
3. 371
4. 145678
5. 0.21345

**import java.util.Scanner;**

**public class Main {**

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

**Scanner scanner = new Scanner(System.in);**

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

**int number = scanner.nextInt();**

**int originalNumber = number;**

**int result = 0;**

**while (originalNumber != 0) {**

**int remainder = originalNumber % 10;**

**result += Math.pow(remainder, 3);**

**originalNumber /= 10;**

**}**

**if (result == number) {**

**System.out.println("Given number is Armstrong number");**

**} else {**

**System.out.println("Given number is not Armstrong number");**

**}**

**}**

**}**

1. Write a program to find the sum of digits of N digit number (sum should be single digit)

Sample Input:

Enter N value: 3

Enter 3 digit numbers: 143

Test cases:

1. N = 2, 158
2. N = 3, 14
3. N = 4, 0148
4. N = 1, 0004
5. N = 4, 7263

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter N value: ");

int n = scanner.nextInt();

System.out.print("Enter " + n + " digit numbers: ");

int number = scanner.nextInt();

int sum = 0;

while (number > 0 || sum > 9) {

if (number == 0) {

number = sum;

sum = 0;

}

sum += number % 10;

number /= 10;

}

System.out.println("Sum of digits of " + n + " digit number: " + sum);

}

}

1. Write a program to find the square root of a perfect square number(print both the positive and negative values)

Sample Input:

Enter the number: 6561

Sample Output:

Square Root: 81, -81

Test cases:

1. 1225
2. 9801
3. 1827
4. -100
5. 0

**import java.util.Scanner;**

**public class Main {**

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

**Scanner scanner = new Scanner(System.in);**

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

**int number = scanner.nextInt();**

**double squareRoot = Math.sqrt(number);**

**if (squareRoot == (int) squareRoot) {**

**System.out.println("Square Root: " + (int) squareRoot + ", " + (int) -squareRoot);**

**} else {**

**System.out.println("The given number is not a perfect square.");**

**}**

**}**

**}**

1. Write a program to given an integer n, return true if it is a power of three. Otherwise, return false.

Input =27

Output= true

Explanation: 27=33

Test cases:

1. 12
2. abc@45
3. 1827
4. -100
5. 0

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Input = ");

int n = scanner.nextInt();

boolean result = isPowerOfThree(n);

System.out.println("Output= " + result);

}

public static boolean isPowerOfThree(int n) {

if (n <= 0) return false;

while (n % 3 == 0) {

n /= 3;

}

return n == 1;

}

}

1. Write a program to given a string paragraph and a string array of the banned words banned, return the most frequent word that is not banned. It is guaranteed there is at least one word that is not banned, and that the answer is unique.

Input Paragraph=”Ram hit a ball, the hit ball flew far after it was hit”,

Banned = [hit]

Output=”Ball”

import java.util.HashMap;

public class Main {

public static void main(String[] args) {

String paragraph = "Ram hit a ball, the hit ball flew far after it was hit";

String[] banned = {"hit"};

String mostFrequentWord = mostCommonWord(paragraph, banned);

System.out.println("Most Frequent Word: " + mostFrequentWord);

}

public static String mostCommonWord(String paragraph, String[] banned) {

paragraph = paragraph.replaceAll("[^a-zA-Z]", " ").toLowerCase();

String[] words = paragraph.split("\\s+");

HashMap<String, Integer> map = new HashMap<>();

for (String word : words) {

if (!isBanned(word, banned)) {

map.put(word, map.getOrDefault(word, 0) + 1);

}

}

int maxFrequency = 0;

String mostFrequentWord = "";

for (String word : map.keySet()) {

if (map.get(word) > maxFrequency) {

maxFrequency = map.get(word);

mostFrequentWord = word;

}

}

return mostFrequentWord;

}

private static boolean isBanned(String word, String[] banned) {

for (String bannedWord : banned) {

if (bannedWord.equals(word)) {

return true;

}

}

return false;

}

}

1. Write a program to given a fixed-length integer array arr, duplicate each occurrence of zero, shifting the remaining elements to the right.

