

WORD COUNTER



A PROJECT REPORT

Submitted by

PADMACHARAN.S (230381171042112)

in partial fulfillment of requirements for the award of the course

CGB1201 - JAVA PROGRAMMING

In

COMPUTER SCIENCE AND ENGINEERING

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

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

SAMAYAPURAM – 621 112

NOVEMBER- 2024

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

SAMAYAPURAM – 621 112

BONAFIDE CERTIFICATE

Certified that this project report on “**WORD COUNTER**” is the bonafide work of **PADMACHARAN.S (2303811710421112)** who carried out the project work during the academic year 2024 - 2025 under my supervision.

CGB1201-JAVA PROGRAMMING
Dr.A.DELPHIN CAROLINA RANI, M.E.,Ph.D.,
HEAD OF THE DEPARTMENT
PROFESSOR

CGB1201-JAVA PROGRAMMING
Mrs.K.VALLI PRIYADHARSHINI, M.E.,(Ph.D.),
SUPERVISOR
ASSISTANT PROFESSOR

SIGNATURE

Dr.A.Delphin Carolina Rani, M.E.,Ph.D.,

HEAD OF THE DEPARTMENT

PROFESSOR

Department of CSE

K.Ramakrishnan College of Technology
(Autonomous)

Samayapuram–621112.

SIGNATURE

Mrs.K.Valli Priyadharshini, M.E.,(Ph.D.),

SUPERVISOR

ASSISTANT PROFESSOR

Department of CSE

K.Ramakrishnan College of Technology
(Autonomous)

Samayapuram–621112.

Submitted for the viva-voce examination held on 03/12/24

CGB1201-JAVA PROGRAMMING
Mr.MADHARMANNAN A, M.E.,
INTERNAL EXAMINER
ASSISTANT PROFESSOR

INTERNAL EXAMINER

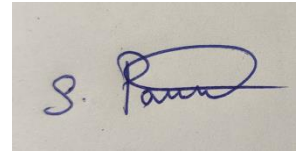
CGB1201-JAVA PROGRAMMING
Mrs.ARUNA PRIYA, M.E.,
EXTERNAL EXAMINER
ASSISTANT PROFESSOR
6104-DSEC, PERAMBALUR.

EXTERNAL EXAMINER

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 **CGB1201 - JAVA PROGRAMMING**.

Signature

A photograph of a handwritten signature in blue ink on a light-colored surface. The signature appears to be 'S. Padmasharan'.

PADMACHARAN S

Place: Samayapuram

Date: 03/12/24

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in-debt to our institution “**K.Ramakrishnan College of Technology (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. N. VASUDEVAN, M.Tech., Ph.D.**, Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to **Dr. A. DELPHIN CAROLINA RANI, 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 **Mrs. K. VALLI PRIYADHARSHINI, M.E., (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 serve the society by offering top-notch technical education on par with global standards

MISSION OF THE INSTITUTION

- Be a center of excellence for technical education in emerging technologies by exceeding the needs of the industry and society.
- Be an institute with world class research facilities
- Be an institute nurturing talent and enhancing the competency of students to transform them as all-round personality respecting moral and ethical values

VISION OF DEPARTMENT

To be a center of eminence in creating competent software professionals with research and innovative skills.

MISSION OF DEPARTMENT

M1: Industry Specific: To nurture students in working with various hardware and software platforms inclined with the best practices of industry.

M2: Research: To prepare students for research-oriented activities.

M3: Society: To empower students with the required skills to solve complex technological problems of society.

PROGRAM EDUCATIONAL OBJECTIVES

1. PEO1: Domain Knowledge

To produce graduates who have strong foundation of knowledge and skills in the field of Computer Science and Engineering.

2. PEO2: Employability Skills and Research

To produce graduates who are employable in industries/public sector/research organizations or work as an entrepreneur.

3. PEO3: Ethics and Values

To develop leadership skills and ethically collaborate with society to tackle real-world challenges.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO 1: Domain Knowledge

To analyze, design and develop computing solutions by applying foundational concepts of Computer Science and Engineering.

PSO 2: Quality Software

To apply software engineering principles and practices for developing quality software for scientific and business applications.

