A PROJECT REPORT ON

SPELLING CHECKER - OOPDS

REPORT SUBMITTED TOWARDS PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE AWARD OF THE DEGREE OF

MASTER OF TECHNOLOGY
IN
Embedded Systems

SUBMITTED BY

Name of the student	PRN
ANKITA GUPTA	23070147001
ANIKETH MEHTA	23070147002
SAILESH KUMAR PASAM	23070147005

॥वसुधैव कुटुम्बकम्॥

SYMBIOSIS INSTITUTE OF TECHNOLOGY, PUNE – 412115
(A CONSTITUENT OF SYMBIOSIS INTERNATIONAL (DEEMED UNIVERSITY))

SYMBIOSIS INSTITUTE OF TECHNOLOGY, PUNE – 412115

(A CONSTITUENT OF SYMBIOSIS INTERNATIONAL (DEEMED UNIVERSITY))

2023-24

MASTER OF TECHNOLOGY

IN

Embedded Systems

CERTIFICATE

This is to certify that **ANKITA GUPTA (23070147001), ANIKETH MEHTA (23070147002), AND SAILESH KUMAR PASAM (23070147005)** studying M.Tech in **EMBEDDED SYSTEM** have satisfactorily completed **SPELLING CHECKER** project in the semester **I** during the academic year 2023-2024

Signature of Course Instructor

ACKNOWLEDGMENT

We would like to express our sincere gratitude to all those who have contributed to the successful completion of this project. First and foremost, we extend our deepest thanks to Dr. Shripad Deshpande, our project supervisor, for providing valuable guidance, support, and encouragement throughout the duration of this project. Your expertise and insights have been instrumental in shaping the direction of our work.

We are also grateful to the Department of E&TC of Symbiosis Institute of Technology for providing the necessary resources and facilities for the completion of this project. The conducive environment and access to relevant materials have greatly contributed to the quality of the outcome.

Lastly, I want to acknowledge the countless sources from various researchers, scholars, and authors whose work laid the foundation for our project. The wealth of knowledge available in the literature has been invaluable in shaping my understanding and approach.

This project would not have been possible without the collective support and contributions of all those mentioned above. Thank you for being a part of this journey.

ANKITA GUPTA - 23070147001
ANIKETH MEHTA - 23070147002
SAILESH KUMAR PASAM - 23070147005

Title: Spelling Checker

Objective: Write a C program to check the spelling of each word in paragraph by comparing with the words in dictionary. If spelling is incorrect or not present in the dictionary, then provide the option to (1) add it to the dictionary (2) ignore and move forward. When you run the program again, the dictionary updated last time should be available.

Algorithm:

Step 1 –

- The program starts by including necessary libraries and defining constants, such as the maximum word length and file names.

Step 2 –

- There's a function called `checkSpelling` that takes a word and a file pointer to a dictionary as input.
- It reads words from the dictionary file and compares them with the provided word, ignoring case.
- If the word is found in the dictionary, it returns 1; otherwise, it returns 0.

Step 3 -

- Another function, `sanitizeWord`, is used to remove trailing commas or full stops from a word.

Step 4 -

- The `toLowercase` function converts a word to lowercase.

Step 5 -

- The `main` function opens the dictionary file for reading and appending. If it fails, an error message is displayed.
- The user is prompted to enter a paragraph (up to 100 words), which is then written to a file called "paragraph.txt."
- The paragraph file is opened for reading.
- Each word from the paragraph is read, sanitized, converted to lowercase, and checked for spelling errors using the `checkSpelling` function.
- If a spelling error is found, the user is given options: add the word to the dictionary or ignore it.
- Depending on the user's choice, the program either adds the word to the dictionary or continues without adding.
- After processing the entire paragraph, the files are closed.

