T.C.

DOKUZ EYLUL UNIVERSITY

ENGINEERING FACULTY

ELECTRICAL & ELECTRONICS ENGINEERING DEPARTMENT

Word Recognition and Correction Algorithm and Application

By

Atakan Ertugrul

Beyza Karaca

Enes Erten

Fatih Korkmaz

Kutlu Uzay Yenidogan

Advisor

Dr. Abdul Balikci

May 2020

Abstract

Nowadays, everything is searched in Google. Google receives over 63.000 searches per second averagely, it makes 5.6 billion per a year also it is increasing exponential. Each of us are making mistakes while entering anything to the search engine. Search engine must recognize the sentence and correct the mistakes to find true results. That is the reason of why this project trying to create a simple and basic recognition and matching algorithm.

This project has four part. Firstly, it creates a word database. Secondly, it gets the word's all combination. Thirdly, it searches the word in database to find the closest word. Fourthly, it displays the result.

The word database has 8 files. These are one_word, two_word, three_word, four_word, five_word, six_word, seven_word, and common_word files. The group members decided to create maximum seven word file because of the limited time. In the database, there are thousands of words. However, it can be expanded any time.

When user is entering a word to system, algorithm creates an array and assigns to all combinations. That is the most important part of the algorithm due to user could forget a letter or press more letter. If we need to get true match we have to search all combinations. At the future algorithm can be expanded.

Third part of algorithm counts how many common words have matched with data words in the Database. The biggest count will be the matching.

Fourth part of algorithm displays the result of matching.

The cost of this project is 0 $ because of no need to any equipment except personal computers.

Key words: Recognition, Matching, Algorithm, Flowchart, File, Database.

TABLES OF CONTENTS

1. EXECUTIVE SUMMARY

Firstly there is a need to introduce what is search engine. According to the Cambridge Dictionary [1] definition, it is a computer program which finds information on the internet by looking for words that you typed in [2]. The most widely used search engine is absolutely GOOGLE [3] search engine. Most people make mistakes while entering words and sentences to the search engines. These mistakes can be caused wrong results and this situation affects them life very harder. However, search engines use some algorithms to fix these mistakes. For example, GOOGLE uses Did you mean [4] algorithm. Basically, this algorithm has two parts. First part is recognition the word or words. The second part is correction the word or words. If there is no word recognition and correction algorithm, everyone spends more time with trying to correct what they wrote. Also this case were caused spend more electrical energy. And also that were caused emit more CO_2 and pollute environment more than today. That algorithm is smart, simple and basic. However, it has countless benefits in the other hand.

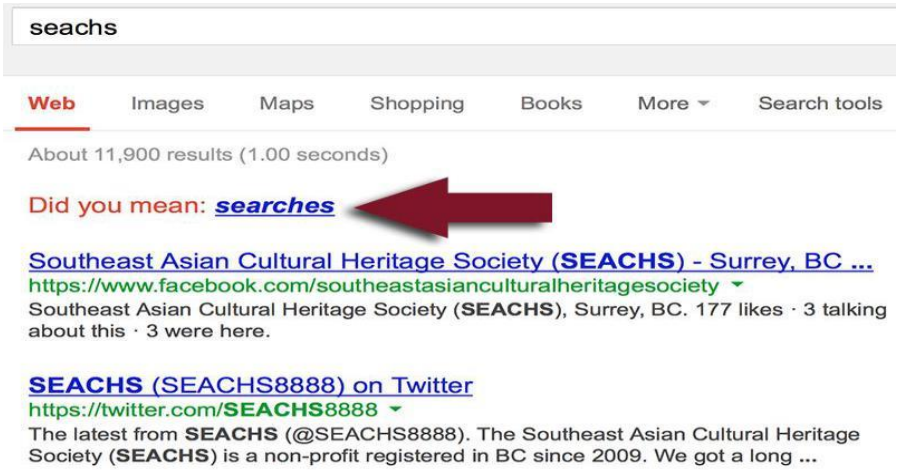


Figure 1

In the big picture, recognizing and correcting algorithms become very popular and this makes human's life easier as it can be seen from the first paragraph. Also algorithms connect to Artificial Intelligence, and Machine Learning. Obviously, these technologies are changing and with no doubt, they will change the world either.

Python programming language [5] were used in this project because of five reasons. First, easy syntax. Second, useful commands. Third, flexible and useful functions. Fourth, useful and flexible data structures and functions. Fifth, the programming language is very useful to create project like this.

PyCharm [6] Compiler and Python shell were used [7] in this project because of the easy usage and free license.

2. STATEMENT OF PROBLEM

Time is an importance for people which cannot waste recklessly. The reason why technology was existed is this opinion. Humans want to save their time. Also they want to save their energy. With these desires, they invent and develop. So, trying to create something which practical and facilitate must be main goal for engineers.

In 21th century, with existence of internet and computer, people wasting their life in internet. They playing, watching, reading, surfing, writing and searching. This situation is a good opportunity to develop something useful for people which about internet.

Internet gives to people some opportunities. This opportunities include easy way to search something. When user enter what it want to learn about to search engine, a hundreds of searching results which connected to user's word was displayed on computer screen in seconds. These results depend on what user's write and it is an inevitable situation that some mistakes are occurred. Normally, user have to notices it's mistake and have to rewrite correctly what it wrote before. But another way can be possible. This way including a software. A kind of program which corrects mistakes in user's words that it entered the searching engine instead of user. With some proper codes this program can facilitate using of internet.

In conclusion, this project will be helpful to many internet users by understand them correctly and completely beside to other assistances which belong to internet.

3. OBJECTIVES

Actually, this project has 4 main part as it mentioned before. In addition, function must be included two this list.

1. Creating word database.
2. Creating essential functions which are length, compare_count, combination, permute, readfile_word.
3. Taking a word from user and get all combinations of word.
4. Compare the combinations of word with words in the database.
5. Displaying the result which is the most closest word.

The database was created manually by .dat files. The database was created just for the English language because of the limited time and fact of most spoken language is English around the world. The database has 8 words files.