PSO 3: Innovation Ideas

To adapt to emerging Information and Communication Technologies (ICT) to innovate ideas and solutions to existing/novel problems

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

The **WORD COUNTER PROGRAM** in Java is a graphical application built using the Abstract Window Toolkit (AWT) and designed following Object-Oriented Programming (OOP) principles. It provides an interactive interface where users can input text, click a button, and view the word and character counts instantly. The program uses AWT components such as `TextArea` for text input, `Button` for triggering actions, `Label` for displaying results, and `Frame` as the application window. The design emphasizes OOP principles: **ENCAPSULATION** is achieved by separating the counting logic into distinct methods, **INHERITANCE** is utilized by extending `Frame` for GUI functionality, and **POLYMORPHISM** is demonstrated through the implementation of the `ActionListener` interface for event handling. By abstracting away low-level complexities and providing a simple yet effective GUI, the program exemplifies how AWT and OOP work together to create a modular and user-friendly application.

ABSTRACT WITH POs AND PSOs MAPPING

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

ABSTRACT	POs MAPPED	PSOs MAPPED
<p>This Java program counts the number of words in a given text input. It takes a string, splits it into words using spaces as delimiters, and then counts how many words are present. The program demonstrates basic string manipulation and can handle various types of input, such as sentences or paragraphs. The primary goal is to efficiently determine the word count in a text.</p>	<p>PO1 -3 PO2 -3 PO3 -3 PO4 -3 PO5 -3 PO6 -3 PO7 -3 PO8 -3 PO9 -3 PO10 -3 PO11-3 PO12 -3</p>	<p>PSO1 -3 PSO2 -3 PSO3 -3</p>

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

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	VIII
1	INTRODUCTION	1
	1.1 Objective	1
	1.2 Overview	1
	1.3 Java Programming concepts	2
2	PROJECT METHODOLOGY	3
	2.1 Proposed Work	3
	2.2 Block Diagram	3
3	MODULE DESCRIPTION	4
	3.1 Text Processing Module	4
	3.2 GUI Module	4
	3.3 Event Handling Module	4
	3.4 Utility Module	5
	3.5 File Handling Module	5
4	CONCLUSION & FUTURE SCOPE	6
	4.1 Conclusion	6
	4.2 Future Scope	6
	APPENDIX A (SOURCE CODE)	7
	APPENDIX B (SCREENSHOTS)	10
	REFERENCES	13

CHAPTER 1

INTRODUCTION

1.1 Objective

The objective of the Word Counter program in Java is to create a user-friendly application that accurately calculates the number of words and characters in a given text input. This program aims to demonstrate the integration of **graphical user interface (GUI)** design using **Swing/AWT** and the application of **Object-Oriented Programming (OOP)** principles to ensure modularity, scalability, and efficient functionality. It provides a simple yet effective tool for text analysis, catering to users who need quick insights into text structure, such as students, writers, or developers. Through this project, concepts like event handling, string manipulation, and GUI-based user interaction are also reinforced.

1.2 Overview

The Word Counter program in Java is a simple tool designed to count the number of words in a given string of text. The program works by taking a string input from the user, then using a regular expression to split the string into individual words based on whitespace characters (such as spaces, tabs, and newlines). It handles edge cases like empty strings and multiple spaces by trimming the input and ensuring that it only counts meaningful words. The result is displayed to the user, showing the total number of words in the string. This program serves as an excellent example for beginners to learn basic Java programming concepts such as input handling using the `Scanner` class, string manipulation with methods like `trim()` and `split()`, and the application of regular expressions. It also provides a foundation for more complex text processing tasks, making it a practical tool in various real-world scenarios, such as text analysis and natural language processing.

1.3 Java Programming Concepts

1. Swing GUI Components

- **Concept:** Swing is a Java library used for creating Graphical User Interfaces (GUIs). The program uses various Swing components to build an interactive user interface.
- **Components Used:**
 - **JFrame:** Represents the window that contains all other components.
 - **JLabel:** Used to display static text, such as instructions and output results.
 - **TextArea:** A text input field that allows users to enter multiple lines of text.
 - **Button:** Buttons that users can click to trigger actions.

