|  |  |
| --- | --- |
|  | Team members:    Abrar Ul Abdin |

Introduction:

Our aim of developing this project is to utilize and furnish our programming skills we learned during data structures course throughout this semester. A dictionary is something that a person of every age group requires and so we chose this as the main concept of our project.

Background:

The basic thing which we focused on the most while selecting this project was that it needed to be unique and different from all others and that it the reason why the pre-processing of this project took more time than required. After selecting the project, we worked on the most suitable data structure for our dictionary project and so we came across a new data structure that was not taught in the class. The data structure was “TRIE”. We learnt its working and functionality and started building our project. The reason for choosing “TRIE” data structure is that [Trie](http://en.wikipedia.org/wiki/Trie) is an efficient information reTRIEval data structure. Using Trie, search complexities can be brought to optimal limit i.e. the (key length).

Project Specifications:

Our project is a word dictionary which has been developed using the c++ programming language. It can perform a variety of functions such as inserting new words into the dictionary, deleting words, looking for a specific word, printing all of the dictionary’s words in alphabetical order, finding synonyms for a word and finally the most distinct of all, a prefix search where if you enter only the initials of a word, the program displays all the related words which you could be searching for and those starting with the given initials.

Problem Analysis:

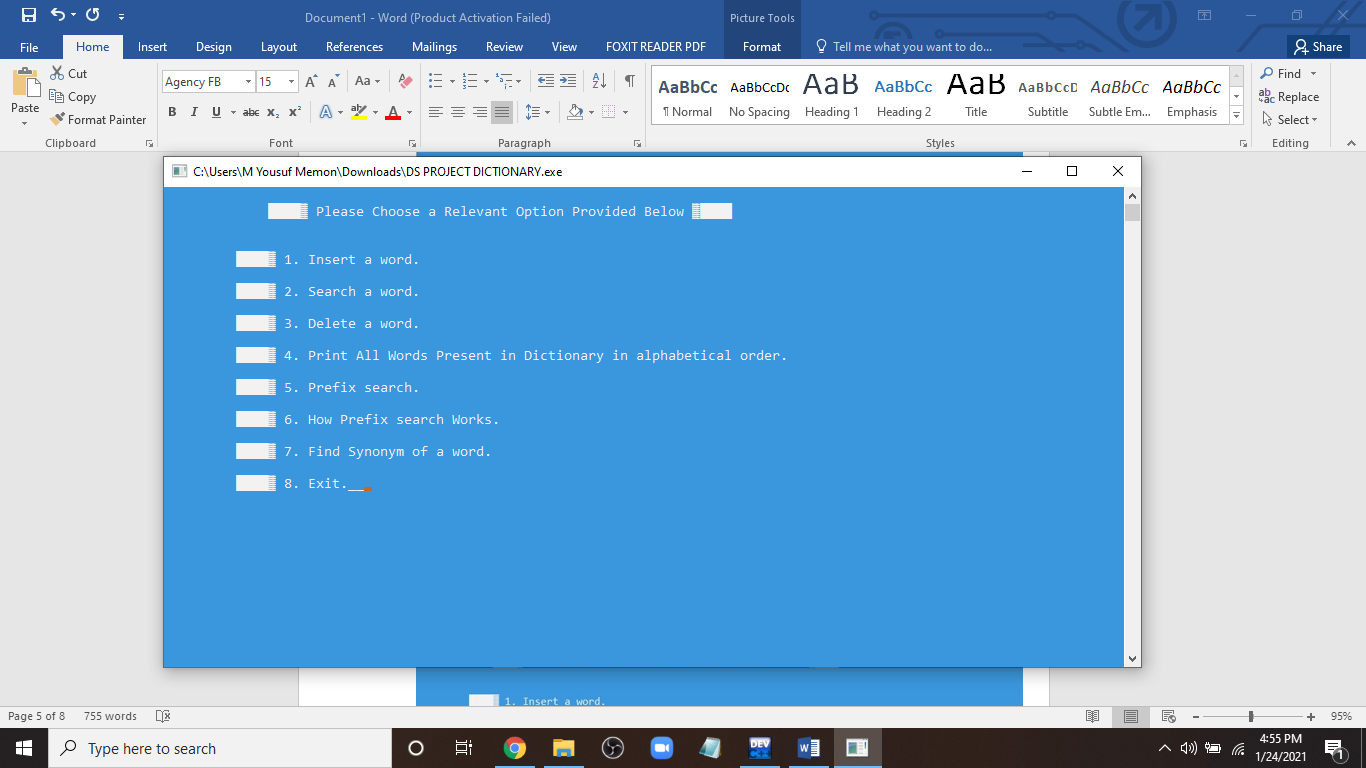
The only problem we faced in making this project was coming up with distinct functions i.e the *prefix search* and *synonyms* functionality because making a word dictionary is a common project which many students opt for. We had to search for many days online and have many discussions to finally come up with these unique ideas.