1. one_letter.dat file includes one letter words.
2. two_letter.dat file includes two letter words.
3. three_letter.dat file includes three letter words.
4. four_letter.dat file includes four letter words.
5. five_letter.dat file includes five letter words.
6. six_letter.dat file includes six letter words.
7. seven_letter.dat file includes seven letter words.
8. common_words.dat file includes common words in English language.

After creating files, it reads from file and it assigns to the lists which are called same with files such as: one_letter_list, two_letter_list, three_letter_list, four_letter_list, five_letter_list, six_letter_list, five_letter_list, seven_letter_list, common_words_list. After that's, words can be used in the program.

The databases can be reached from APPENDIX A [1].

This program includes 4 function and main.

1. permute
2. combinations
3. length
4. compare_count
5. main

Permute function is a recursively function. It assigns result list for all permutations of string. This function's algorithm is simple. It uses swap operations to change letters and get permutations of string. For instance, "ABC" string will assigned to the ["ABC,"BCA","ACB","BAC","CAB","CBA"] result list. It can be seen clearly with flowchart.

- The flowchart of the permute function.

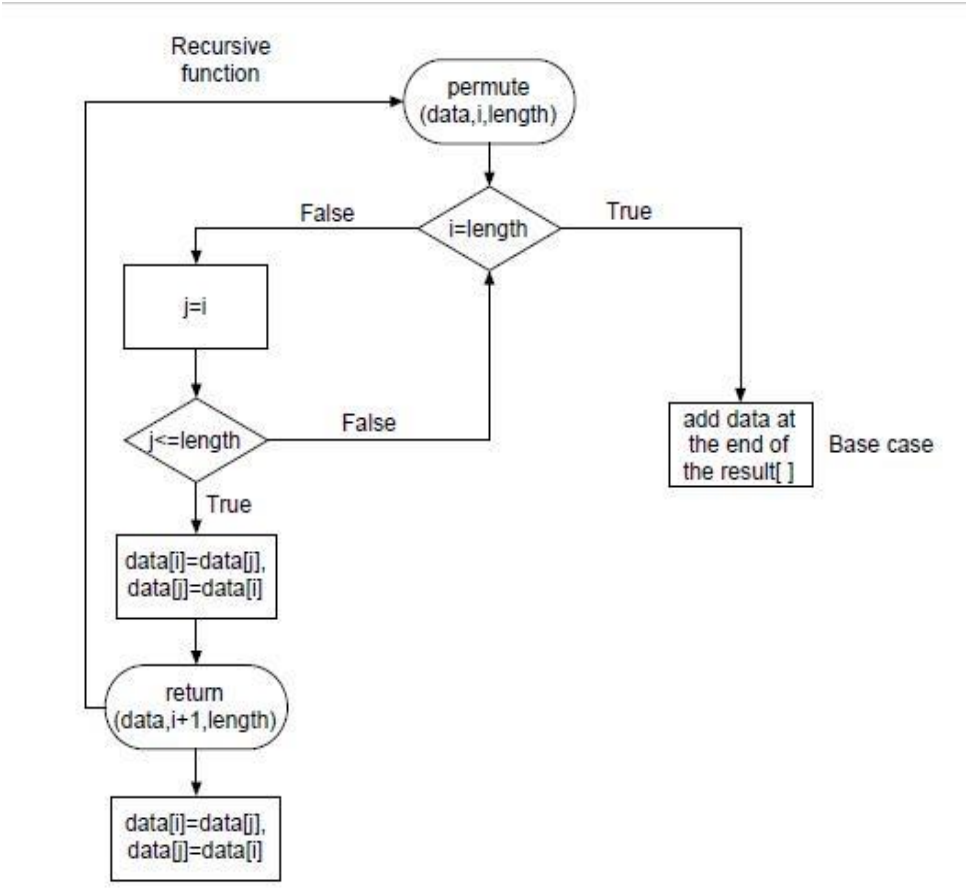


Figure 2

The second function is a combination function. This function is also a recursively function. It gets combinations of a string. For instance, if “ABC” string was sendt to function, it will returned ['', 'A', 'B', 'AB', 'C', 'AC', 'BC', 'ABC'] . If combinations and permutations were summed up, all combinations of string were came up.

