31.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}

ANSWER:

import java.util.Scanner;

public class LCM\_GCD {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int N = scanner.nextInt();

if (N <= 0) {

System.out.println("Invalid input for N. N should be a positive integer.");

return;

}

int[] numbers = new int[N];

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

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

numbers[i] = scanner.nextInt();

}

int lcm = calculateLCM(numbers);

int gcd = calculateGCD(numbers);

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

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

}

private static int calculateLCM(int[] numbers) {

int lcm = numbers[0];

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

lcm = (lcm \* numbers[i]) / calculateGCD(lcm, numbers[i]);

}

return lcm;

}

private static int calculateGCD(int a, int b) {

while (b != 0) {

int temp = b;

b = a % b;

a = temp;

}

return a;

}

private static int calculateGCD(int[] numbers) {

int gcd = numbers[0];

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

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

}

return gcd;

}

}

32.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

ANSWER:

import java.util.Scanner;

public class SimpleInterestCalculator {

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 a senior citizen? (y/n): ");

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

if (principal <= 0 || years < 0) {

System.out.println("Invalid input. Principal amount and number of years should be positive.");

return;

}

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

double interest = calculateSimpleInterest(principal, rateOfInterest, years);

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

}

private static double calculateSimpleInterest(double principal, double rateOfInterest, int years) {

return principal \* rateOfInterest \* years;

}

}

33.Write a program to print the Fibonacci series.

Sample Input:

Enter the n value: 6

**Sample Output:**

0 1 1 2 3 5

ANSWER:

import java.util.Scanner;

public class FibonacciSeries {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int n = scanner.nextInt();

if (n <= 0) {

System.out.println("Invalid input. Please enter a positive integer.");

return;

}

System.out.println("Fibonacci Series up to " + n + " terms:");

printFibonacciSeries(n);

}

private static void printFibonacciSeries(int n) {

int firstTerm = 0, secondTerm = 1;

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

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

int nextTerm = firstTerm + secondTerm;

firstTerm = secondTerm;

secondTerm = nextTerm;

}

}

}

**34:** [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)

ANSWER:

import java.util.Scanner;

public class EvenSumFibonacci {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int n = scanner.nextInt();

if (n <= 0) {

System.out.println("Invalid input. Please enter a positive integer.");

return;

}

System.out.println("Even Sum of Fibonacci Series up to " + n + ": " + findEvenSumFibonacci(n));

}

private static int findEvenSumFibonacci(int n) {

int sum = 0;

int firstTerm = 0, secondTerm = 1;

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

if (i % 2 == 0) {

sum += firstTerm;

}

int nextTerm = firstTerm + secondTerm;

firstTerm = secondTerm;

secondTerm = nextTerm;

}

return sum;

}

}

35: 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

ANSWER:

import java.util.Scanner;

public class SkipNumbers {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int M = scanner.nextInt();

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

int N = scanner.nextInt();

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

int K = scanner.nextInt();

if (K <= 0) {

System.out.println("Invalid input. K should be a positive integer.");

return;

}

System.out.println("Numbers from " + M + " to " + N + " by skipping " + K + " numbers:");

printNumbersWithSkip(M, N, K);

}

private static void printNumbersWithSkip(int M, int N, int K) {

for (int i = M; i <= N; i += K) {

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

}

}

}

36: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

ANSWER:

import java.util.Scanner;

public class CompositeNumbers {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int A = scanner.nextInt();

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

int B = scanner.nextInt();

if (A > B) {

System.out.println("Invalid input. A should be less than or equal to B.");

return;

}

System.out.println("Composite numbers between " + A + " and " + B + ":");

printCompositeNumbers(A, B);

}

private static boolean isComposite(int num) {

if (num <= 1) {

return false;

}

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

if (num % i == 0) {

return true;

}

}

return false;

}

private static void printCompositeNumbers(int A, int B) {

for (int i = A; i <= B; i++) {

if (isComposite(i)) {

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

}

}

}

}

37: 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

ANSWER:

import java.util.Scanner;

public class Factorial {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

String input = scanner.next();

if (!input.matches("\\d+")) {

System.out.println("Invalid input. Please enter a non-negative integer.");

return;

}

int N = Integer.parseInt(input);

if (N < 0) {

System.out.println("Invalid input. N should be a non-negative integer.");

return;