Solution Design:

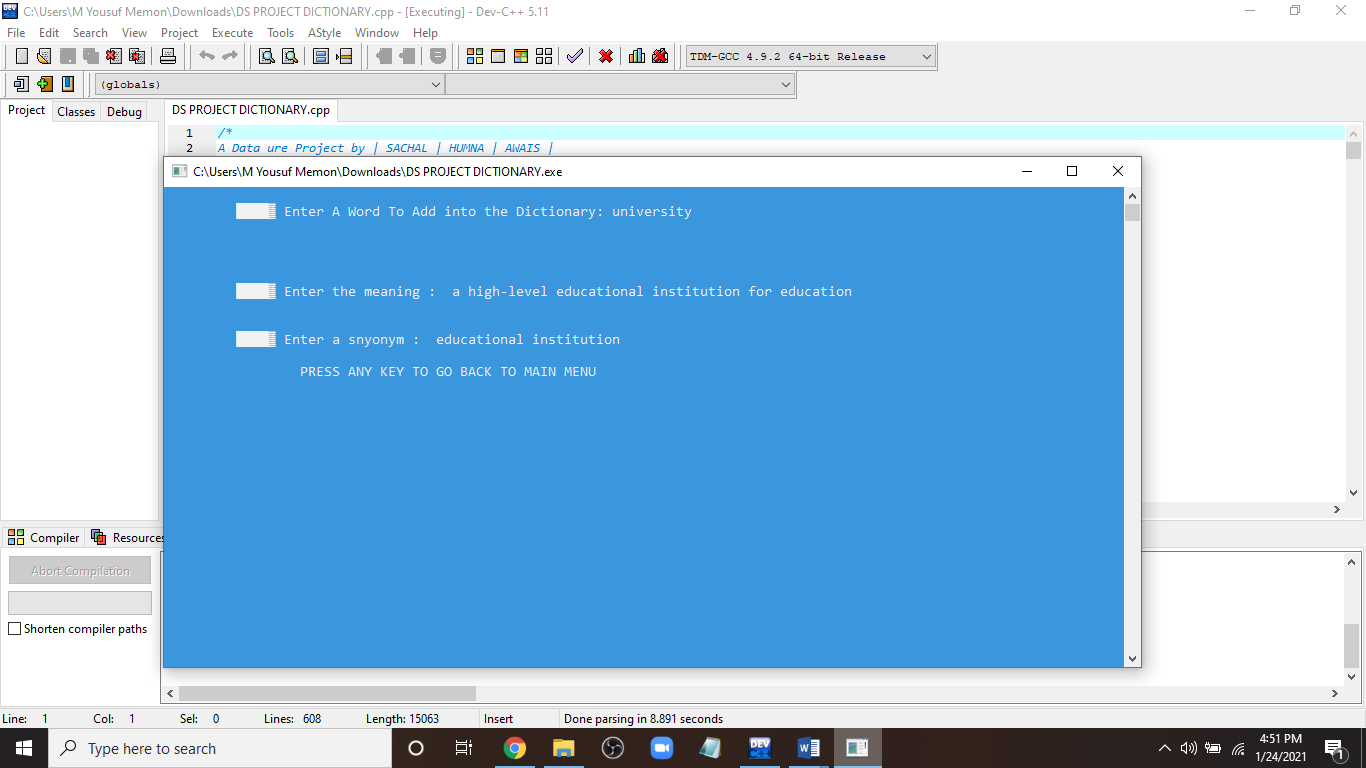
All of the work done is a group effort and we did not divide our workload as in the end it gets hard to compile all the different logics together, but rather thought and worked together, mostly on zoom because of covid-19 SOPs. Working together helped us a great deal and also helped us save time.