- The flowchart of the combinations

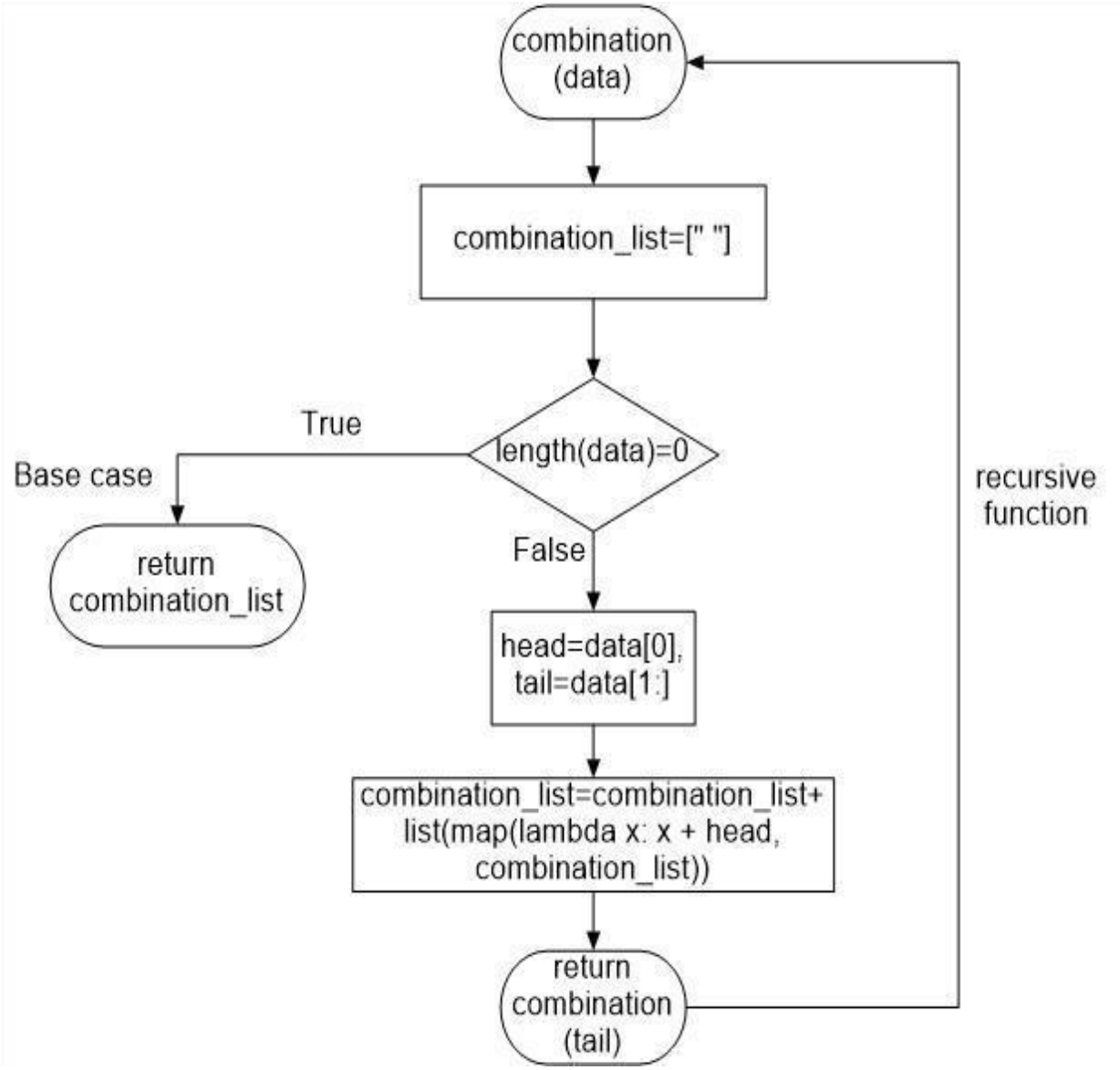


Figure 3

The length functions returns the length of list, string, etc. It is worked iteratively it is counting all the elements of string, list, etc.

- The flowchart of the length function

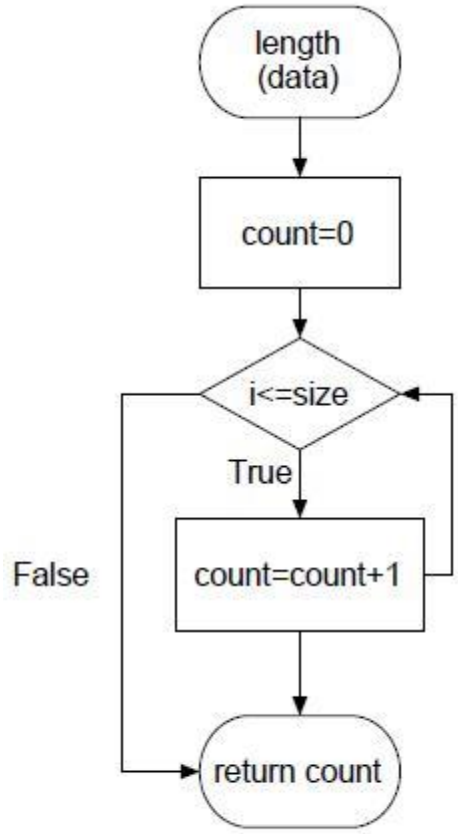


Figure 4

The compare_count function compares how many letters are same in two strings and it returns the number of the same letters.

