



**CHANDIGARH
UNIVERSITY**

Discover. Learn. Empower.

MINOR PROJECT REPORT ON

WORD FREQUENCY COUNTER

in partial fulfilment for the award of the degree of

Bachelors of Computer Applications



**CHANDIGARH
UNIVERSITY**

Discover. Learn. Empower.

SUBJECT – Data Structures

Submitted by:

Name: Sukhpreet Singh

UID: 23BCA10398

Section: BCA – 3 “A”

Group: 4th

Submitted to:

Name: Mrs. Shilpi Mittal

Designation: Co-ordinator

ABSTRACT

Introduction:

The word frequency counter is a simple yet effective program that analyzes a given string of text to determine the frequency of each word. This project demonstrates the use of basic data structures in C++, specifically the map container, to efficiently store and retrieve word counts. The ability to count words is essential in many applications, such as text analysis, natural language processing, and data mining.

Technique:

□ Data Structures:

- Map: Utilizes `std::map` from the C++ Standard Library to associate words with their frequency counts.
- Strings: Uses `std::string` for handling text data.

□ Control Structures:

- Loops: Iterates through characters in the input string to extract words.
- Conditional Statements: Checks for spaces to determine word boundaries and updates frequency counts.

□ String Manipulation:

- Builds words character by character and manages transitions between words based on space characters.

Operating System:

□ **Operating System:** The program can be compiled and run on any operating system that supports a C++ compiler, such as:

- Windows (using MinGW or Visual Studio)
- Linux (using GCC)
- macOS (using Xcode or GCC)

□ **Hardware Requirements:** The program can run on standard computer hardware, including:

- Any modern CPU with at least 1 GHz clock speed
- Minimum of 1 GB RAM (more recommended for larger texts)
- Storage for the compiler and source files (a few MBs)

SUMMARY

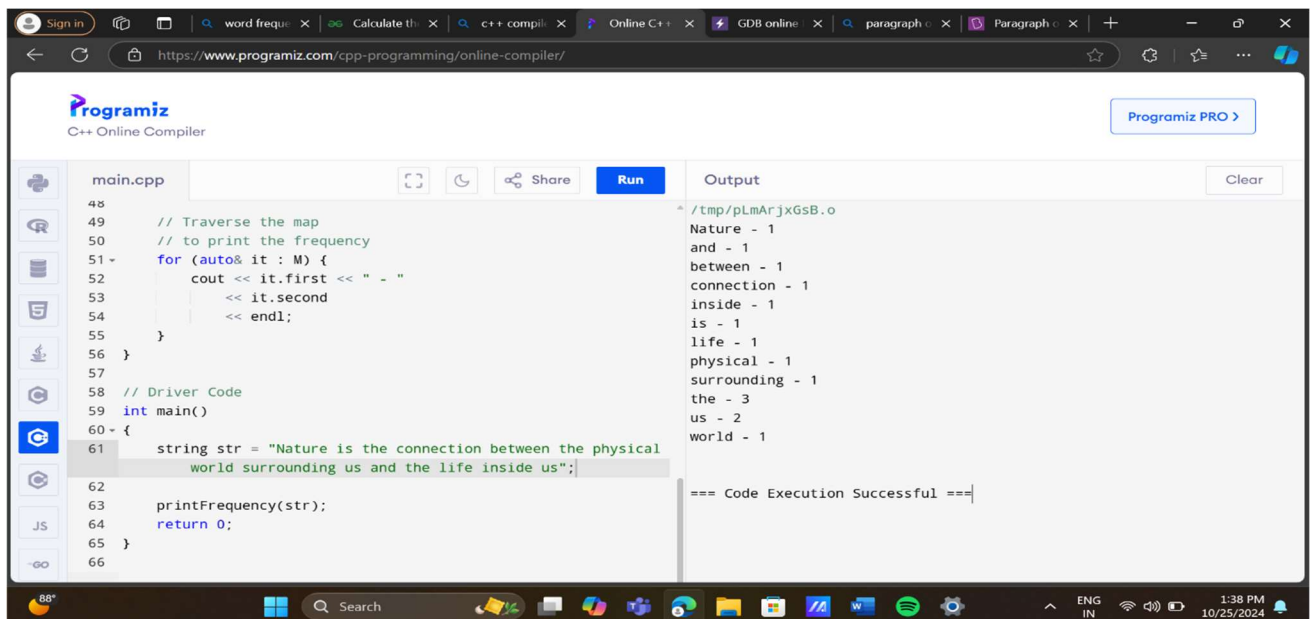
Input:

- The input for this program is a string of text, which can include multiple words separated by spaces. For example: "my name is varun".
- This input can be modified to accept user input from the console or read from a file.

Process:

- The program processes the input string character by character to identify words.
- It uses a map to store each word as a key and its frequency as the associated value.
- As it encounters a space, it checks whether the current word already exists in the map. If it does, it increments the count; if not, it inserts the word with a count of one.
- Finally, it handles the last word after the loop completes..

Output:



The screenshot displays the Programiz C++ Online Compiler interface. The code editor on the left shows a C++ program named `main.cpp` that calculates the frequency of words in a given string. The string is: "Nature is the connection between the physical world surrounding us and the life inside us". The program uses a `map` to store word frequencies and prints them out. The `Run` button has been clicked, and the output is displayed on the right. The output shows the frequency of each word: Nature - 1, and - 1, between - 1, connection - 1, inside - 1, is - 1, life - 1, physical - 1, surrounding - 1, the - 3, us - 2, and world - 1. The status bar at the bottom indicates "Code Execution Successful".

```
main.cpp
48
49 // Traverse the map
50 // to print the frequency
51 for (auto& it : M) {
52     cout << it.first << " - "
53         << it.second
54         << endl;
55 }
56 }
57
58 // Driver Code
59 int main()
60 {
61     string str = "Nature is the connection between the physical
62                 world surrounding us and the life inside us";
63
64     printFrequency(str);
65     return 0;
66 }
```

Output

```
^/tmp/pLmArjXGsB.o
Nature - 1
and - 1
between - 1
connection - 1
inside - 1
is - 1
life - 1
physical - 1
surrounding - 1
the - 3
us - 2
world - 1

=== Code Execution Successful ===
```