2. Event Handling (ActionListeners)

- **Concept:** Event handling is crucial in GUI programming. Java allows you to respond to events (e.g., button clicks) using event listeners, specifically ActionListener for button events.
- **Usage in Program:**
 - The **Submit** and **Clear** buttons have action listeners that define what happens when the user clicks them.
 - When the **Submit** button is clicked, it processes the text in the JTextArea to calculate character and word counts.
 - When the **Clear** button is clicked, it resets the text area and the output labels.

3. GUI Window Management

- **Concept:** Proper management of the GUI window and ensuring that it behaves as expected is important for user experience.
- **Usage in Program:**
 - The program sets the window's size using `f.setSize()` and centers it on the screen with `f.setLocationRelativeTo(null)`.
 - It ensures the window is not resizable using `f.setResizable(false)`, which prevents the user from resizing the window.
 - The program also ensures that the application closes when the user clicks the close button using `f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE)`.

4. Exception Handling (Basic Concept)

- **Concept:** While not explicitly demonstrated in this program, the ability to handle exceptions is crucial in robust applications. A more complex version of this program might implement exception handling using try-catch blocks to manage unexpected input or other runtime issues.

5. String Manipulation

- **Usage in Program:**
 - The program processes the input string using methods like `strip()` (to remove leading and trailing spaces) and `length()` (to count characters).
 - The program also uses a loop to count characters excluding spaces and calculate word counts by counting spaces in the input string.

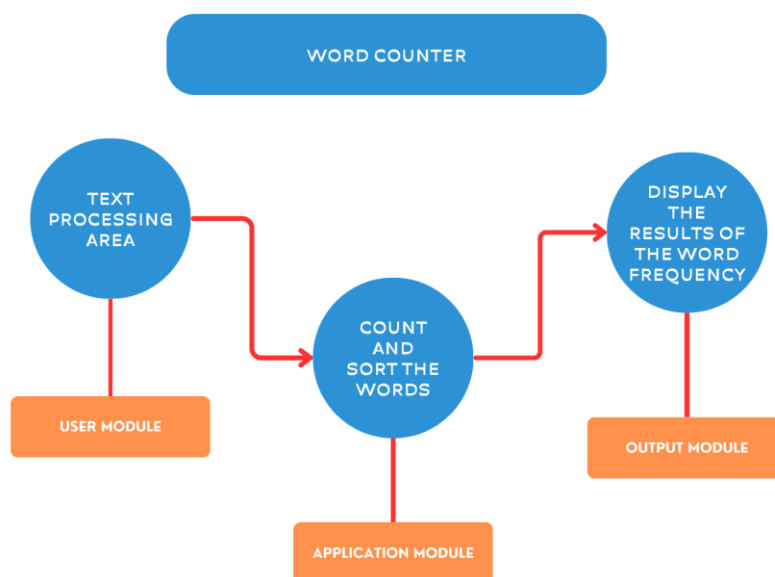
CHAPTER 2

PROJECT METHODOLOGY

2.1 Proposed Work

The proposed work for the Character Count Program in Java aims to enhance its functionality, aesthetics, and technical robustness, transforming it into a more versatile and user-friendly tool. Functionally, the program can be expanded to include advanced features such as counting sentences, paragraphs, special characters, and digits, alongside real-time updates as the user types. Additional capabilities like file handling, saving and loading text, and multi-language support will further increase its utility. Aesthetically, the user interface can be modernized using libraries like JavaFX, offering a sleek design with customizable themes, responsive layouts, and dynamic feedback for results. Technically, the program can benefit from algorithm optimization for large text inputs, robust error handling, and a modular code structure for maintainability and scalability. Advanced features such as text summarization, sentiment analysis via API integration, and cross-platform versions for web and mobile will broaden its application scope.

2.2 Block Diagram



CHAPTER 3

MODULE DESCRIPTION

3.1 Text Processing Module

The **Text Processing Module** is responsible for handling all the text-related operations, such as counting characters, words, and other relevant metrics. This module is designed to be independent and reusable, making it easy to extend or modify without affecting other parts of the program.

3.2 GUI Module

The **GUI Module** is responsible for designing and managing the graphical user interface (GUI) of the Character Count Program. It provides an interactive platform for users to input text, trigger analysis, and view the results. The GUI module integrates seamlessly with the **Text Processing Module** to display processed text metrics.

