

# **WORD COUNTER**



## **A PROJECT REPORT**

*Submitted by*

**PAVITHRA V (8115U23EC073)**

*in partial fulfillment of requirements for the award of the course*

**ECB1201 - JAVA PROGRAMMING**

*in*

**ELECTRONICS AND COMMUNICATION  
ENGINEERING**

**K. RAMAKRISHNAN COLLEGE OF ENGINEERING**

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

**SAMAYAPURAM – 621 112**

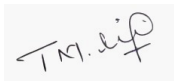
**DECEMBER - 2024**

**K. RAMAKRISHNAN COLLEGE OF ENGINEERING  
(AUTONOMOUS)**

**SAMAYAPURAM – 621 112**

**BONAFIDE CERTIFICATE**

Certified that this project report on “ ” is the bonafide work of **WORD COUNTER** is the bonafide work of **PAVITHRA V (8115U23EC073)** who carried out the project work during the academic year 2024 - 2025 under my supervision.



**SIGNATURE**

**Dr. T. M. NITHYA, M.E.,Ph.D.,**

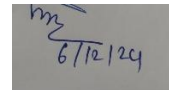
**HEAD OF THE DEPARTMENT**

**ASSOCIATE PROFESSOR**

Department of CSE

K.Ramakrishnan College of Engineering  
(Autonomous)

Samayapuram-621112.



**SIGNATURE**

**Mr. K . SWAMINATHAN,M.E.,MBA.,(Ph.D).,**

**SUPERVISOR**

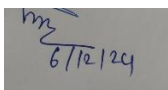
**ASSISTANT PROFESSOR**

Department of CSE

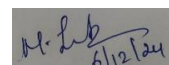
K.Ramakrishnan College of Engineering  
(Autonomous)

Samayapuram-621112.

Submitted for the viva-voce examination held on ...6.12.24.....



**INTERNAL EXAMINER  
EXAMINER**



**EXTERNAL**

## DECLARATION

I declare that the project report on **“WORD COUNTER”** is the result of original work done by us and best of our knowledge, similar work has not been submitted to **“ANNA UNIVERSITY CHENNAI”** for the requirement of Degree of **BACHELOR OF ENGINEERING**. This project report is submitted on the partial fulfilment of the requirement of the completion of the course **ECB1201 - JAVA PROGRAMMING**.

**Signature**

---

PAVITHRA V

Place: Samayapuram

Date: 6.12.24

## ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in-debt to our institution “**K.Ramakrishnan College of Engineering (Autonomous)**”, for providing us with the opportunity to do this project.

I glad to credit honourable chairman **Dr. K. RAMAKRISHNAN, B.E.**, for having provided for the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director **Dr. S. KUPPUSAMY, MBA, Ph.D.**, for forwarding to our project and offering adequate duration in completing our project.

I would like to thank **Dr. D. SRINIVASAN, B.E, M.E., Ph.D.**, Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to **Dr. T. M. NITHYA, M.E.,Ph.D.**, Head of the department, **COMPUTER SCIENCE AND ENGINEERING** for providing her encourage pursuing this project.

I express our deep expression and sincere gratitude to our project supervisor **Mr. K. SWAMINATHAN, M.E., MBA., (Ph.D.),** Department of **COMPUTER SCIENCE AND ENGINEERING**, for his incalculable suggestions, creativity, assistance and patience which motivated us to carry out this project.

I render our sincere thanks to Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

## VISION OF THE INSTITUTION

To achieve a prominent position among the top technical institutions.

### **MISSION OF THE INSTITUTION**

- M1: To bestow standard technical education par excellence through state of the art infrastructure, competent faculty and high ethical standards.
- M2: To nurture research and entrepreneurial skills among students in cutting edge technologies.
- M3: To provide education for developing high-quality professionals to transform the society.

### **VISION OF DEPARTMENT**

To create eminent professionals of Computer Science and Engineering by imparting quality education.

### **MISSION OF DEPARTMENT**

**M1:** To provide technical exposure in the field of Computer Science and Engineering through state of the art infrastructure and ethical standards.

**M2:** To engage the students in research and development activities in the field of Computer Science and Engineering.

**M3:** To empower the learners to involve in industrial and multi-disciplinary projects for addressing the societal needs.

## PROGRAM EDUCATIONAL OBJECTIVES

Our graduates shall

PEO1: Analyse, design and create innovative products for addressing social needs.

