MINOR PROJECT REPORT

ON

WORD FREQUENCY COUNTER

in partial fulfilment for the award of the degree of

Bachelors of Computer Applications



SUBJECT – Data Structures

**Submitted by: Submitted to**:

**Name**: Armaan Singh **Name:** Mrs. Shilpi Mittal

**UID:** 23BCA10363 **Designation**: Co-ordinator

**Section:** BCA – 3 “A”

Group: **4**th

# 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 Armaan".

 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 output is a list of unique words along with their corresponding frequencies, printed to the console in the format:

arduino

word - frequency

* For the example input, the output would be:

csharp

my - 1

name - 1

is - 1

Armaan - 1

This project serves as a fundamental example of string processing and can be expanded further to include additional features, such as ignoring punctuation or handling case sensitivity.