}

System.out.println(N + " Factorial = " + calculateFactorial(N));

}

private static long calculateFactorial(int n) {

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

return 1;

}

long factorial = 1;

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

factorial \*= i;

}

return factorial;

}

}

38.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

ANSWER:

import java.util.Scanner;

public class LeapYearChecker {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

String date = scanner.next();

if (!date.matches("\\d{2}/\\d{2}/\\d{4}")) {

System.out.println("Invalid date format. Please enter date in MM/DD/YYYY format.");

return;

}

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

if (isLeapYear(year)) {

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

} else {

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

}

}private static boolean isLeapYear(int year) {

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

}

}

39.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

ANSWER:

import java.util.Scanner;

public class NumberOfFactors {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int number = scanner.nextInt();

if (number < 0) {

System.out.println("Invalid input. Please enter a non-negative integer.");

return;

}

int factorCount = calculateNumberOfFactors(number);

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

}

private static int calculateNumberOfFactors(int number) {

int count = 0;

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

if (number % i == 0) {

if (number / i == i) {

count++;

} else {

count += 2; // Count both factors (i and number/i)

}

}

}

return count;

}

}

40.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

ANSWER:

import java.util.Scanner;

public class PerfectNumber {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int number = scanner.nextInt();

if (number <= 0) {

System.out.println("Invalid input. Please enter a positive integer.");

return;

}

if (isPerfectNumber(number)) {

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

} else {

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

}

}

private static boolean isPerfectNumber(int number) {

int sum = 1;

public class PerfectNumber {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int number = scanner.nextInt();

if (number <= 0) {

System.out.println("Invalid input. Please enter a positive integer.");

return;

}

if (isPerfectNumber(number)) {

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

} else {

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

}

}

private static boolean isPerfectNumber(int number) {

int sum = 1; // Start with 1 as all numbers are divisible by 1

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

if (number % i == 0) {

sum += i;

if (number / i != i) {

sum += number / i;

}

}

}

return sum == number;

}

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

if (number % i == 0) {

sum += i;

if (number / i != i) {

sum += number / i;

}

}

}

return sum == number;

}

}

41.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

ANSWER:

import java.util.Scanner;

public class SquareAndCube {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

double givenNumber = scanner.nextDouble();

double square = calculateSquare(givenNumber);

double cube = calculateCube(givenNumber);

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

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

}

private static double calculateSquare(double number) {

return Math.pow(number, 2);

}

private static double calculateCube(double number) {

return Math.pow(number, 3);

}

}

42.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

ANSWER:

import java.util.Scanner;

public class NthOddNumber {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int N = scanner.nextInt();

if (N < 0) {

System.out.println("Invalid input. N should be a non-negative integer.");

return;

}

int oddNumber = findNthOddNumber(N);

System.out.println("The nth odd number after " + N + " odd numbers is: " + oddNumber);

}

private static int findNthOddNumber(int N) {

int count = 0;

int number = 1;

while (count < N) {

number += 2;

count++;

}

return number;

}

}

43.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

ANSWER:

import java.util.Scanner;

public class ArmstrongNumber {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int number = scanner.nextInt();

if (number < 0) {

System.out.println("Invalid input. Please enter a non-negative integer.");

return;

}

if (isArmstrongNumber(number)) {

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

} else {

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

}

}

private static boolean isArmstrongNumber(int number) {

int originalNumber = number;

int sum = 0;

int digits = countDigits(number);

while (number > 0) {

int digit = number % 10;

sum += Math.pow(digit, digits);

number /= 10;

}

return sum == originalNumber;

}

private static int countDigits(int number) {

int count = 0;

while (number > 0) {

number /= 10;

count++;

}

return count;

}

}

44.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

ANSWER:

import java.util.Scanner;

public class SingleDigitSum {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int N = scanner.nextInt();

if (N <= 0) {

System.out.println("Invalid input. N should be a positive integer.");

return;

}

System.out.print("Enter " + N + " digit number: ");

int num = scanner.nextInt();

if (countDigits(num) != N) {

System.out.println("Invalid input. Please enter a " + N + " digit number.");

return;

}

int singleDigitSum = calculateSingleDigitSum(num);

System.out.println("Sum of digits (single digit): " + singleDigitSum);

