Dictionary Application, Summer 2017

Oprea Denisa

September 2, 2017

Teachers: Costin Bădică and Alex Becheru
Section: Computers with teaching in Romanian Class
Group: C.R. 1.2.
Year of study: I

1 Problem statement

Write a dictionary application. The dictionary should allow for inserting a word, updating the denition of an existing word and looking up denition of a word. Note that a word may have multiple denitions attached to it. The dictionary should use les for storing and loading its data.

2 Pseudocode

```

add_word (struct dictionary *dictionary, char word[100]) {

    struct dictionary *iterator = dictionary;
    struct dictionary *newWord = malloc (sizeof (struct dictionary));
    char yon;
    char definition[100];

    while (iterator->nextWord != NULL) {
        if (strcmp (iterator->nextWord, word) == 0) {
            printf("\nWord already exists.");
            return;
        }
        iterator = iterator->nextWord;
    }

    iterator->nextWord = newWord;
    newWord->nextWord = NULL;
    strcpy(newWord->word, word);
    printf("\nYou've added a new word \"%s\" do you want to add a definition(Y/N)? ", word);
    yon = getch();
    if (yon == 'n' || yon == 'N')
        return;
    printf("\nEnter the word first definition: ");
    scanf("%s", &definition);
    strcpy(newWord->definition[0], definition);

}


 add_definition (struct dictionary *dictionary, char word[100], char definition[100]) {

```

```
    int i;
    struct dictionary *iterator = dictionary;

    while (iterator->nextWord != NULL) {
        if (strcmp (iterator->nextWord->word, word) == 0) {
            for (i = 0; strlen(iterator->nextWord->definition[i]) != 0; i++);
            strcpy(iterator->nextWord->definition[i], definition);
            printf("\nDefinition added successfully.");
            return;
        }
        iterator = iterator->nextWord;
    }
    printf("\nGiven word does not exists in dictionary.");
}


void save_to_file(struct dictionary *dictionary){




    FILE *f = fopen("dictionary.txt", "a+");

    if (f == NULL)
    {
        printf("\nError saving to file!");
        return;
    }

    int i;
    struct dictionary *iterator = dictionary;

    while (iterator->nextWord != NULL) {
        fprintf(f, "\n\nWord: %s", iterator->nextWord->word);
        for (i = 0; strlen(iterator->nextWord->definition[i]) != 0; i++)
            fprintf(f, "\nDefinition %d: %s", i + 1, iterator->nextWord->definition[i]);
        fprintf(f, "\n");
```

```
        iterator = iterator->nextWord;
    }
    fclose(f);

    printf("\nYour content has been saved to file \"%s\".", "dictionary.txt");
}
```

2.1 Pseudocode description

The **add_word** function is for adding a new word to our dictionary. It accepts 1 parameter, the name of the word we will add.

The **add_definition** function is for adding a new definition to an existing word.

The **save_to_file** is for saving the current dictionary to a file.

3 Application Design

3.1 Main

The **main** of my program contains a **while** loop so the user will be forced to choose a valid option. He has the option to choose from ten different options.

3.2 Input Data

For my program, input data are "decision", "decision_1", "decision_2", "decision_3"."decision" is the choice that you have to make in order to choose an option from the menu and in some cases will be overwritten with something else, "decision_1" is used for telling the word that will be added or the word we search for. "decision_2" variable is used to tell the definition we will add or the definition we search for. "decision_3" is used to tell the new definition in order to change an old one.

3.3 Output Data

The data outputs resulted from functions processing. The functions include adding, deleting, changing a word or a definition from our dictionary.

3.4 Functions used

void add_word (struct dictionary *dictionary, char word[100]) function is for adding a new word to our dictionary. If the user want he can immediately add a new definition.

void add_definition (struct dictionary *dictionary, char word[100], char definition[100]) function is for adding a new definition to an existing word.

void change_word (struct dictionary *dictionary, char word[100], char new_word[100]) function is used to find the given word and replace it with the one user specified.

void change_definition (struct dictionary *dictionary, char word[100], char definition[100], char new_definition[100]) function is used to find the given word and definition, will replace definition with the one user specified.

void show_all_words (struct dictionary *dictionary), function is used to print all words and their definitions to the console.

void show_all_definitions (struct dictionary *dictionary, char word[100]) function is used to print all definitions of a given word to the console.

void delete_word (struct dictionary *dictionary, char word[100]) function is used to delete a given word from the dictionary.

void delete_definition (struct dictionary *dictionary, char word[100], char definition[100]) function is used to delete a given definition of a given word.

void save_to_file (struct dictionary *dictionary) is for saving the current dictionary to a file under the same folder.

4 Source code

My program is called "Dictionary Application". It is created in C99 standard.

The code is compiled with the following compiler: "GNU GCC Compiler"

5 Experiments and results

5.1 GCC Compiler

For GCC compiler here are the results of running the application.