- compare_count function flowchart

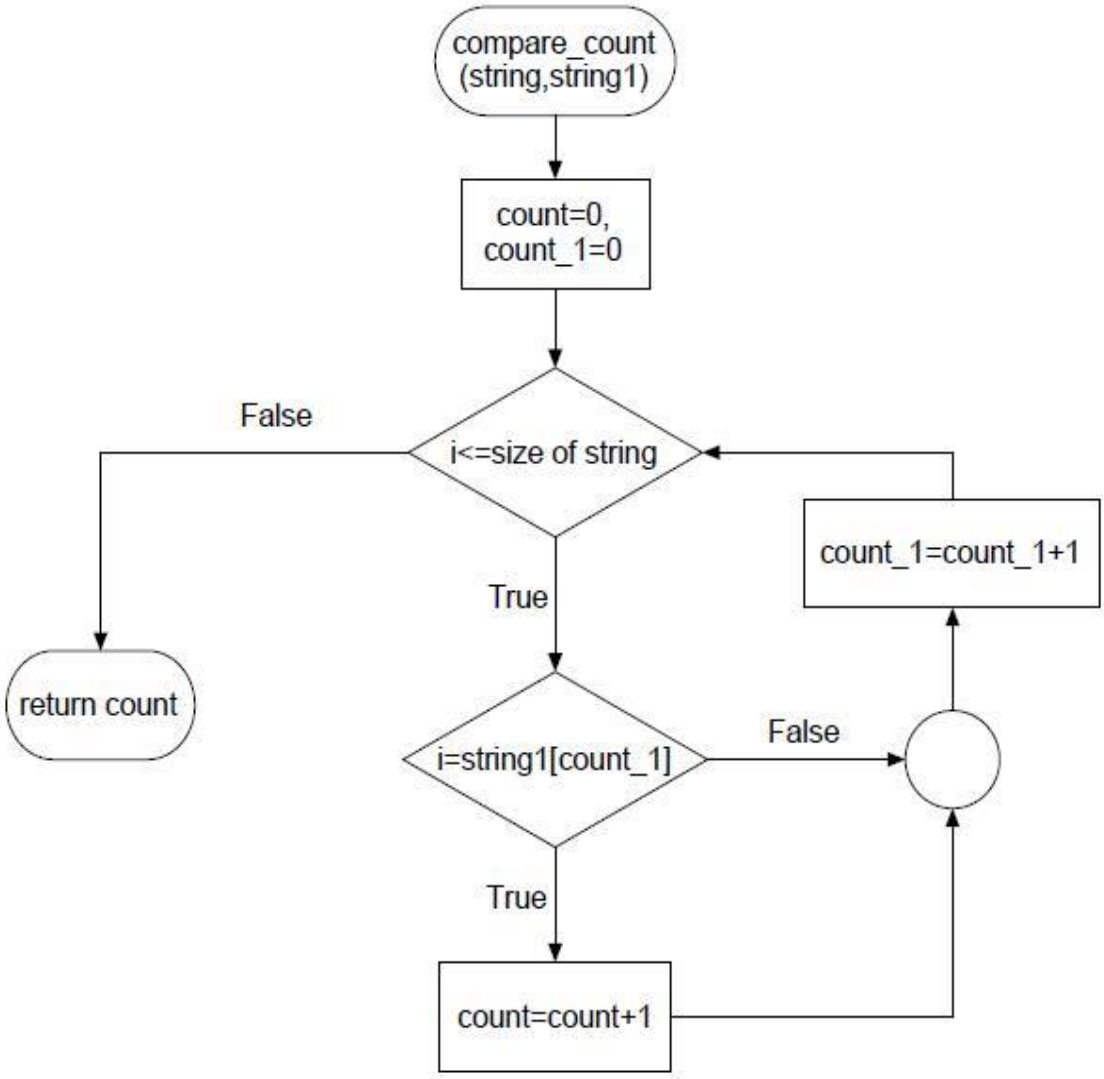


Figure 5

The main function is gets a list which include the combination and permute results. Then searches in the word database. At the end, it displays the results.

- The flowchart of the main function is in the APPENDIX B due to the it is large size it is seperated to five parts (part 1,2,3,4,5, and 5) and the whole flowchart (part 6).

4. TECHNICAL APPROACH

Nowadays, computers and software are becoming inseparable in lifetime. Almost all of the houses have more than one computer and have a number of software. Computers and software are making life easier.

Identifying Customer Needs

As it mentioned many times in the report, a little mistake about any word can be caused loss of time and energy. This amount of loss can be enormous. This project offers an algorithmic solution and creates the program of algorithm.

Identifiying Algorithm Specifications

As it mentioned many times in the report, the specifications of the algorithm is the recognize a word and try to match with the closest word in the database.

Generating Algorithm

Firstly, this project is needing thousands of classified words. Creating these words's database is the important pillar of this structure. These words were found in the Cambridge dictionary [1] and they were used in the classified in the .dat files. The classification is the length of the words's:

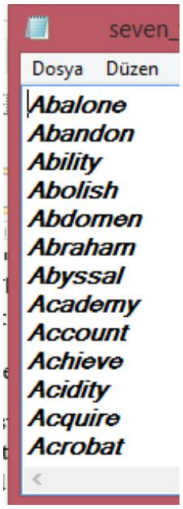


Figure 8

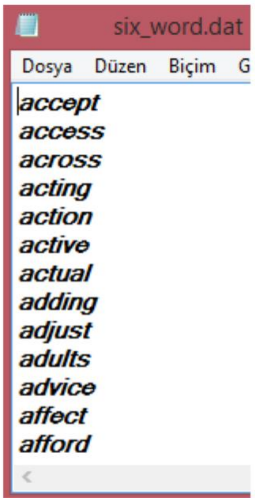


Figure 9

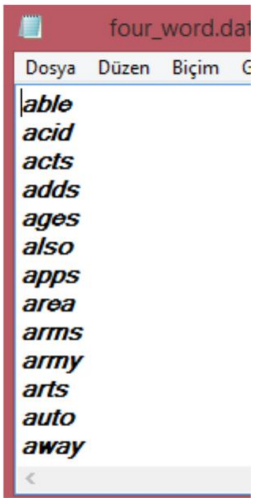


Figure 10

The data base can be reached on the APPENDIX A [1]

The permute function code can be find in [2] APPENDIX A. This function is getting a word all permutations. For instance, assume “ABC” string is send to function “ABC”, “ACB”, “BAC” , “BCA” , “CAB” , and “CAB” string will assign to a list.

The combination function code can be find in [3] APPENDIX A. This function is getting a word all combinations except permutations of the word. For instance, assume “ABC” string is send to the function, “ABC”, “AB”, “AC”, “BA”, “BC”, “CA”, “CB”, “A”, “B”, and “C” list will be returned.

The length function code can be find in [4] APPENDIX A. This function is returns the number of elements of a list. For instance, “ABC” string will send to the function, 3 will be return.

The compare_count function code can be find in [5] APPENDIX A. This function is returns the common letters of two words. For instance, assume “ABC” and “ABD” string will be sent to the function, 2 will be return due to, ‘A’ and ‘B’ are common and same location in the string.

The main function code can be find in [6] APPENDIX A. This function is taking a string from user (stdin) after that opens the database (common_words) and searchs in the file and gets the correction or most closest word in the database after that displays the word which is correction by the program.

In addition to that demo will be decide the word length and open read file of the word if word doesn’t found the program looks for different word data’s.

Selecting Algorithm

As it mentioned before all the algorithm is ready. However, the main function just need an upgrade. Lengths of the words and capital letter determinations will be upgraded.

5. PROJECT MANAGEMENT

Works/Months	MARCH		APRIL				MAY		
	3. WEEK	4.WEEK	1. WEEK	2. WEEK	3. WEEK	4. WEEK	1. WEEK	2. WEEK	3. WEEK
Formin the team									
First meeting with group members									
Determining the project steps									
Discussion about the ideas									
Research about the project									
Examining the codes									
Writing codes									
Entering data									
First program test									
Final version of the program									

Figure 10: Gantt chart for the project. Painted areas show that the job has been done successfully.

Deliverables

This program provides convenience for anyone who wants to research any phrases at the Internet. The feature of this program is translating the typed word into the closest and most meaningful word to the user's write and return to the user with the correct word. This condition provides great convenience for internet users.

Budget

There is no need any budget because of program was produced and used digitally. This situation is one of the most useful feature of program. The amount of money which spent on making such a convenience is zero.

Team Qualifications

In this section qualifications and duties of team members are stated.