Implementation and Testing:

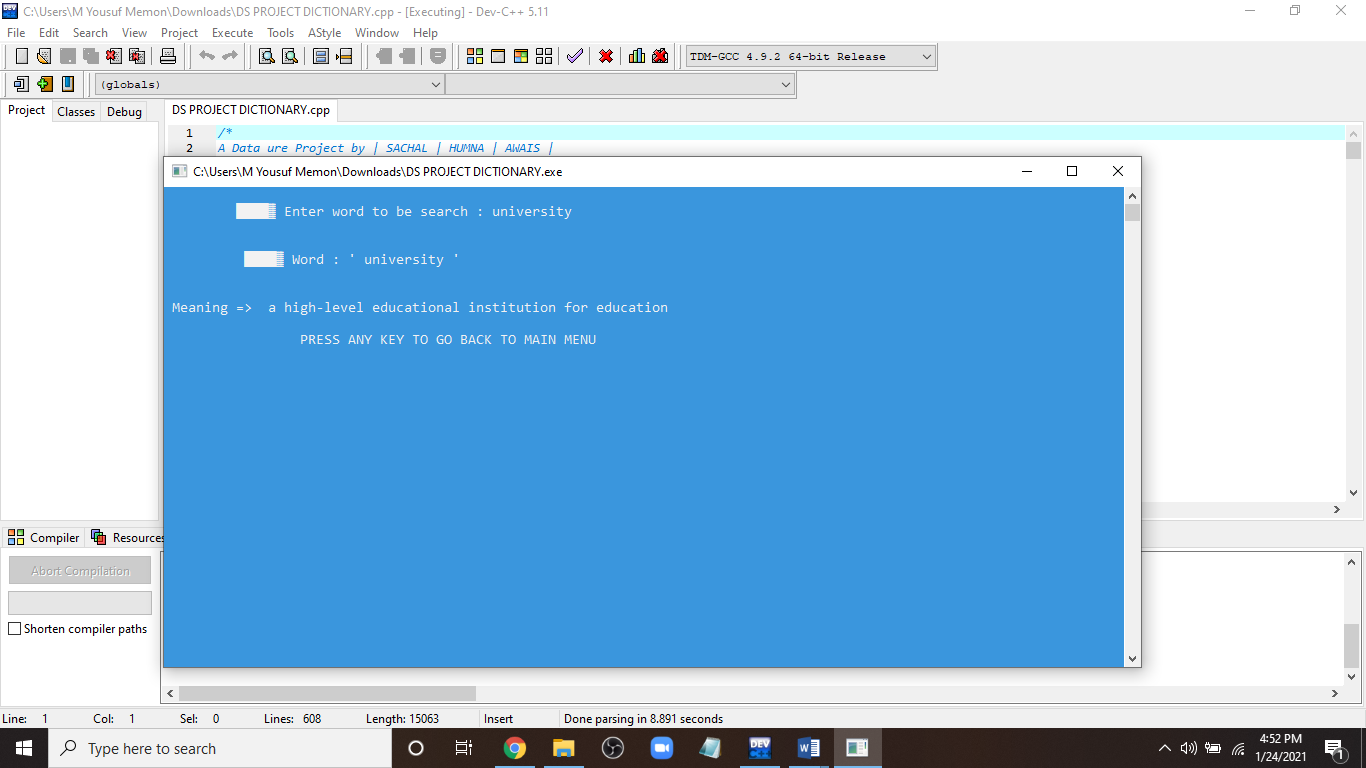
1. MENU :