3.3 Event Handling Module

The **Event Handling Module** is responsible for managing user interactions and triggering appropriate actions based on those events. It acts as a bridge between the **GUI Module** and the **Text Processing Module**, ensuring that user inputs and button clicks are processed effectively and results are displayed accurately.

3.4 Utility Module

The **Utility Module** serves as a collection of helper functions and reusable components that simplify tasks like input validation, formatting, and exception handling. It is designed to be lightweight, modular, and extendable, providing utility methods that can be called across various modules without duplicating code.

The Utility Module is a backbone component that simplifies and standardizes repetitive tasks across the program. It enhances the overall robustness, maintainability, and usability of the Character Count Program by providing versatile, reusable, and easy-to-use methods.

3.5 File Handling Module

The **File Handling Module** is responsible for managing file operations such as reading text input from files and saving processed results to files. It enhances the usability of the program by enabling users to work with external text files, making the program more versatile and functional for larger text-based tasks.

By incorporating this module, the program becomes more versatile and user-friendly, capable of handling both individual and large-scale text processing tasks.

CHAPTER 4

CONCLUSION & FUTURE SCOPE

4.1 CONCLUSION

The **Word Counter Program** in Java provides an efficient and user-friendly solution for counting words, characters, and analyzing text input. By combining fundamental Java programming concepts such as **GUI creation**, **event handling**, and **text processing**, the program offers a functional tool that can be used for various text-related tasks, whether for educational purposes, personal use, or as a starting point for more advanced text analysis applications.

4.2 FUTURE SCOPE

The **Java Word Counter Program** has a significant potential for future development. From real-time counting to advanced text analysis, multilingual support, and cloud integration, there are numerous opportunities to enhance its functionality. By addressing user needs and staying up-to-date with the latest technology trends, the program can evolve into a powerful and versatile tool suitable for a wide range of professional, academic, and personal use cases. With continued development and feature expansion, it can become a comprehensive text processing solution with a broad user base.