Atakan Ertugrul **:** Dokuz Eylül University
Electric and Electronic Engineering
A student from first grade

- Made the distribution of tasks in project.
- Wrote the sections Executive Summary and Conclusion in report.
- Edited the references section.
- Improve some parts of report.

Beyza Karaca **:** Dokuz Eylül University
Electric and Electronic Engineering
A student from first grade

- Determined how the project would proceed.
- Wrote Statement of Problem and Technical approach sections.
- Translated some article about project.
- Improving some parts of report

Enes Erten **:** Dokuz Eylül University
Electric and Electronic Engineering
A student from first grade

- Organized the team.
- Wrote the main code.
- Wrote the functions.
- Wrote the sections Abstract in the report.

Fatih Korkmaz **:** Dokuz Eylül University
Electric and Electronic Engineering
A student from first grade

- Arranged the group member's second online meeting.
- Wrote the Statement of Problem and Project Management parts.
- Improve some parts of report.

Kutlu Uzay Yenidogan : Dokuz Eylül University
Electric and Electronic Engineering
A student from first grade

- Arranged the first meeting of the group members.
- Wrote the sections Objectives in the report.
- Helped the code writing phase.
- Drawn the flowchart in report.

6. CONCLUSION

There are a number of words which are written in search engine. Sometimes they are written wrong and sometimes they are written properly.

Google developed an algorithm called “Did You Mean” in case of writing a wrong word in search engine. This algorithm is a genius code and it need to develop and examination. This project is trying to describe “Did You Mean” algorithm.

Writing this algorithm’s code in C language is quite difficult so using Phyton is a more sensible way in this situation. Selecting a project about programming and algorithms will be more beneficial for freshmen students and this way is more cheap so everyone of us agreed in this project. The main goal is find a solution that prevent to loss of time and energy. This project is a kind of program that recognize word and correct them if there is a mistake.

7. REFERENCES

[1] Cambridge Dictionary, [online] [Citied May 25.05.2020] World Wide Web: https://dictionary.cambridge.org/

[2] Cambridge Dictionary, [online] [Citied May 25.05.2020] World Wide Web: https://dictionary.cambridge.org/dictionary/english-turkish/search-engine

[3] Google Search Engine, [online] https://www.google.com/

[4] Google , [online]https://www.google.com/search?sxsrf=ALeKk02HraoUYJCns1gYVpFCYpk7CKFgWA%3A1590143224109&ei=KjHXqmvBoeWaOK0u8AK&q=seaches&oq=seaches&gs_lcp=CgZwc3ktYWIQAzIECMQJzIICAAQBxAKEB4yCAgAEAcQChAeMgQIABAKMgIIADIECAAQCjIECAAQCjIECAAQCjIECAAQCjoECAAQRzoGCAAQBxAeOgUIABDLAToFCAAQkQI6BwgAEBQQhwJQmB9YvDpgokpoAHABeACAAboBiAGlBJIBAzAuM5gBAKABAaoBB2d3cy13aXo&sclient=psy-ab&ved=0ahUKEwipi4QocfpAhUHCxoKHWLaDqgQ4dUDCAw&uact=5

[5] Python programming Language**,** https://www.python.org/

[6] PyCharm Compiler, https://www.jetbrains.com/pycharm/, ver. 3.8.3, JetBrains

[7] Python Shell, https://www.python.org/downloads/, ver. 3.8.3 , Python

8. APPENDIX A

[1] Word database in Github Environment
https://github.com/EnesErten/Spellcheckalgorithm/blob/master/two_words.dat

https://github.com/EnesErten/Spellcheckalgorithm/blob/master/three_words.dat

https://github.com/EnesErten/Spellcheckalgorithm/blob/master/six_word.dat

https://github.com/EnesErten/Spellcheckalgorithm/blob/master/seven_word.dat

https://github.com/EnesErten/Spellcheckalgorithm/blob/master/one_word.dat

https://github.com/EnesErten/Spellcheckalgorithm/blob/master/four_word.dat

https://github.com/EnesErten/Spellcheckalgorithm/blob/master/five_words.dat

https://github.com/EnesErten/Spellcheckalgorithm/blob/master/common_words.dat

[2] The permute function:

```
def permute(data, i, len):
    if i == len:
        # base case
        result.append(''.join(data))
        # append the list

    else:
        # swap operation

        for j in range(i, len):
            # swap values
            data[i], data[j] = data[j], data[i]
            # recursively call
            permute(data, i + 1, len)
            # after recursively call
            data[i], data[j] = data[j], data[i]
```

[3] the combinations functions:

```
def combination(data, combination_list=[""]):
    if length(data) == 0:
        # base case
        return combination_list
        # return comblist
    head, tail = data[0], data[1:]
    # head tail head is keeping the firs element of the data
    # tail is keeping the rest of string
    combination_list = combination_list + list(map(lambda x: x + head, combination_list))
    # shuffling the words
    return combination(tail, combination_list)
    # recursively call
```

[4] the length function :

```
def length(data):
    # create a counter initialize to 0
    count = 0
    # for loop loops data times
    # make data to list
    for i in list(data):
        # count the elements of list(data)
        count += 1

    # return the count length of the string
    return count
```

[5] the compare_count:

```
def compare_count(string, string1):
    # import library collections
    # use Counter functions
    from collections import Counter

    return sum((Counter(string) & Counter(string1)).values())
    # return the number of common letters
```

[6] the main function:

```
        if __name__ == "__main__":
            result = []
            datas = []

            # assign ch ""
            ch = ""
            # open file common_words.dat
            with open("common_words.dat", "r") as file:
                # loops 1000 times
                for i in range(1001):
                    # read one line
                    ch = file.readline()
                    # take len-1
                    ch = ch[:-1]
                    # append to the list
                    datas.append(ch)

            # take string from user
            word = input("Enter a word\n>>")
            # send to the function
            comb = combination(word)
            # send to the function
            permute(list(word), 0, length(word))
            # get all combinations
            all_comb = result
            # assign 0 and ""
            c = 0
            c_st = ""

            # for loop, loops datas
            for k in datas:
                # for loop, loops all_comb
                for j in all_comb:
                    # if statement is true
                    if c < compare_count(j, k):
                        # update c value
                        c = compare_count(j, k)
                        # update c_st
                        c_st = k

            # display the result
    print("Did you mean : ", c_st)
```