PEO2: Equip themselves for employability, higher studies and research.

PEO3: Nurture the leadership qualities and entrepreneurial skills for their successful career.

## PROGRAM SPECIFIC OUTCOMES (PSOs)

- **PSO1:** Apply the basic and advanced knowledge in developing software, hardware and firmware solutions addressing real life problems.
- **PSO2:** Design, develop, test and implement product-based solutions for their career enhancement.

## PROGRAM OUTCOMES (POs)

Engineering students will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions

- 5. Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations
- 6. The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice
- 7. Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development
- 8. Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
- 9. Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
- 10. Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
- 11. Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
- 12. Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

## **ABSTRACT**

This project aims to develop a Word Counter program in Java that analyzes a given text and calculates the frequency of each unique word. The program leverages string manipulation techniques to preprocess the text by converting it to lowercase and removing punctuation, ensuring consistent word matching. Using the Collections Framework, specifically a HashMap, it efficiently stores and retrieves word frequencies. A simple loop iterates through the processed words to update their counts in the map. The program outputs the frequency of each word, making it a practical tool for text analysis in applications like content processing, data mining, and natural language processing. The implementation emphasizes clarity, efficiency, and reusability. The word counter application is designed to analyze textual data and generate a frequency report of all unique words within a given text. This tool efficiently processes input texts, counts occurrences of each word, and outputs a comprehensive summary of word frequency. The application utilizes algorithms to handle various text formats, ensuring accurate counting even in cases with punctuation, case sensitivity, and stop words. The word counter can be applied in multiple fields such as linguistics, data analysis, and content optimization, offering a valuable resource for individuals and organizations looking to understand word patterns, improve readability, or analyze content for SEO and writing refinement.



## ABSTRACT WITH POs AND PSO<sub>s</sub> MAPPING

### CO 5 : BUILD JAVA APPLICATIONS FOR SOLVING REAL-TIME PROBLEMS.

ABSTRACT	POs MAPPED	PSOs MAPPED
The Word Counter application is designed to analyze textual data and generate a frequency report of all unique words within a given text. The development of this application aligns with key Program Outcomes (POs) such as PO1 (Engineering Knowledge), by applying fundamental principles of computer science to solve text analysis problems, and PO2 (Problem Analysis), by creating a solution that processes textual data effectively	<b>PO1 -3</b> <b>PO2 -3</b> <b>PO3 -3</b> <b>PO4 -3</b> <b>PO5 -3</b> <b>PO6 -3</b> <b>PO7 -3</b> <b>PO8 -3</b> <b>PO9 -3</b> <b>PO10 -3</b> <b>PO11-3</b> <b>PO12 -3</b>	<b>PSO1 -3</b> <b>PSO2 -3</b>

Note: 1- Low, 2-Medium, 3- High

# TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	viii
1	INTRODUCTION	
1.1	Objective	1
1.2	Overview	1
1.3	Java Programming concepts	2
2	PROJECT METHODOLOGY	
2.1	Proposed Work	4
2.2	Block Diagram	4
3	MODULE DESCRIPTION	
3.1	User Interface Design	5
3.2	Text Preprocessing	5
3.3	Text Analysis	5
3.4	Event Handling	6
3.5	Output and Visualization	6
4	CONCLUSION & FUTURE SCOPE	
4.1	Conclusion	7
4.2	Future Scope	8
	APPENDIX A (SOURCE CODE)	9
	APPENDIX B (SCREENSHOTS)	11
	REFERENCE	12

# **CHAPTER 1**

## **INTRODUCTION**

### **1.1 Objective**

The objective of this Java program is to develop a graphical user interface (GUI) application for performing advanced text analysis. Using Java's Abstract Window Toolkit (AWT), the application provides an interactive platform where users can input text, analyze its structure, and receive detailed statistics. The program calculates the total number of lines, words, and characters in the input text while also determining the frequency of each word, ignoring case and punctuation for consistency. By integrating these functionalities into a userfriendly interface with components such as text areas for input and output and a button to trigger the analysis, the application offers an efficient and intuitive tool for exploring text data. This project not only showcases the practical use of AWT for building desktop applications but also demonstrates the application of string manipulation and data structures for textual analysis

### **1.2 Overview**

The program is a desktop application developed using Java's Abstract Window Toolkit (AWT) that allows users to perform detailed text analysis. It features a graphical user interface (GUI) where users can input text, click a button to initiate analysis, and view results in a separate output area. The application processes the input to compute the total number of lines, words, and characters, along with generating a frequency count of each word, ensuring case insensitivity and ignoring punctuation. The design focuses on simplicity and usability, demonstrating the integration of GUI components . A GUI for text input and analysis results. □ Functions to calculate lines, words, characters,

and word frequencies. □ A simple, user-friendly interface showcasing Java's text processing and GUI capabilities.