Input: arr = [1, 0, 2, 3, 0, 4, 5, 0]

Output: [1, 0, 0, 2, 3, 0, 0, 4]

Explanation: After calling your function, the input array is modified to [1, 0, 0, 2, 3, 0, 0, 4]

public class Main {

public static void main(String[] args) {

int[] arr = {1, 0, 2, 3, 0, 4, 5, 0};

duplicateZeros(arr);

for (int num : arr) {

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

}

}

public static void duplicateZeros(int[] arr) {

int n = arr.length;

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

if (arr[i] == 0) {

for (int j = n - 1; j > i; j--) {

arr[j] = arr[j - 1];

}

i++;

}

}

}

}

1. Write a program to given an array nums containing n distinct numbers in the range [0, n], return the only number in the range that is missing from the array.

Input nums = [3, 0, 1]

Output: 2

Explanation: n = 3 since there are 3 numbers, so all numbers are in the range [0, 3]. 2 is the missing number in the range since it does not appear in nums.

import java.util.\*;

public class MissingNumber {

public static int missingNumber(int[] nums) {

int n = nums.length;

// Calculate the sum of all numbers from 0 to n

int sum = (n \* (n + 1)) / 2;

// Calculate the sum of all numbers in the array

int arraySum = 0;

for (int num : nums) {

arraySum += num;

}

// The missing number is the difference between the two sums

return sum - arraySum;

}

public static void main(String[] args) {

int[] nums = {3, 0, 1};

System.out.println("Missing number: " + missingNumber(nums));

}

}

1. Write a program to given an integer array nums, find the subarray with the largest sum, and return its sum.

Input nums = [-2, 1,-3, 4,-1, 2, 1,-5, 4]

Output: 6

Explanation: The subarray [4,-1, 2, 1] has the largest sum 6.

public class LargestSumSubarray {

public static int maxSubArray(int[] nums) {

int maxSum = nums[0];

int currentSum = nums[0];

for (int i = 1; i < nums.length; i++) {

currentSum = Math.max(nums[i], currentSum + nums[i]);

maxSum = Math.max(maxSum, currentSum);

}

return maxSum;

}

public static void main(String[] args) {

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

System.out.println("Largest sum subarray: " + maxSubArray(nums));

}

}

1. Write a program to print the multiplication table of number m up to n.

Sample Input:

M = 4

N = 5

Sample Output:

1x4=4

2x4=8

3x4=12

4x4=16

5x4=20

Test cases:

M = 6, N = -3

M = -3, N = 5

M = 4, N = 0

M = 0, N = 0

M = -5, N = -5

public class MultiplicationTable {

public static void printMultiplicationTable(int m, int n) {

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

System.out.println(i + "x" + m + "=" + (i \* m));

}

}

public static void main(String[] args) {

int m = 4;

int n = 5;

printMultiplicationTable(m, n);

}

}

1. Write Java programs to implement multiple threads and apply join method for thread and thread has to be started after 500ms using sleep ().

public class MyThread extends Thread {

public void run() {

try {

Thread.sleep(500);

System.out.println("Thread running: " + Thread.currentThread().getName());

} catch (InterruptedException e) {

e.printStackTrace();

}

}

public static void main(String[] args) {

MyThread t1 = new MyThread();

MyThread t2 = new MyThread();

t1.start();

t2.start();

try {

t1.join();

t2.join();

} catch (InterruptedException e) {

e.printStackTrace();

}

System.out.println("Both threads have finished executing.");

}

}

1. Generate a Java code that implements java selection and iteration statements. Use do while loop to process a menu selection. When a menu is selected, it should display the syntax of the selected statements.

public class SelectionAndIteration {

public static void main(String[] args) {

int choice = 1;

do {

switch (choice) {

case 1:

System.out.println("Syntax of if-else statement:");

System.out.println("if (condition) {");

System.out.println(" // Code block to be executed if condition is true");

System.out.println("} else {");

System.out.println(" // Code block to be executed if condition is false");

System.out.println("}");

break;

case 2:

System.out.println("Syntax of for loop:");

System.out.println("for (initialization; condition; iteration) {");

System.out.println(" // Code block to be executed");

System.out.println("}");

break;

case 3:

System.out.println("Syntax of while loop:");

System.out.println("while (condition) {");

System.out.println(" // Code block to be executed");

System.out.println("}");

break;

default:

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

}

} while (choice < 4);

}

}

1. Create a simple generics class with type parameters for sorting values of different types.

import java.util.\*;

public class GenericSort<T extends Comparable<T>> {

private List<T> list;

public GenericSort(List<T> list) {

this.list = list;

}

public void sort() {

Collections.sort(list);