9. APPENDIX B

Sample pictures of the databases

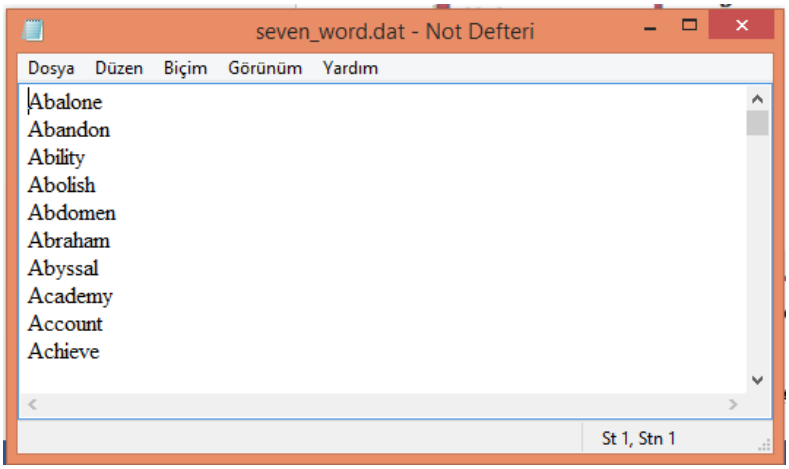


Figure 8

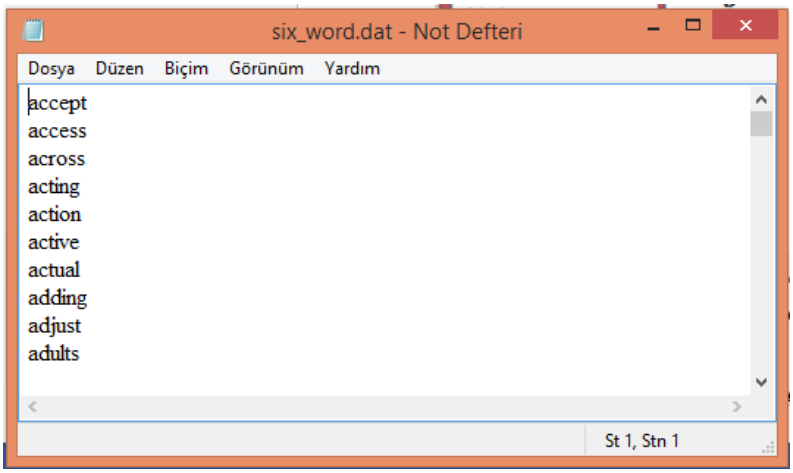


Figure 9

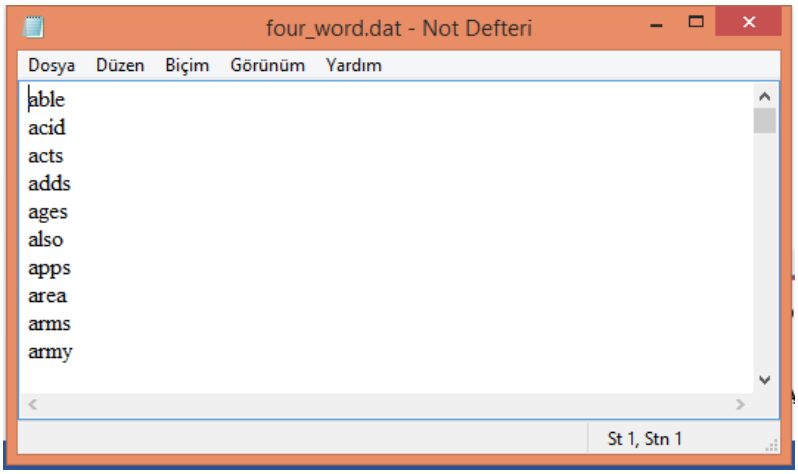
