Spell Checker System

PROJECT REPORT

Submitted by

KAPEESH M

RegisterNo:717822F122

ARUNESHWARAN V

RegisterNo:717822F106

LOKESH SELVAM P T

RegisterNo:717822F127

DATA STRUCTURES LABORATORY

(III SEMESTER)

KARPAGAM COLLEGE OF ENGINEERING

(Autonomous)

COIMBATORE–641032

DECEMBER 2023

KARPAGAM
COLLEGE OF ENGINEERING
Rediscover | Refine | Redefine
(An ISO 9001:2015 and ISO 14001:2015 Certified institution)

CERTIFICATE

Certified that this project report titled **"SPELL CHECKER SYSTEM"**is the bonafide work of **Mr. KAPEESH.M**, **Mr. ARUNESWARAN.V** ,and **Mr. LOKESH SELVAM.P.T** who carried out the project under my supervision. Certified further, that to the best of my knowledge the work reported here in does not form part of any other project report. I understand the policy on plagiarism and declare that the project and publications are my own work, except where specifically acknowledged and has not been copied from other sources or been previously submitted for award or assessment.

Examiner

Faculty In-Charge

ABSTRACT

A spell-checker program is designed to ensure the correctness of words in a given text by comparing them against a predefined dictionary of correctly spelled words. The primary goal of the spell- checker is to identify and flag words that do not match entries in the dictionary, thereby assisting users in identifying and correcting potential spelling errors.

The program employs string comparison techniques to assess the accuracy of each input word, providing feedback on whether the word is correctly spelled or if alternative suggestions for correction are available.

1 PROBLEM STATEMENT

Spell Checker system:

PROJECT INTRODUCTION:

The spell checker should be able to " quickly" find out all spelling mistakes in a given text file written in English. Use any data structure and algorithm of your own choice.

FEATURES:

- Check grammar and spelling
- Correct your spellings using suggestions
- Ignored words

2 REQUIRED SPECIFICATIONS

HARDWARE REQUIREMENTS:

- Processor: Pentium III866MHz
- RAM:128 MDSDEAM
- Monitor:15 inch color
- HardDisk:20 GB
- Floppy drive: 1.44MB
- Keyboard:102Keys
- CD drive: LG52X
- Mouse : 2 buttons

SOFTWARE REQUIREMENTS:

Operating System:Windows 8/10

Language: C

Software:DevC++Compiler

3 DATA STRUCTURE USED

This Employee Record System C project comprises the following data structures:

- An Array data structure, or simply an Array is a data structure consisting of a collection of elements,each identified by atleast one array index or key. An Array is stored such that position of each element can be computed from its index.

- The simplest type of data structure is a linear array,also called one- dimensional array. The elements of an array data structure are required to have the same size and should use the same data representation.

- Arrays are among the oldest and most important data structure, and are used in almost every program . They are also used to implement many other data structure, such as strings and lists.

APPLICATIONS:

- Arrays are used to implement other data structures ,such as lists, heaps,hash tables ,queues ,stacks, strings etc.,
- Array-based implementations of other data structures are frequently simple and space efficient, requiring little space overhead.
- Arrays can be used to determine partial or complete control flow in programs , as a compact alternatives to multiple if statements.
- Array are used to implement mathematical vectors matrices, as well as other kinds of rectangular tables.

ALOGRITHM :

STEP 1:	
	Accept the input word that needs to be checked for spelling.
STEP 2:	
	Load a dictionary of correctly spelled words into the program. This dictionary can be a text file containing a list of words, and you can load it into an array or a data structure suitable for quick lookups.
STEP 3:	
	Compare the input word against the words in the dictionary to check if it is correctly spelled. You can use string comparison functions in C to check if the input word matches any word in the dictionary.
STEP 4:	
	If the input word is not found in the dictionary, provide suggestions for possible correct spellings. You can implement a simple algorithm to find words in the dictionary that are close in terms of edit distance or phonetic similarity.
STEP 5:	
	Display the result of the spell check to the user.
if	If the word is correctly spelled, inform the user that it is correct. If the word is not found in the dictionary, provide suggestions available, or simply indicate that the word is misspelled.