}

private static int countDigits(int number) {

int count = 0;

while (number > 0) {

number /= 10;

count++;

}

return count;

}

private static int calculateSingleDigitSum(int number) {

int sum = 0;

while (number > 0) {

sum += number % 10;

number /= 10;

}

while (sum >= 10) {

sum = sum % 10 + sum / 10;

}

return sum;

}

}

45.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

ANSWER:

import java.util.Scanner;

public class SquareRoot {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

if (number < 0) {

System.out.println("Invalid input. Please enter a non-negative integer.");

return;

}

double squareRoot1 = Math.sqrt(number);

double squareRoot2 = -Math.sqrt(number);

System.out.println("Square Root: " + squareRoot1 + ", " + squareRoot2);

}

}

**46.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**

**ANSWER;**

public class PowerOfThree {

public static void main(String[] args) {

System.out.println(isPowerOfThree(12));

System.out.println(isPowerOfThree("abc@45"));

System.out.println(isPowerOfThree(1827));

System.out.println(isPowerOfThree(-100));

System.out.println(isPowerOfThree(0));

System.out.println(isPowerOfThree(27));

}

public static boolean isPowerOfThree(int n) {

if (n <= 0) {

return false;

}

while (n % 3 == 0) {

n /= 3;

}

return n == 1;

}

public static boolean isPowerOfThree(String input) {

try {

int n = Integer.parseInt(input);

return isPowerOfThree(n);

} catch (NumberFormatException e) {

return false;

}

}

}

**47.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”**

**ANSWER:**

import java.util.HashMap;

import java.util.HashSet;

import java.util.Map;

public class MostFrequentWord {

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 result = mostCommonWord(paragraph, banned);

System.out.println("Output: " + result); // Output: "ball"

}

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

paragraph = paragraph.toLowerCase().replaceAll("[^a-z ]", " "); // Convert to lowercase and remove non-alphabetic characters

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

HashSet<String> bannedSet = new HashSet<>();

for (String word : banned) {

bannedSet.add(word);

}

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

String mostFrequentWord = "";

int maxFrequency = 0;

for (String word : words) {

if (!bannedSet.contains(word) && !word.isEmpty()) {

int frequency = frequencyMap.getOrDefault(word, 0) + 1;

frequencyMap.put(word, frequency);

if (frequency > maxFrequency) {

maxFrequency = frequency;

mostFrequentWord = word;

}

}

}

return mostFrequentWord;

}

}

**48.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]**

**ANSWER;**

public class DuplicateZeros {

public static void main(String[ ] args) {

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

duplicateZeros(arr);

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

for (int num : arr) {

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

}

}

public static void duplicateZeros(int[ ] arr) {

int length = arr.length;

int zeroCount = 0;

for (int num : arr) {

if (num == 0) {

zeroCount++;

}

}

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

if (i + zeroCount < length) {

arr[i + zeroCount] = arr[i];

}

if (arr[i] == 0) {

zeroCount--;

if (i + zeroCount < length) {

arr[i + zeroCount] = 0;

}

}

}

}

}

**49.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.**

**ANSWER:**

public class MissingNumber {

public static void main(String[] args) {

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

int result = findMissingNumber(nums);

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

}

public static int findMissingNumber(int[] nums) {

int n = nums.length;

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

int actualSum = 0;

for (int num : nums) {

actualSum += num;

}

return expectedSum - actualSum;

}

}

**50.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.**

**ANSWER:**

public class MaxSubarraySum {

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};

int result = maxSubArray(nums);

System.out.println("The maximum subarray sum is: " + result);

}

}

**51.Write a java 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**

**ANSWER:**

import java.util.Scanner;

public class MultiplicationTable

{

public static void main(String[] args)

{

Scanner scanner = new Scanner(System.in);

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

int m = scanner.nextInt();

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

int n = scanner.nextInt();

if (n < 0)

{

System.out.println("Invalid input. N should be a non-negative integer.");

return;

}

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

{

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

}

}

}

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

**ANSWER:**

class MyThread extends Thread {

public void run()

{

try

{

Thread.sleep(500);

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

}

catch (InterruptedException e)

{

e.printStackTrace();

}

}

}

public class JoinExampleThread {

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("Main thread is finished.");

}

}

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

**ANSWER:**

import java.util.Arrays;