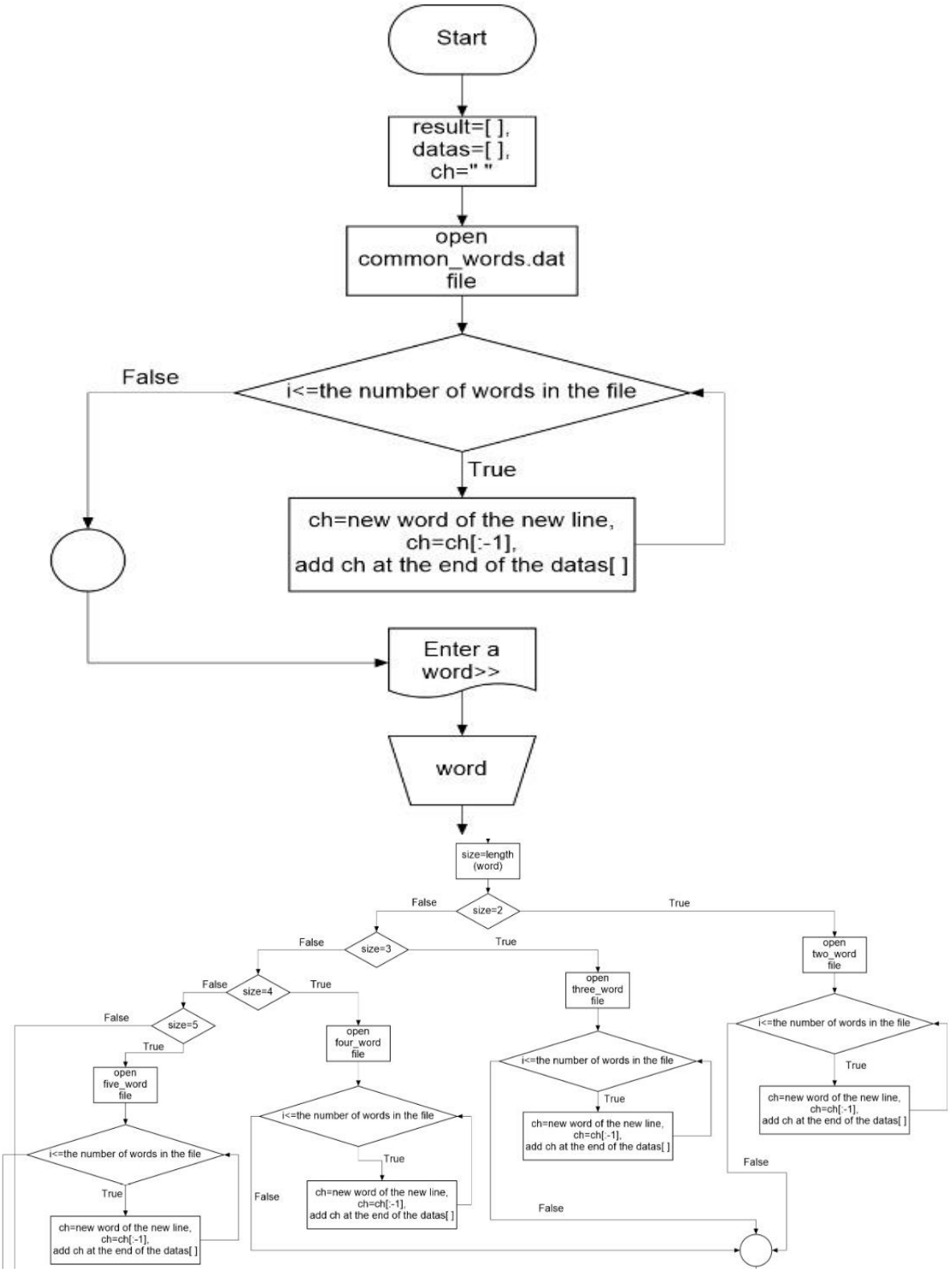
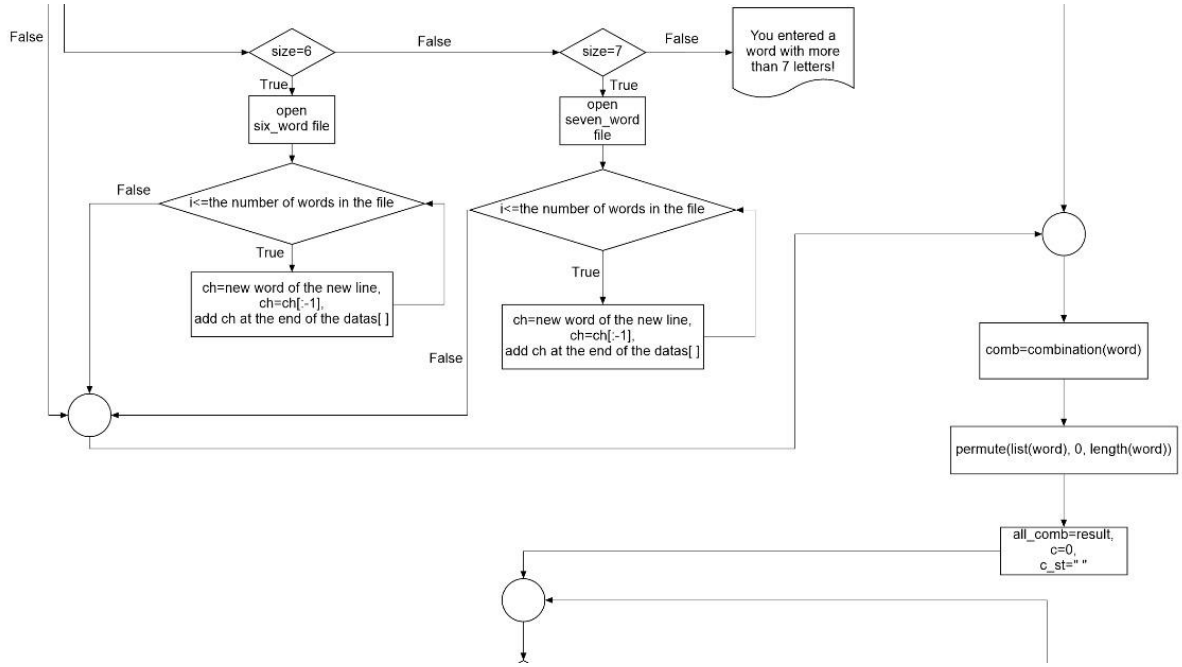


Figure 10

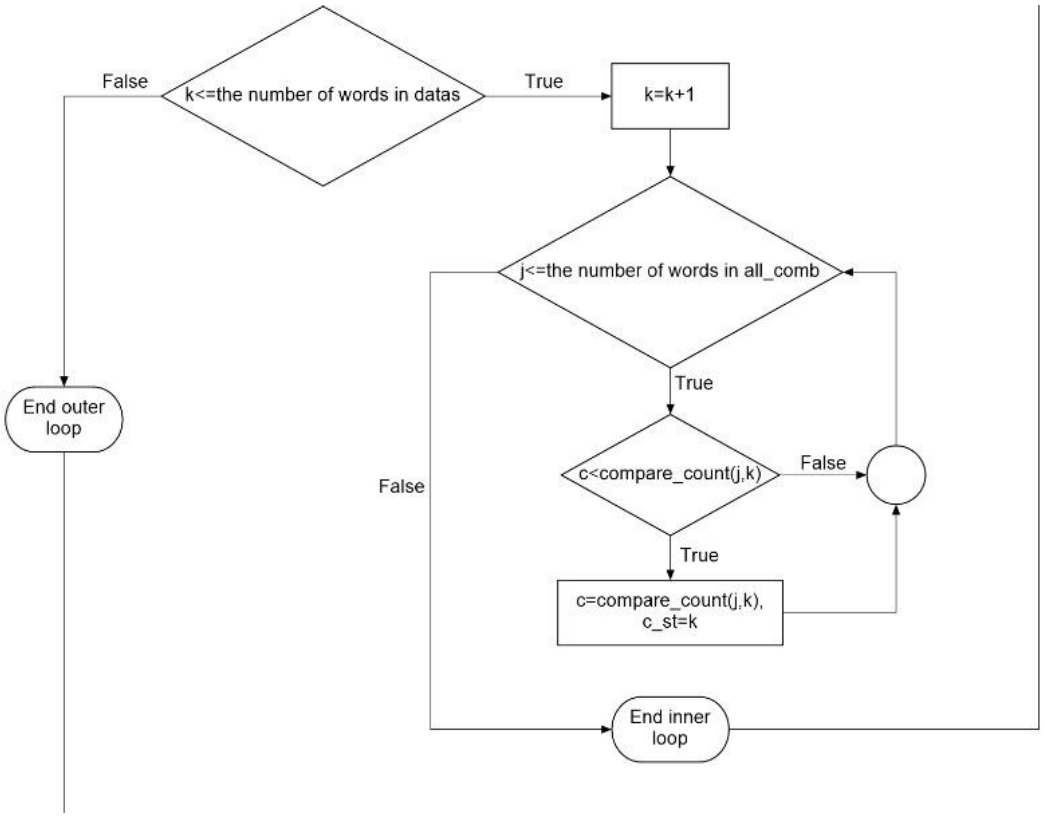
The flowchart of the main function part 1