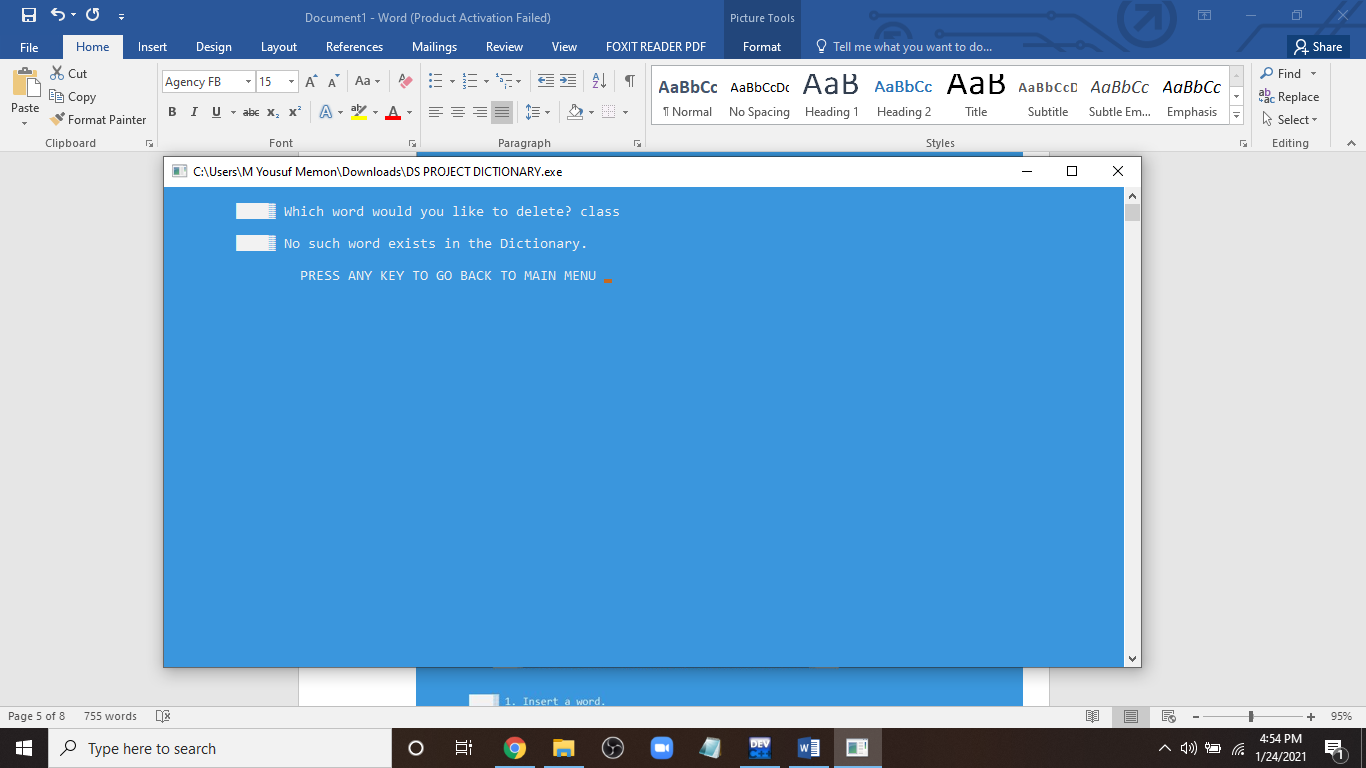
1. Adding a word :