## 1.3 Java Programming Concepts

### Basic concepts of OOPS

**Encapsulation:** The program encapsulates functionality within classes and methods, providing a clear separation of concerns. Example: The WordCounterAWT class contains all the properties (e.g., GUI components) and methods (e.g., analyzeText(), countWords()) related to the text analysis tool.

**Abstraction:** The program abstracts away the implementation details of text analysis by providing a simple and intuitive interface for the user. Users interact with the application via GUI components without needing to understand the underlying logic.

**Inheritance:** The AdvancedWordCounterAWT class inherits from the Frame class, leveraging Java's AWT framework to create and manage GUI components.

**Polymorphism:** Polymorphism is utilized in the ActionListener and WindowAdapter implementations, where methods like actionPerformed() and windowClosing() are overridden to provide custom functionality.

### Project-Specific Java Concepts

1. **Graphical User Interface (GUI):** Utilizes Java AWT components such as Frame, TextArea, Button, and Label to create an interactive and user-friendly interface.
2. **Event Handling:** Implements action listeners and window adapters to manage user interactions, such as button clicks and window events.

3. **Text Processing:** Applies string manipulation techniques, including splitting, replacing, and counting, to analyze text for word frequencies, line counts, and character counts.
4. **Data Structures:** Leverages a HashMap to efficiently store and retrieve word frequencies.
5. **Regular Expressions:** Uses regex to clean the input text by removing punctuation and ensuring case insensitivity.
6. **Modular Design:** Encapsulates functionality into methods, ensuring clarity, reusability, and maintainability of the code

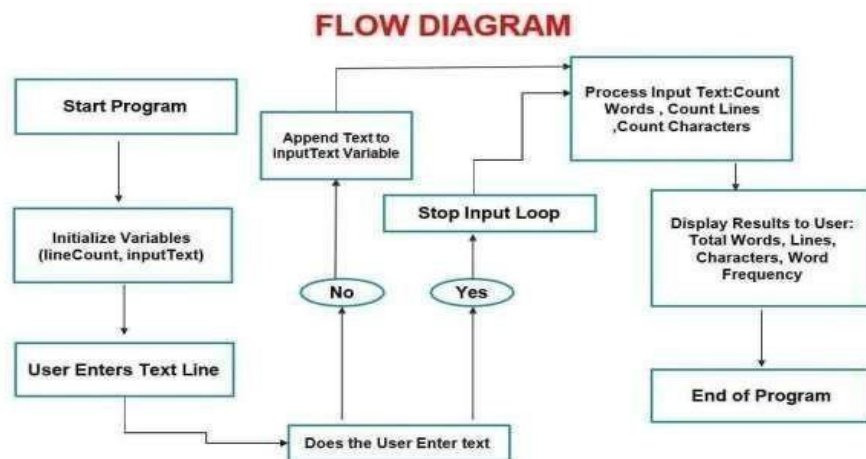
## CHAPTER 2

### PROJECT METHODOLOGY

#### 2.1 Proposed Work

The proposed work involves creating a Java-based desktop application using AWT for advanced text analysis. The application provides a user-friendly interface for inputting text and delivers statistics such as line, word, and character counts, along with word frequencies. By leveraging text processing techniques, efficient data structures, and event-driven programming, the project aims to offer an intuitive and practical tool for analyzing textual data.

#### 2.2 Block Diagram



## **CHAPTER 3**

### **MODULE DESCRIPTION**

#### **3.1 Module 1 User Interface Design**

This module is responsible for the user interface to developed using Java's Abstract Window Toolkit (AWT), featuring a simple and intuitive design. It includes components such as a TextArea for text input, a button to trigger analysis, and an output area to display results. The layout ensures usability by organizing elements in logical sections for input, interaction, and output.