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

private T[ ] array;

public GenericSorter(T[ ] array) {

this.array = array;

}

public void sort( ) {

Arrays.sort(array);

}

public void displaySortedArray() {

System.out.println("Sorted Array: " + Arrays.toString(array));

}

public static void main(String[ ] args) {

Integer[ ] intArray = {5, 2, 8, 1, 3};

GenericSorter<Integer> intSorter = new GenericSorter<>(intArray);

intSorter.sort();

intSorter.displaySortedArray();

String[] stringArray = {"apple", "orange", "banana", "grape", "kiwi"};

GenericSorter<String> stringSorter = new GenericSorter<>(stringArray);

stringSorter.sort();

stringSorter.displaySortedArray();

}

}

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

**ANSWER:**

public class Overload {

public void assignValues(int a, int b) {

System.out.println("Assigning values using two arguments:");

System.out.println("Value of a: " + a);

System.out.println("Value of b: " + b);

}

public void assignValues(int a, int b, int c) {

System.out.println("Assigning values using three arguments:");

System.out.println("Value of a: " + a);

System.out.println("Value of b: " + b);

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

}

public void assignValues(int a, int b, int c, int d) {

System.out.println("Assigning values using four arguments:");

System.out.println("Value of a: " + a);

System.out.println("Value of b: " + b);

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

System.out.println("Value of d: " + d);

}

public static void main(String[] args)

{

Overload overloadObj = new Overload();

overloadObj.assignValues(10, 20);

overloadObj.assignValues(30, 40, 50);

overloadObj.assignValues(60, 70, 80, 90);

}

}

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

**ANSWER:**

import java.util.HashMap;

import java.util.Map;

public class WordCount

{

public static void main(String[ ] args)

{

String inputString = "This is a sample string. This string has words, and we want to count them.";

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

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

for (String word : words)

{

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

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

}

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

for (Map.Entry<String, Integer> entry : wordFrequencyMap.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.**

**ANSWERS;**

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

import org.apache.poi.xssf.usermodel.XSSFWorkbook;

import java.io.FileInputStream;

import java.io.IOException;

public class ExcelReader {

public static void main(String[] args) {

String excelFilePath = "path/to/your/excel/file.xlsx";

try (FileInputStream inputStream = new FileInputStream(excelFilePath);

Workbook workbook = new XSSFWorkbook(inputStream)) {

Sheet sheet = workbook.getSheetAt(0);

Row emailRow = sheet.getRow(0);

Cell emailCell = emailRow.getCell(0);

String email = emailCell.getStringCellValue();

Row passwordRow = sheet.getRow(1);

Cell passwordCell = passwordRow.getCell(0);

String password = passwordCell.getStringCellValue();

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

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

}

catch (IOException e)

{

e.printStackTrace();

}

}

}

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

**ANSWER;**

import java.util.HashMap;

import java.util.Map;

import java.util.TreeMap;

public class SortHashMap {

public static void main(String[ ] args) {

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

unsortedMap.put("Alice", 30);

unsortedMap.put("Bob", 25);

unsortedMap.put("Charlie", 35);

unsortedMap.put("David", 28);

TreeMap<String, Integer> sortedMap = new TreeMap<>(unsortedMap);

System.out.println("Unsorted HashMap:");

displayMap(unsortedMap);

System.out.println("\nSorted TreeMap:");

displayMap(sortedMap);

}

private static void displayMap(Map<String, Integer> map) {

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

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

}

}

}

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

**ANSWER:**

import java.util.HashMap;

import java.util.Map;

public class DistinctCharactersCount {

public static void main(String[] args) {

String inputString = "programming";

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

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

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

}

System.out.println("Distinct Characters and their Count:");

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

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

}

}

}

**60.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.**

**ANSWER:**

import java.util.HashSet;

import java.util.Set;

public class UniqueCharacters {

public static void main(String[] args) {

String inputString = "abcb";

Set<Character> uniqueCharacters = new HashSet<>();

Set<Character> repeatedCharacters = new HashSet<>();

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

set

if (!uniqueCharacters.add(ch)) {

repeatedCharacters.add(ch);

}

}

uniqueCharacters.removeAll(repeatedCharacters);

System.out.println("Unique Characters in the String:");

for (char uniqueChar : uniqueCharacters) {

System.out.println(uniqueChar);

}

}

}