}

public List<T> getList() {

return list;

}

public static void main(String[] args) {

List<Integer> intList = Arrays.asList(5, 2, 8, 1, 9);

GenericSort<Integer> intSort = new GenericSort<>(intList);

intSort.sort();

System.out.println("Sorted Integer List: " + intSort.getList());

List<String> stringList = Arrays.asList("apple", "banana", "pear", "orange");

GenericSort<String> stringSort = new GenericSort<>(stringList);

stringSort.sort();

System.out.println("Sorted String List: " + stringSort.getList());

}

}

1. Create a class name ‘overload’. Write a program to assign the values for two values by different number of arguments using a single function.

public class Overload {

public void printValues(int... values) {

for (int value : values) {

System.out.println(value);

}

}

public static void main(String[] args) {

Overload obj = new Overload();

obj.printValues(1, 2, 3);

obj.printValues(4, 5, 6, 7, 8);

}

}

1. Write a Java Program to count the number of words in a string using Hash Map.

import java.util.\*;

public class WordCounter {

public static void main(String[] args) {

String paragraph = "Java is a programming language. Java is object-oriented.";

String[] words = paragraph.split("\\s+");

Map<String, Integer> wordCountMap = new HashMap<>();

for (String word : words) {

wordCountMap.put(word, wordCountMap.getOrDefault(word, 0) + 1);

}

System.out.println("Word count:");

for (Map.Entry<String, Integer> entry : wordCountMap.entrySet()) {

System.out.println(entry.getKey() + ": " + entry.getValue());

}

}

}

1. Write a Java Program to read an email and password from excel sheet by retrieving the cell using getRow() and getCell() method.

import org.apache.poi.ss.usermodel.\*;

import java.io.FileInputStream;

import java.io.InputStream;

public class ExcelReader {

public static void main(String[] args) {

try {

InputStream inputStream = new FileInputStream("credentials.xlsx");

Workbook workbook = WorkbookFactory.create(inputStream);

Sheet sheet = workbook.getSheetAt(0);

Row row = sheet.getRow(0);

String email = row.getCell(0).getStringCellValue();

String password = row.getCell(1).getStringCellValue();

System.out.println("Email: " + email);

System.out.println("Password: " + password);

workbook.close();

inputStream.close();

} catch (Exception e) {

e.printStackTrace();

}

}

}

1. Write a Java program to sorts the given value using Hash Map.

import java.util.\*;

public class HashMapSort {

public static void main(String[] args) {

Map<String, Integer> map = new HashMap<>();

map.put("apple", 5);

map.put("banana", 2);

map.put("orange", 8);

map.put("pear", 3);

// Sort map by values

List<Map.Entry<String, Integer>> sortedList = new ArrayList<>(map.entrySet());

sortedList.sort(Map.Entry.comparingByValue());

System.out.println("Sorted Map:");

for (Map.Entry<String, Integer> entry : sortedList) {

System.out.println(entry.getKey() + ": " + entry.getValue());

}

}

}

1. Write a Java program to find distinct characters and their count in a string.

import java.util.HashMap;

import java.util.Map;

public class DistinctCharacterCount {

public static void main(String[] args) {

String str = "hello world";

Map<Character, Integer> charCountMap = new HashMap<>();

// Count the occurrence of each character

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

if (c != ' ') {

charCountMap.put(c, charCountMap.getOrDefault(c, 0) + 1);

}

}

// Print distinct characters and their counts

System.out.println("Distinct characters and their counts:");

for (Map.Entry<Character, Integer> entry : charCountMap.entrySet()) {

System.out.println(entry.getKey() + ": " + entry.getValue());

}

}

}

1. Write a program to print all the unique characters in a String. For instance, if the input string is “abcb”, the output will be the characters ‘a’ and ‘c’ as they are unique. The character ‘b’ repeats twice and so it will not be printed.

import java.util.HashMap;

import java.util.Map;

public class UniqueCharacters {

public static void main(String[] args) {

String str = "abcb";

Map<Character, Integer> charCountMap = new HashMap<>();

// Count the occurrence of each character

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

charCountMap.put(c, charCountMap.getOrDefault(c, 0) + 1);

}

// Print unique characters

System.out.println("Unique characters:");

for (Map.Entry<Character, Integer> entry : charCountMap.entrySet()) {

if (entry.getValue() == 1) {

System.out.println(entry.getKey());

}

}

}

}