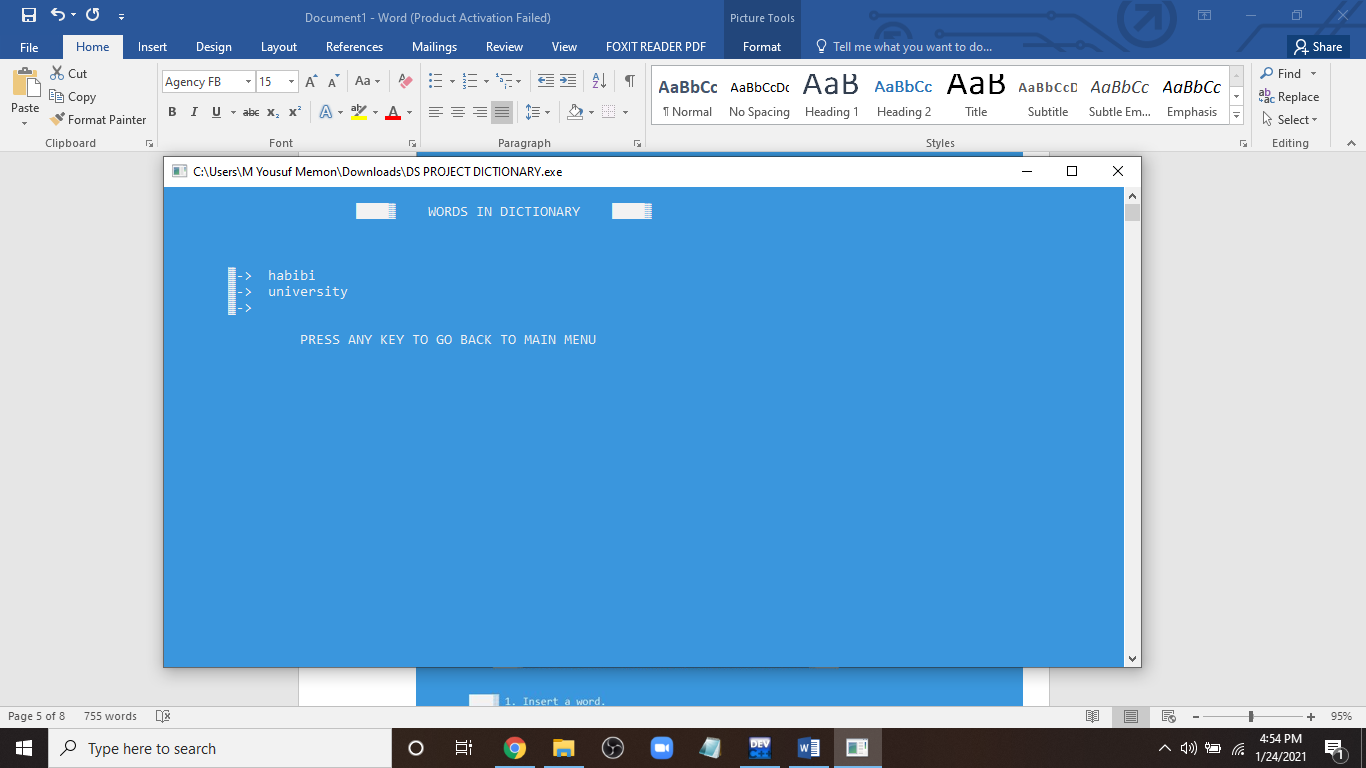
1. Finding a word :