#### **3.2 Module 2 Text Preprocessing**

Before analysis, this module handles the input text undergoes preprocessing to ensure accuracy. This includes converting text to lowercase, removing punctuation using regular expressions, and splitting it into lines or words.

These steps standardize the input, making it ready for efficient analysis.

#### **3.3 Module 3 Text Analysis**

This module manages the application performs various analyses on the processed text, including counting the total number of lines, words, and characters. Additionally, a HashMap is used to calculate the frequency of each word, providing a detailed breakdown of word usage.

### **3.4 Module 4 Event Handling**

This module checks for Event handling is implemented to manage user interactions. The button click event triggers the text analysis process, and a window closing event ensures proper disposal of the application. Action listeners and window adapters are used to handle these events seamlessly

### **3.5 Module 5 Output and Visualization**

This module is responsible for the results of the text analysis are displayed in a read only TextArea within the GUI. This includes statistics such as the total number of lines, words, characters, and a detailed word frequency list. The output is formatted for clarity, ensuring that users can easily interpret the results



## **CHAPTER 4**

### **CONCLUSION & FUTURE SCOPE**

#### **4.1 CONCLUSION**

The Word Counter program effectively demonstrates the use of fundamental programming concepts such as string manipulation, regular expressions, and data structures like HashMap. By efficiently counting word frequencies in a given text, it serves as a practical tool for basic text analysis. Its modular design ensures ease of understanding, scalability, and adaptability for advanced applications, such as natural language processing and data analysis. This program provides a strong foundation for tackling real-world challenges in text analytics and processing. The Word Counter program is a versatile application that highlights essential Java programming concepts, including string manipulation, regular expressions, and efficient data handling through HashMap. By normalizing text, splitting it into words, and counting their occurrences, the program provides a practical solution for analyzing text. Its modular and straightforward design ensures ease of implementation and adaptability for a wide range of use cases, such as text preprocessing, keyword analysis, and content summarization. Additionally, it demonstrates the power of structured coding for solving real-world problems, serving as a stepping stone toward more advanced applications like machine learning, natural language processing, and big data analytics.

This program is a simple yet powerful tool for learning programming while addressing practical challenges in text-based applications.

## **4.2 FUTURE SCOPE**

The Word Counter program has significant potential for growth and practical applications in various domains. It can be extended to support advanced text analysis, such as detecting word patterns, sentiment analysis, or summarizing large documents. By incorporating multi-language support, it can handle diverse datasets and global use cases. Scalability can be achieved through integration with big data frameworks like Apache Hadoop or Spark, enabling the program to process massive datasets efficiently. Additionally, it can be enhanced for real-time applications, such as analyzing live social media streams or chat logs. The program could also evolve into a user-friendly tool for web or mobile platforms, offering graphical visualizations like word clouds or charts for better data representation. Furthermore, its integration with databases, AI, and machine learning models could open doors to predictive text analytics and intelligent insights, making it a foundational tool for modern text processing and analysis tasks.

## APPENDIX A

```
import java.util.*;

public class WordCounter {    public
static void main(String[] args) {
    // Input text
    String text = "This is a sample text. This text is a sample.";

    // Step 1: Normalize text (lowercase and remove punctuation)    text =
text.toLowerCase().replaceAll("[^a-z ]", "");

    // Step 2: Split text into words
    String[] words = text.split("\\s+");

    // Step 3: Create a map to store word frequencies
    Map<String, Integer> wordCount = new HashMap<>();

    // Step 4: Count word occurrences    for (String word :
words) {        wordCount.put(word, wordCount.getDefault(word,
0) + 1);
    }
```

```
//      Step      5:      Print      word      frequencies
System.out.println("Word frequencies:");
    for (Map.Entry<String, Integer> entry : wordCount.entrySet()) {
        System.out.println(entry.getKey() + ": " + entry.getValue());
    }
}
}
```

## APPENDIX B

Output:

Word frequencies:

a: 2

this: 2

is: 2

text: 2

sample: 2

## REFERENCES

1. Java Documentation Official Java Platform Documentation by Oracle  
<https://docs.oracle.com/javase/>
2. OOP Principles and Design Patterns Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). Design Patterns: Elements of Reusable Object-Oriented Software.
3. Schildt, H. (2018). Java: The Complete Reference (11th Edition, Vol. 1, pp. 326–410). McGraw-Hill Education