STOP

4 SOURCE CODE

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include <limits.h>

#define MAX_WORD_LENGTH 50
#define HASH_TABLE_SIZE 100

struct Node {
   char word[MAX_WORD_LENGTH];
   struct Node* next;
};

struct HashTable {
   struct Node* table[HASH_TABLE_SIZE];
};

void toLowerCase(char *word) {
   for (int i = 0; word[i]; i++) {
      word[i] = tolower(word[i]);
   }
}
int hashFunction(char *word) {
     int hash = 0;
     while (*word) {
            hash = (hash << 5) + *word++;
         }
         return hash % HASH_TABLE_SIZE;
     }
void insertIntoHashTable(struct HashTable* ht, char *word) {
        int index = hashFunction(word);
        struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
        if (newNode == NULL) {
         // Handle memory allocation failure
          exit(1);
        }
```

```
    strcpy(newNode->word, word);
    newNode->next = ht->table[index];
    ht->table[index] = newNode;
}

int isInHashTable(struct HashTable* ht, char *word) {
    int index = hashFunction(word);
    struct Node* current = ht->table[index];
    while (current != NULL) {
        if (strcmp(word, current->word) == 0) {
            return 1; // Word is in the hash table
        }
        current = current->next;
    }

return 0; // Word is not in the hash table
}

int min(int a, int b, int c) {
    return a < b ? (a < c ? a : c) : (b < c ? b : c);
}

int levenshteinDistance(char *word1, char *word2) {
    int len1 = strlen(word1);
    int len2 = strlen(word2);

int dp[len1 + 1][len2 + 1];
for (int i = 0; i <= len1; i++) {
     for (int j = 0; j <= len2; j++) {
          if (i == 0) {
             dp[i][j] = j;
          } else if (j == 0) {
             dp[i][j] = i;
          } else {
             dp[i][j] = min(
                dp[i - 1][j] + 1,
                dp[i][j - 1] + 1,
                dp[i - 1][j - 1] + (word1[i - 1] == word2[j - 1] ? 0 : 1)
             );
          }
        }
```

```
            }
    return dp[len1][len2];
    }


    void suggestCorrections(struct HashTable* ht, char *word) {
        printf("Suggestions for '%s':\n", word);

       for (int i = 0; i < HASH_TABLE_SIZE; i++) {
                struct Node* current = ht->table[i];
                while (current != NULL) {
                    int distance = levenshteinDistance(word, current->word);
                    if (distance <= 2) {
                        printf("%s (distance: %d)\n", current->word, distance);
                    }
                    current = current->next;
                }
            }
          }

    int main() {
        struct HashTable dictionary;
        for (int i = 0; i < HASH_TABLE_SIZE; i++) {
                    dictionary.table[i] = NULL;
                }

                // Load dictionary into hash table (replace with your dictionary file)
                char *dictionaryWords[] = {"example", "correct", "spelling", "words", "more", "here"};
      int dictSize = sizeof(dictionaryWords) / sizeof(dictionaryWords[0]);

                for(int i = 0; i < dictSize; i++) {
                    insertIntoHashTable(&dictionary, dictionaryWords[i]);
                }
                FILE *textFile = fopen("textfile.txt", "r");

                if (textFile == NULL) {
                    printf("Error opening file.\n");
                    return 1;
                }
```

```
            char word[MAX_WORD_LENGTH];

          while (fscanf(textFile, "%s", word) != EOF) {
                  toLowerCase(word);
        if (!isInHashTable(&dictionary, word)) {
           printf("Misspelled word: %s\n", word);
           suggestCorrections(&dictionary, word);
        }
     }

     fclose(textFile);
  // Free allocated memory in the hash table
     for (int i = 0; i < HASH_TABLE_SIZE; i++) {
        struct Node* current = dictionary.table[i];
        while (current != NULL) {
           struct Node* temp = current;
           current = current->next;
           free(temp);
        }
     }
     return 0;
  }
```

5 RESULTS AND DISCUSSION:

```
textfile.txt
C is an imperative procedural languag,supporting
structured programing, lexical variable scope, and recursion, with a static type system.
 It was desined to be compiled to provid low-level access to memory and language constructs
that map efficiently to macine instructions, all with minimal runtime support.
Despite its low-level capabilities, the language was designed to encouage cross-platform programming.
A standards-compliant C program written with portability in mind can be compiled for a
wide variety of computer platforms and operating systems with few changes to its source code.
Missspelled words:languag
Suggestion for languag:language
Missspelled word:programing
Suggestion for programing:programming
Missspelled word:desined
Suggestion for desined:designed
Missspelled word:provid
Suggestion for provid:provide
Misspelled word:macine
Suggestion for macine:machine
Missspelled word:encouage
Suggestion for encouage:encourage
--------------------------------
Process exited after 0.08741 seconds with return value 953
Press any key to continue . . .
```

6 CONCLUSIONS:

- Spell check identifies and corrects misspelled words. It also allows you to search a document yourself for words you know you've misspelled.
- In Microsoft Word, spell check options, like spelling and grammar may be found under the 'review' tab and 'proofing' window. The system checks whether a given word is in the dictionary. If it cannot find the word in the dictionary, it will give correct word suggestions from the words in the dictionary. If the word is misspelled or is not likely to be found, the necessary information will be provided