APPENDIX A

(SOURCE CODE)

```
import javax.swing.*;
import java.awt.event.*;
import java.awt.Font;

class Main {
    public static void main(String args[]) {
        JFrame f = new JFrame("Character Count");
        JLabel l2, l3, l4;
        JTextArea text;
        JLabel l1;
        JButton submit, clear;
        text = new JTextArea("");
        submit = new JButton("SUBMIT");
        clear = new JButton("CLEAR");
        l1 = new JLabel("Enter Your string Here : ");
        l2 = new JLabel("Character with Spaces : ");
        l3 = new JLabel("Character Without Spaces : ");
        l4 = new JLabel("Words : ");
        Font txtFont = new Font(Font.SERIF, Font.PLAIN, 16);
        l1.setFont(txtFont);
        l2.setFont(txtFont);
        l3.setFont(txtFont);
        l4.setFont(txtFont);
        l1.setBounds(10, 25, 200, 30);
        text.setBounds(18, 60, 450, 300);
```

```

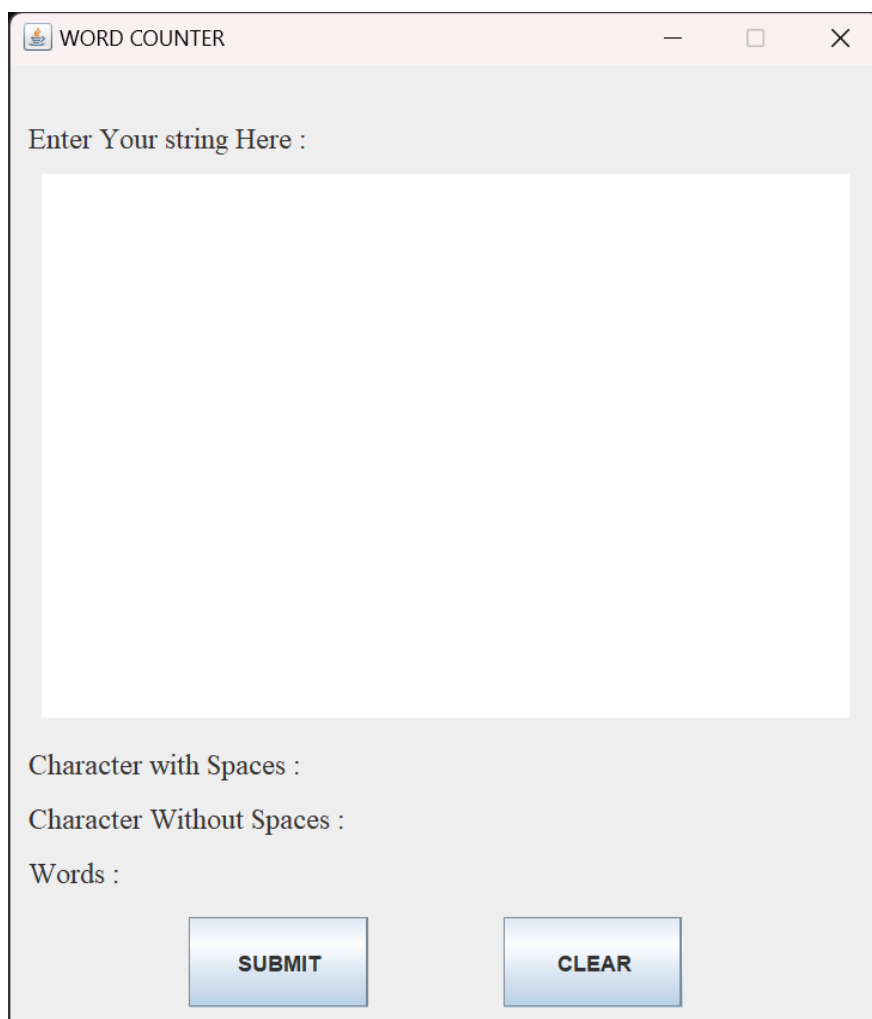
12.setBounds(10, 370, 400, 30);
13.setBounds(10, 400, 400, 30);
14.setBounds(10, 430, 400, 30);
submit.setBounds(100, 470, 100, 50);
clear.setBounds(275, 470, 100, 50);
text.setLineWrap(true);
text.setWrapStyleWord(true);
submit.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        String str = text.getText().strip();
        int count = 0, i, word = 0;
        12.setText("Character with Spaces : " + str.length());
        for (i = 0; i < str.length(); i++) {
            if (str.charAt(i) != ' ')
                count++;
            else
                word++;
        }
        13.setText("Character Without Spaces : " + count);
        14.setText("Words : " + (word + 1));
    }
});
clear.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        text.setText("");
        12.setText("Character with Spaces : ");
        13.setText("Character Without Spaces : ");
        14.setText("Words : ");
    }
}

```

```
    });  
    f.add(l1);  
    f.add(text);  
    f.add(l2);  
    f.add(l3);  
    f.add(l4);  
    f.add(submit);  
    f.add(clear);  
    f.setSize(500, 570);  
    f.setResizable(false);  
    f.setLayout(null);  
    f.setLocationRelativeTo(null);  
    f.setVisible(true);  
    f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
}  
}
```

APPENDIX B (SCREENSHOTS)

INPUT CONSOLE



A screenshot of a Java Swing window titled "WORD COUNTER". The window has a standard Mac OS X title bar with a red close button, a yellow maximize button, and a green minimize button. The main content area is light gray and contains the following elements:

- A label "Enter Your string Here :" in a dark gray font.
- A large white rectangular text area for input.
- Three labels at the bottom left: "Character with Spaces :", "Character Without Spaces :", and "Words :", each in a dark gray font.
- Two blue buttons with white text: "SUBMIT" and "CLEAR", positioned at the bottom right.

