CODING QUESTIONS – TCS NQT 3rd October 2024

Q1. Find the Total minutes of exercise done and it’s average for a week

Input:

Day 1 exercise duration: 25

Day 2 exercise duration: 26

Day 3 exercise duration: 23

Day 4 exercise duration: 15

Day 5 exercise duration: 14

Day 6 exercise duration: 38

Day 7 exercise duration: 44

Result: 185 26.4

Solution –

C++ code

```
#include <bits/stdc++.h>

Using namespace std;


Int main() {

    Int duration, sum = 0;

    For(int I = 0; I < 7; i++) {

        Cout << “Day “ << i+1 << “ exercise duration: “;

        Cin >> duration;

        Sum += duration;
    }
    Double avg = static_cast<double>(sum) / 7;
    Cout << "\nTotal minutes: " << sum;
    Cout << "\nAverage minutes per day: " << avg;
    Return 0;
}
```

Java –

```
Import java.util.Scanner;

Public class Main {
    Public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Int sum = 0, duration;

        For (int I = 0; I < 7; i++) {
            System.out.print("Day " + (I + 1) + " exercise duration: ");
            Duration = scanner.nextInt();
            Sum += duration;
        }

        Double avg = (double) sum / 7;
        System.out.println("\nTotal minutes: " + sum);
        System.out.println("Average minutes per day: " + avg);

        Scanner.close();
    }
}
```

If the loop runs for n days instead of a fixed 7 days:

The time complexity is O(n) due to the loop running n times, while the space complexity is O(1) as only constant memory is used.

For Week : TC is O(1)

                    SC is O(1)

Q2. (print the total number of palindrome between the given range m and n,0<=m,n<=1000)

for example input1 (lowest range =0 and Highest range =20)

Input: 0 20

Output: 11

Reason: 0,1,2,3,4,5,6,7,8,9,11

These numbers are palindrome

Solution –

C++

```
#include <bits/stdc++.h>
Using namespace std;

Bool is_palindrome(int n) {
    Int original = n, reversed = 0;
    While (n > 0) {
        Reversed = reversed * 10 + (n % 10);
        N /= 10;
    }
    Return original == reversed;
}
Int main() {
    Int m, n, count = 0;
    Cout << “Enter the range of m and n: “;
    Cin >> m >> n;

    For (int I = m; I <= n; i++) {
        If (is_palindrome(i)) count++;
    }

    Cout << “Number of palindromes: “ << count;
    Return 0;
}
```

Java code –

```
Import java.util.Scanner;

Public class Main {
}

    Public static boolean isPalindrome(int n) {
        Int original = n, reversed = 0;
        While (n > 0) {
            Reversed = reversed * 10 + (n % 10);
            N /= 10;
        }
        Return original == reversed;
    }

    Public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the range of m and n: ");
        Int m = scanner.nextInt();
        Int n = scanner.nextInt();
        Int count = 0;

        For (int I = m; I <= n; i++) {
            If (isPalindrome(i)) {
                Count++;
            }
        }

        System.out.println("Number of palindromes: " + count);
        Scanner.close();
    }
}
```

The time complexity is O(k * log n), where k is the range size and log n is the number of digits in the largest number, while the space complexity is O(1) since no additional data structures are used.

04 OCT TCS NQT 2025

Question: Check if a number is perfect

Input: 28

Output: True (28 is a perfect number)

Solution:

C++ Code-

```
#include <iostream>
using namespace std;

bool isPerfect(int n) {
if (n <= 1) return false;
int sum = 1;
for (int i = 2; i * i <= n; i++) {
if (n % i == 0) {
sum += i;
if (i!= n/i) {
sum += n/i;
} }
}
return sum == n;
}
```

```
int main() {
int n;
cin >> n;
if (isPerfect(n)) {
cout << "True" << endl;  }
 else {
cout << "False" << endl;
}
return 0;
}
```

JAVA code –

```
import java.util.Scanner;

public class PerfectNumber {
    public static boolean isPerfect(int n) {
        if (n <= 1) return false;
        int sum = 1;
        for (int i = 2; i * i <= n; i++) {
            if (n % i == 0) {
                sum += i;
                if (i != n / i) {
                    sum += n / i;
                }   }
       }
        return sum == n;
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int n = scanner.nextInt();
        System.out.println(isPerfect(n));
    }
}
```

Question:

Given a space-separated string of words, write a function to count the frequency of each word in the string. The output should display each unique word followed by its frequency, with the words in the order of their first appearance. The output should capitalize the first letter of each word.