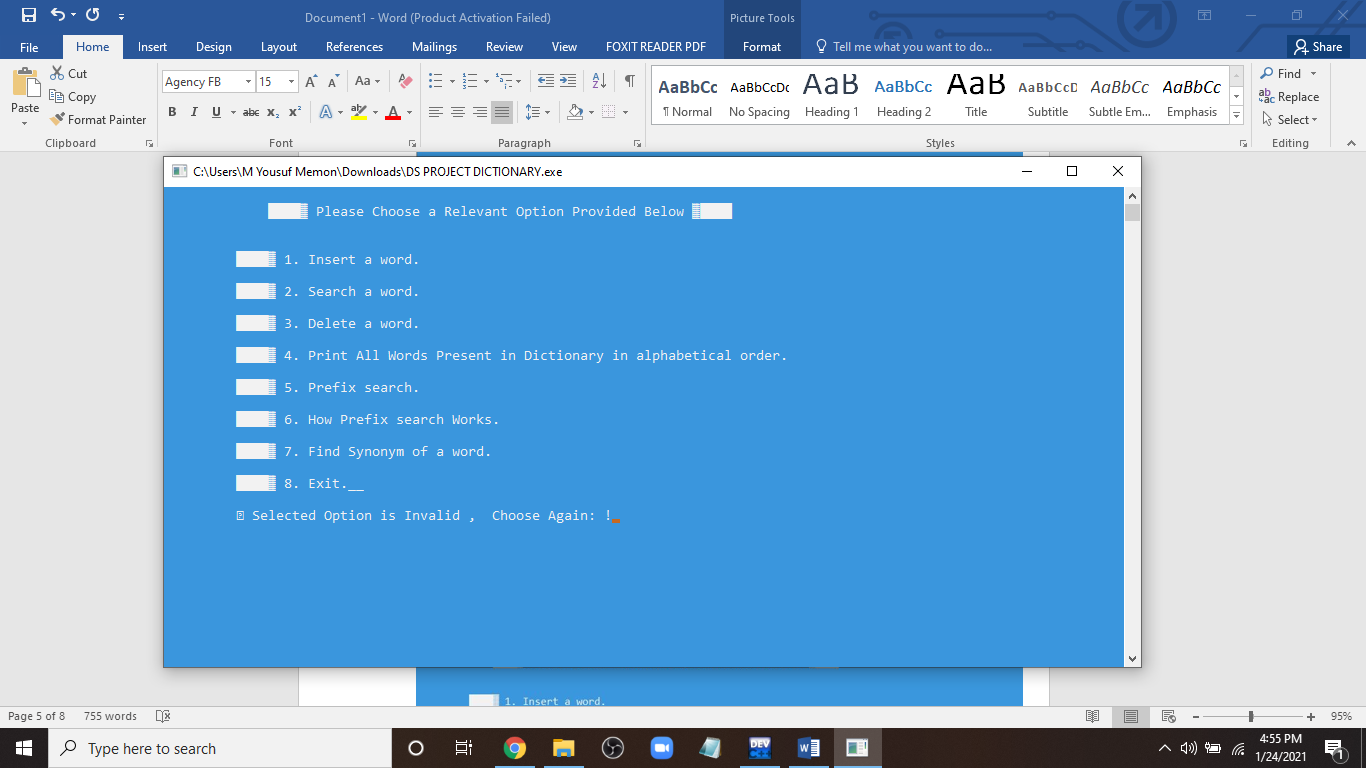
1. Finding a word That is not present :



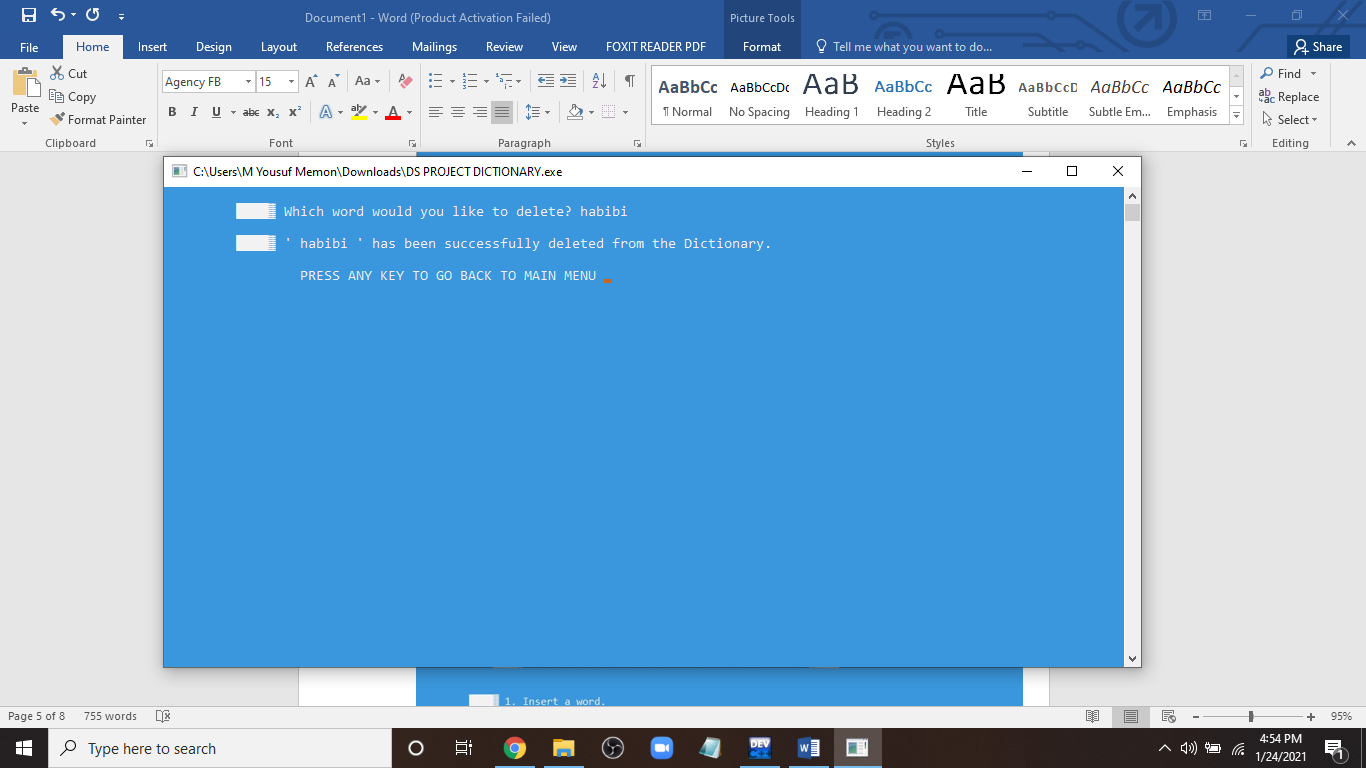
1. Printing all words :