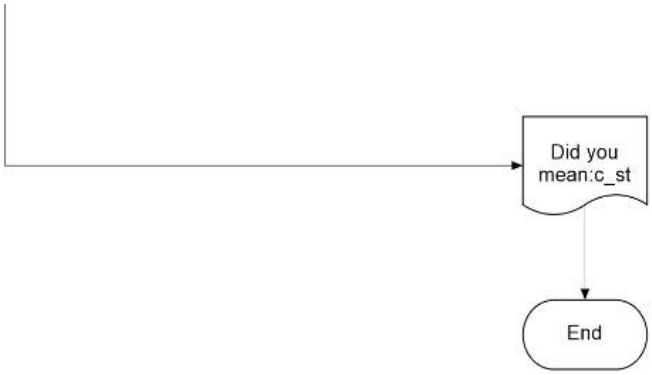
The flowchart of the main function part 2



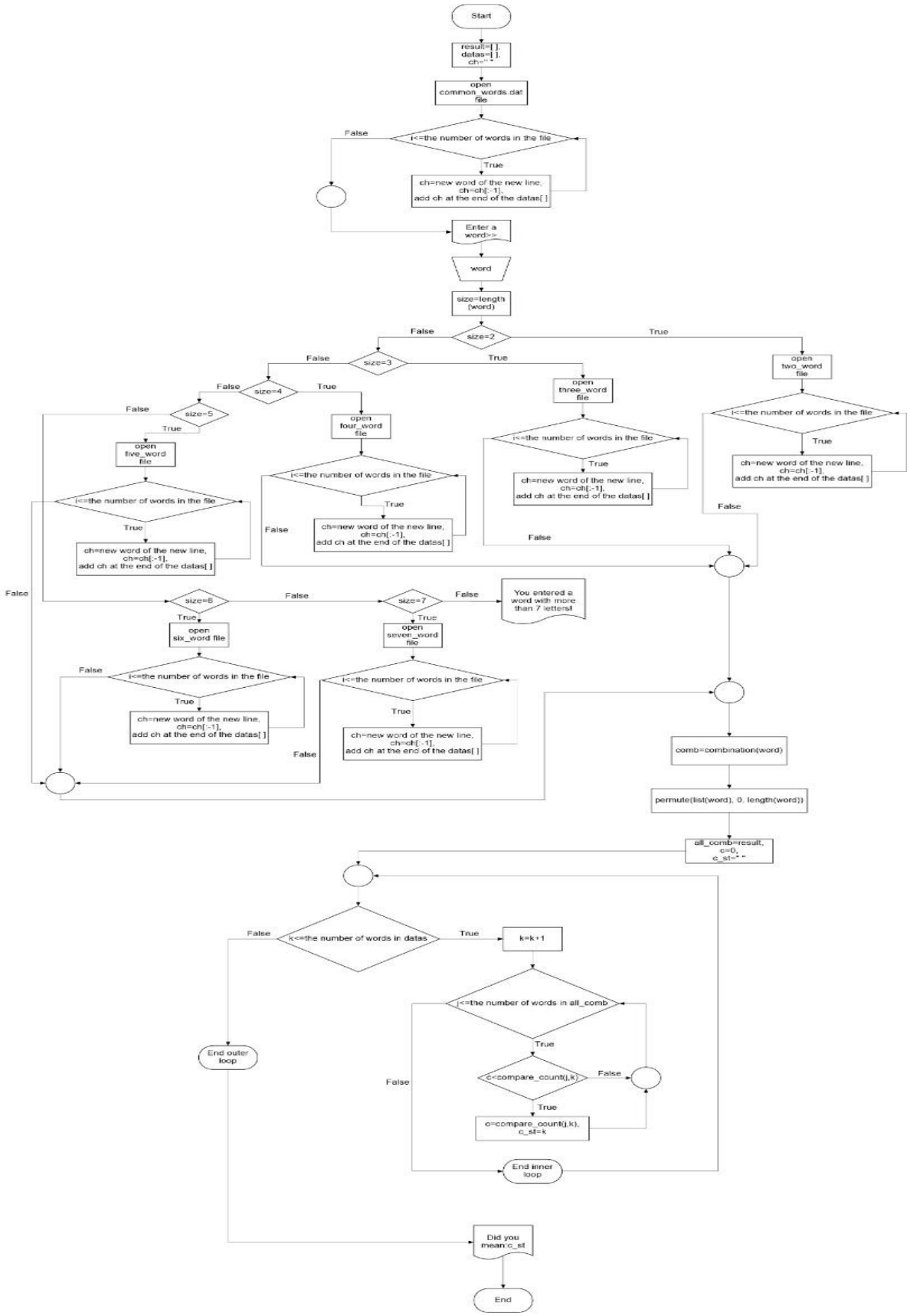
The flowchart of the main function part 3



The flowchart of the main function part 4



The flowchart of the main function part 5



The flowchart of the main function part6