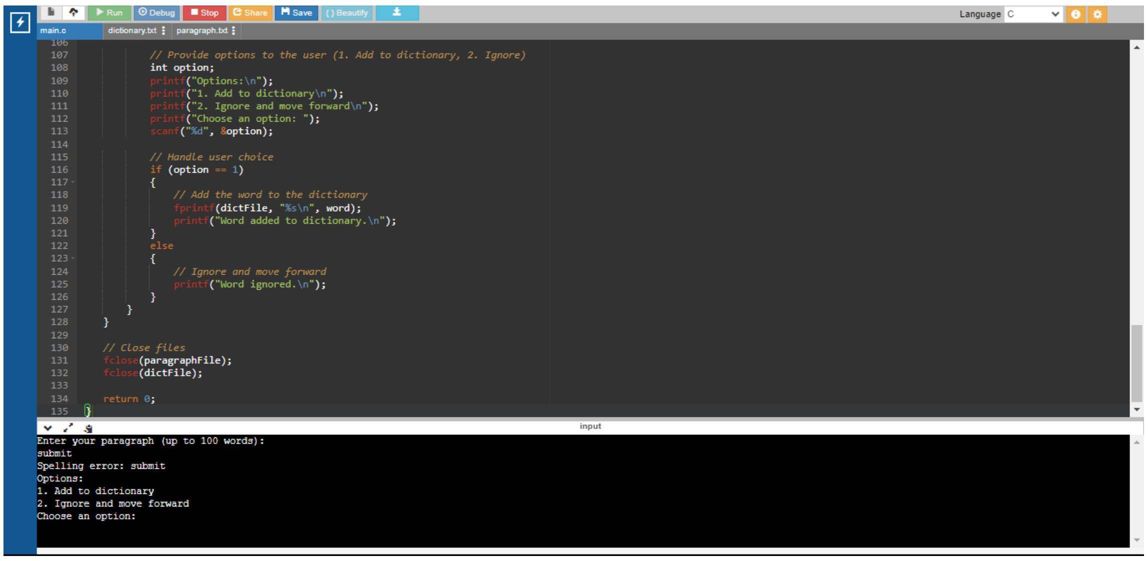
Step 6 –

- If the user chooses to add a word to the dictionary, the program appends the new word to the dictionary file.

Step 7 -

- Finally, all open files are closed, and the program ends. The next time when run the program, it will use the updated dictionary from the last run. This way, the dictionary grows over time as the user adds new w

Output:



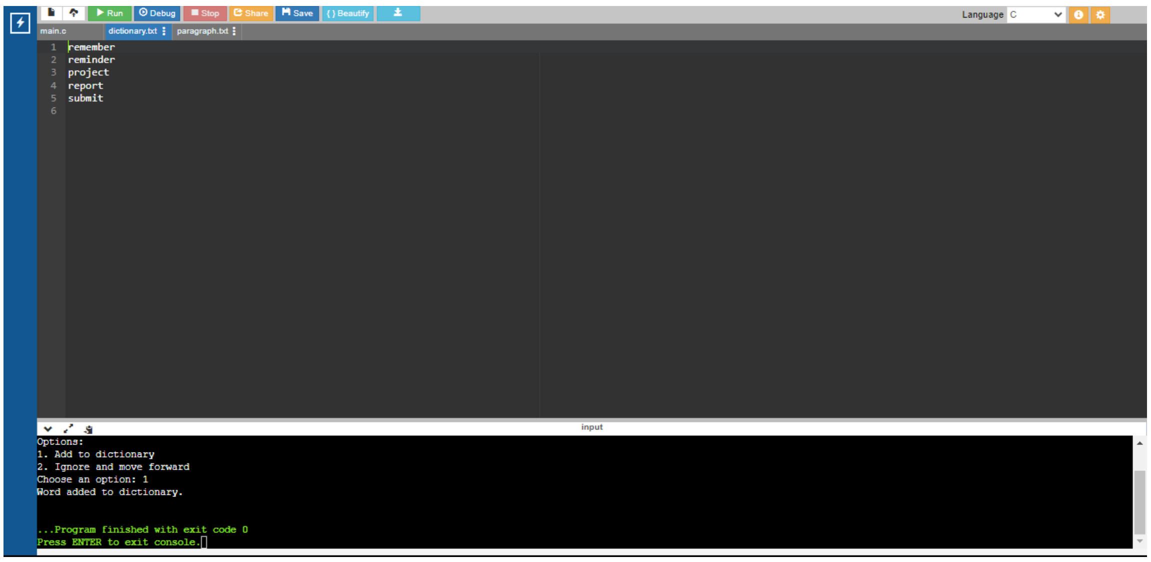
```c
            // Provide options to the user (1. Add to dictionary, 2. Ignore)
            int option;
            printf("Options:\n");
            printf("1. Add to dictionary\n");
            printf("2. Ignore and move forward\n");
            printf("Choose an option: ");
            scanf("%d", &option);

            // Handle user choice
            if (option == 1)
            {
                // Add the word to the dictionary
                fprintf(dictFile, "%s\n", word);
                printf("Word added to dictionary.\n");
            }
            else
            {
                // Ignore and move forward
                printf("Word ignored.\n");
            }
        }
    }

    // Close files
    fclose(paragraphFile);
    fclose(dictFile);

    return 0;
}
```

```
Enter your paragraph (up to 100 words):
submit
Spelling error: submit
Options:
1. Add to dictionary
2. Ignore and move forward
Choose an option:
```



```
remember
reminder
project
report
submit
```

```
Options:
1. Add to dictionary
2. Ignore and move forward
Choose an option: 1
Word added to dictionary.


...Program finished with exit code 0
Press ENTER to exit console.
```