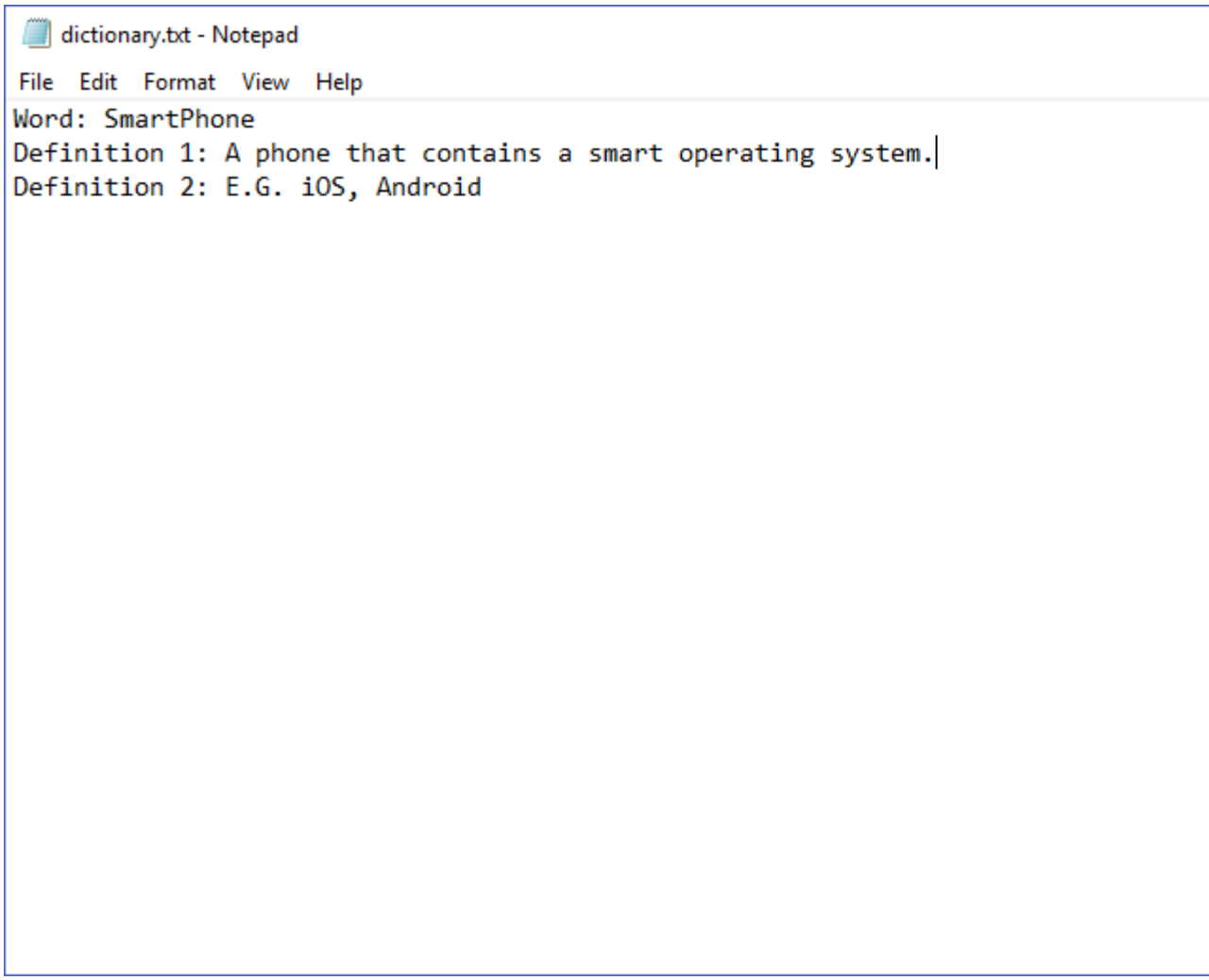
It runs as follow:

```
          Menu
1.Add a word to the dictionary
2.Add a definition for a word
3.Show definitions of a word
4.Show all dictionary
5.Delete a word.
6.Delete a definition of a word
7.Change word
8.Change definition of a word
9.Print dictionary to a file
10.Exit

Chose your option: 1

Enter the word you want to add: SmartPhone

You've added a new word "SmartPhone" do you want to add a definition(Y/
Enter the word first definition: A phone that contains a smart operatin
```

```
Word: SmartPhone
Definition 1: A phone that contains a smart operating system

         Menu
1.Add a word to the dictionary
2.Add a definition for a word
3.Show definitions of a word
4.Show all dictionary
5.Delete a word.
6.Delete a definition of a word
7.Change word
8.Change definition of a word
9.Print dictionary to a file
10.Exit

Chose your option: 2

Enter the word for which you will insert a new definition: SmartPhone

Enter the definition: E.G. iOS, Android
```

```
Chose your option: 4


Word: SmartPhone
Definition 1: A phone that contains a smart operating system
Definition 2: E.G. iOS, Android

         Menu
1.Add a word to the dictionary
2.Add a definition for a word
3.Show definitions of a word
4.Show all dictionary
5.Delete a word.
6.Delete a definition of a word
7.Change word
8.Change definition of a word
9.Print dictionary to a file
10.Exit

Chose your option:
```

```
Enter the word you want to delete: SmartPhone

Word "SmartPhone" successfully deleted.
         Menu
1.Add a word to the dictionary
2.Add a definition for a word
3.Show definitions of a word
4.Show all dictionary
5.Delete a word.
6.Delete a definition of a word
7.Change word
8.Change definition of a word
9.Print dictionary to a file
10.Exit

Chose your option: 4


Word: Phone
Definition 1: A device that allows you to communicate through long dist
```


dictionary.txt - Notepad
File Edit Format View Help
Word: SmartPhone
Definition 1: A phone that contains a smart operating system.
Definition 2: E.G. iOS, Android

6 Conclusions

Using linked list was a great because of the performance. Code is simple and easy to understand because i used linked lists instead of matrices.

References

[1] **Robert I. Pitts**
https://www.cs.bu.edu/teaching/cpp/string/array-vs-ptr/