Example:

Input:

apple banana apple banana apple orange banana

Output:

apple 3 banana 2 orange 1

Solution :

C++ Code –

```
#include <bits/stdc++.h>
using namespace std;


void count_word_frequencies(const string &input_string) {
    stringstream ss(input_string);
    string word;
    map<string, int> frequency;
    vector<string> words_in_order;
    while (ss >> word) {
        // Capitalize the first letter of each word
        word[0] = toupper(word[0]);

        if (frequency.find(word) == frequency.end()) {
            words_in_order.push_back(word);
        }

        frequency[word]++;
    }

    for (const string &w : words_in_order) {
        cout << w << " " << frequency[w] << endl;
    }
}

int main() {
    string input_string;
    getline(cin, input_string);
    count_word_frequencies(input_string);
    return 0;
}
```

Java Code –

```
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

public class WordFrequency {

    public static void countWordFrequencies(String inputString) {
        String[] words = inputString.split(" ");
        Map<String, Integer> frequency = new HashMap<>();

        for (String word : words) {
            // Ensure the word contains only letters
            if (word.chars().allMatch(Character::isLetter)) {
                // Capitalize the first letter of the word
                word = Character.toUpperCase(word.charAt(0)) + word.substring(1);

                // Update frequency in the map
                frequency.put(word, frequency.getOrDefault(word, 0) + 1);
            }
        }

        // Print the words with their frequencies
        for (Map.Entry<String, Integer> entry : frequency.entrySet()) {
            System.out.print(entry.getKey() + " " + entry.getValue() + " ");
        }
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        String inputString = scanner.nextLine();


        countWordFrequencies(inputString);
        scanner.close();
    }
}
```

05 Oct TCS NQT 2025

QUESTION:

You are giving an integer like 123 Return true if sum of the digits is multiple of 3

INPUT:

123

OUTPUT:

true/True

Solution:

C++ Code –

```
#include <bits/stdc++.h>
using namespace std;

int main() {
    int n;
    cin >> n;
    int sum = 0;
    while (n > 0) {
        int digit = n % 10;  // Fixed the assignment operator
        sum += digit;
        n /= 10;
    }
    if (sum % 3 == 0)
        cout << "True" << endl;
    else
cout << "False" << endl;
    return 0;
 }
```

Java Code –

```
import java.util.Scanner;

public class Main {

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        int n = scanner.nextInt(); // Corrected assignment operator

        int sum = 0;

        while (n > 0) {
            int digit = n % 10; // Corrected assignment operator
            sum += digit;
            n /= 10; // Fixed spacing for clarity
        }

        // Corrected the print statement
        System.out.println(sum % 3 == 0 ? "True" : "False");
        scanner.close();
    }
}
```

QUESTION:

Given an integer array nums and an integer k, return the number of pairs (i, j) where i < j such that nums[i] - nums[j]] == k.

INPUT

12211

OUTPUT

4

Solution:

C++ Code –

```
#include <bits/stdc++.h>
using namespace std;

// Function to build a frequency map of the elements in nums
unordered_map<int, int> buildFrequencyMap(const vector<int>& nums) {
    unordered_map<int, int> store;
    for (int num : nums) {
        store[num]++;
    }
    return store;
}

// Function to count pairs with a difference of k
int countPairsWithDifference(const vector<int>& nums, int k, const unordered_map<int,
int>& store) {
    int ans = 0;

    // Iterate through the numbers to find pairs
    for (int num : nums) {
        if (store.find(num + k) != store.end()) {
            ans += store.at(num + k); // Count pairs where nums[i] - nums[j] = k
        }
    }

    return ans;
}

int main() {
    vector<int> arr;
    string line;
    getline(cin, line);
    stringstream ss(line);
    int ele;

    // Read the integers from the input line
    while (ss >> ele) {
        arr.push_back(ele);
    }

    // Read k from the input
    int k;
    cin >> k;

    // Build frequency map and count pairs
    unordered_map<int, int> store = buildFrequencyMap(arr);
    int result = countPairsWithDifference(arr, k, store);

    cout << result << endl; // Output the result
    return 0;
}
```

Java Code –

```
import java.util.*;

public class Main {

    // Function to build a frequency map of the elements in the list
    public static Map<Integer, Integer> buildFrequencyMap(List<Integer> nums) {
        Map<Integer, Integer> store = new HashMap<>();
        for (int num : nums) {
            store.put(num, store.getOrDefault(num, 0) + 1);
        }
        return store;
    }

    // Function to count pairs with a difference of k
    public static int countPairsWithDifference(List<Integer> nums, int k, Map<Integer,
Integer> store) {
        int ans = 0;

        for (int num : nums) {
            // Check if (num + k) exists in the frequency map
            if (store.containsKey(num + k)) {
                ans += store.get(num + k); // Add the frequency of (num + k)
            }
        }

        return ans;
    }

    public static void main(String[] args) {
        List<Integer> arr = new ArrayList<>();
        Scanner scanner = new Scanner(System.in);

        // Read the line of integers
        String line = scanner.nextLine();
        Scanner lineScanner = new Scanner(line);
        while (lineScanner.hasNextInt()) {
            arr.add(lineScanner.nextInt());
        }

        // Read k from the input
        int k = arr.remove(arr.size() - 1); // Remove the last element as k

        // Build frequency map and count pairs
        Map<Integer, Integer> store = buildFrequencyMap(arr);
        int result = countPairsWithDifference(arr, k, store);

        System.out.println(result); // Output the result

        lineScanner.close();
        scanner.close();
    }
}
```