TEXT PROCESSING

WORD COUNTER

Enter Your string Here :

K RAMAKRISHNAN COLLEGE OF TECHNOLOGY TRICHY

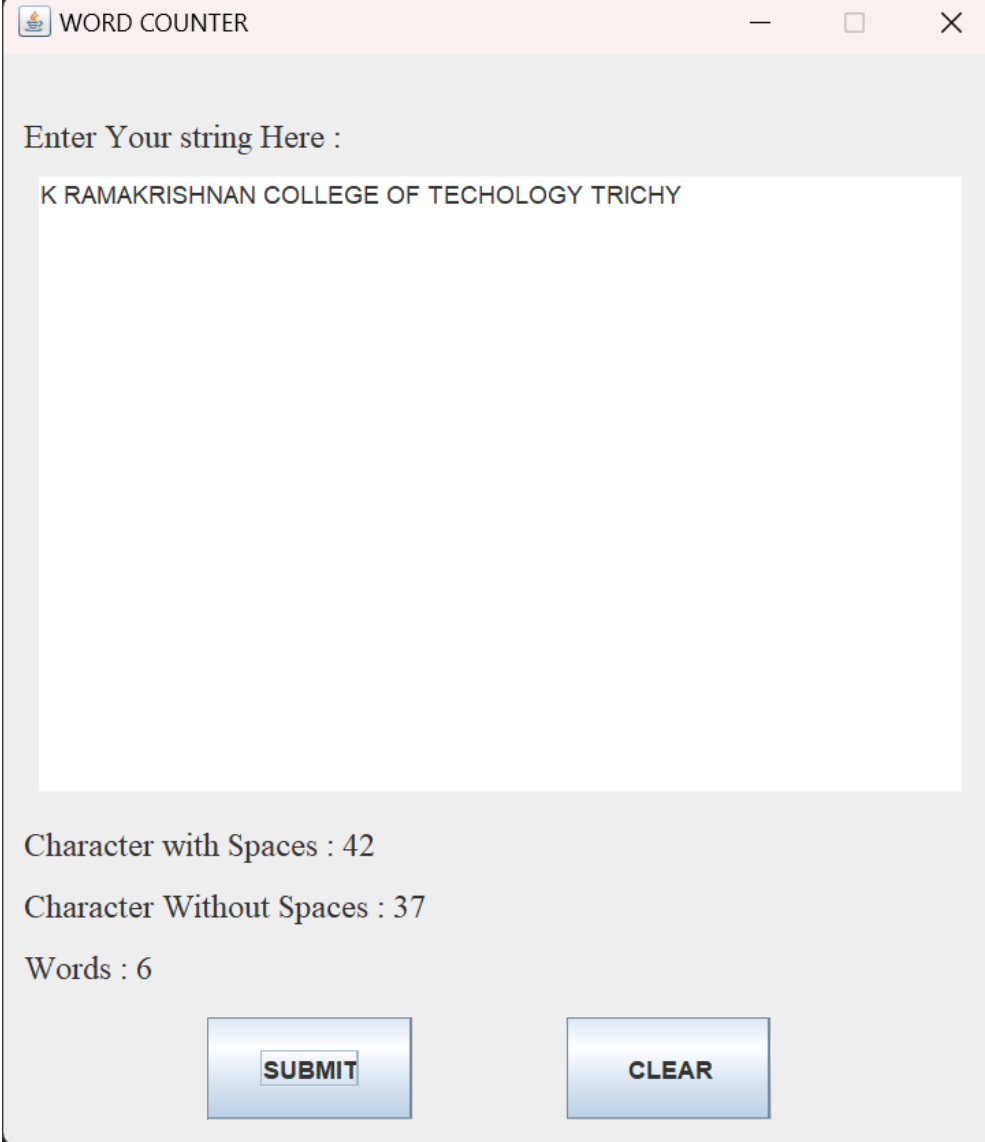
Character with Spaces :

Character Without Spaces :

Words :

SUBMIT CLEAR

FINAL PROCESS



WORD COUNTER

Enter Your string Here :

K RAMAKRISHNAN COLLEGE OF TECHNOLOGY TRICHY

Character with Spaces : 42

Character Without Spaces : 37

Words : 6

SUBMIT CLEAR

REFERENCES

❖ WEBSITES

STACK OVERFLOWS
BAELDUNG
JAVATPONIT

❖ YOUTUBE VIDEOS

KEVIN'S GUIDE

LINK : <https://www.youtube.com/watch?v=jsZo3JBjr90>

SDET-QA

LINK : https://youtu.be/dTWfNdfpr_g?si=dbpmYSjdP32symuj