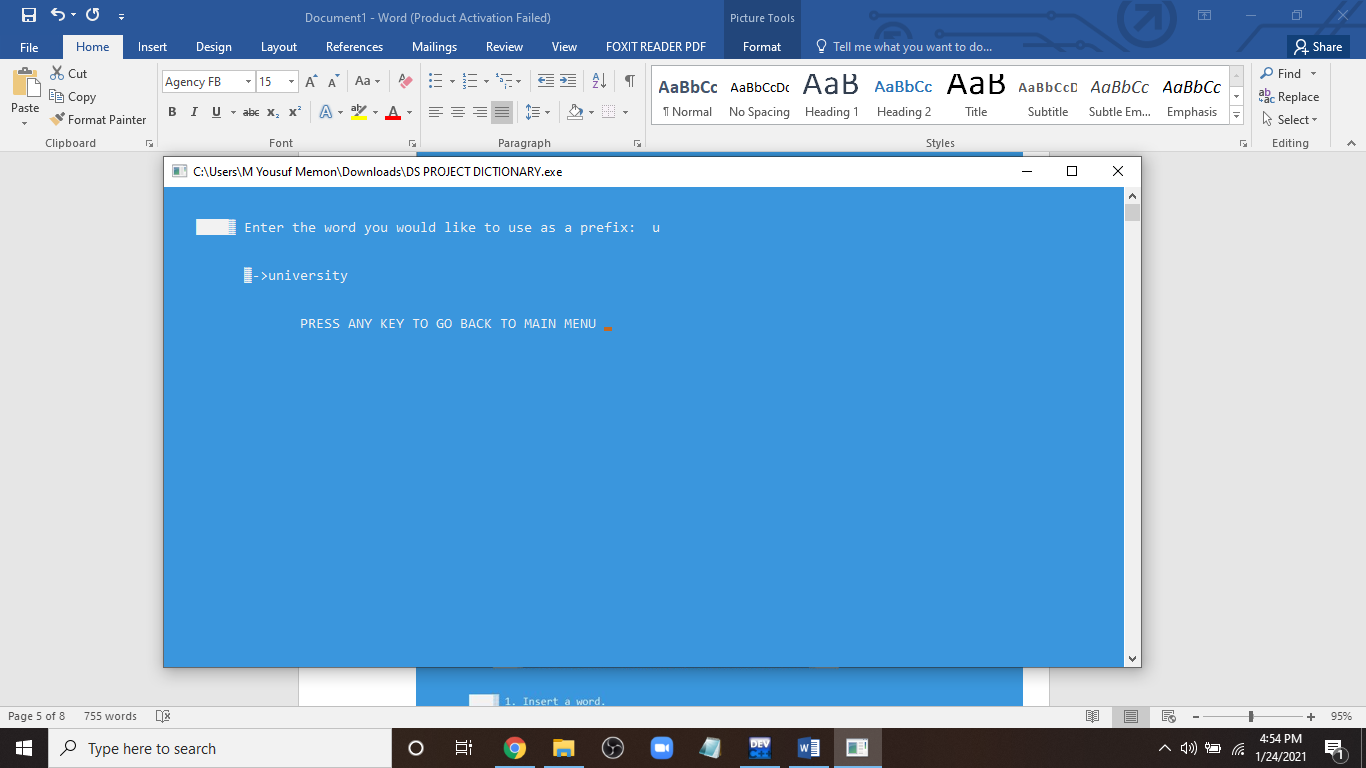
1. Invalid selection:



1. Deleting Words:



1. Prefix Search:



# Data Structure Time Complexity:

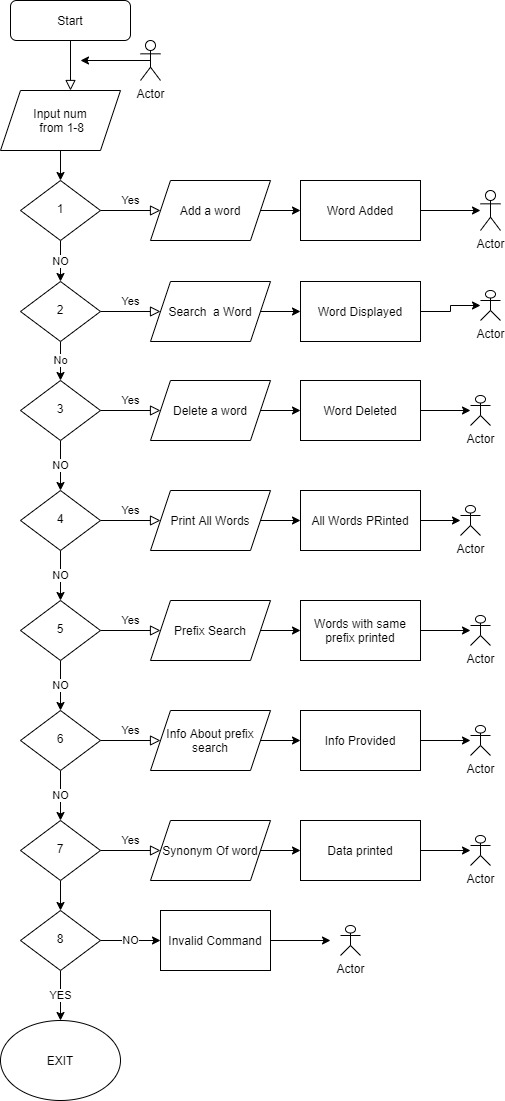
# Using Trie, search complexities can be brought to optimal limit (key length). If we store keys in binary search tree, a well-balanced BST will need time proportional to **M \* log N**, where M is maximum string length and N is number of keys in tree. However, the penalty is on Trie storage requirements. Insert and search costs **O(key-length)**, however the memory requirements of Trie is **O(ALPHABET\_SIZE \* key-length \* N)** where N is number of keys in Trie.

# Trie supports search, insert and delete operations in O(M) time where **M** is length of key.

# Results:

After working for quite some time on this, we have developed this dictionary that also has features of thesaurus, since it gives us synonyms of a word as well. The main sources of help for us were YouTube and our seniors as they helped us greatly whenever we faced a problem. The project is a well-developed working application that can be used for daily life and can also be implemented into a software so that others can also use and benefit from it.

Flow Chart:



# Conclusion:

# We have learned a lot from the following project of creating a word dictionary, got to know how to divide workload into group members, how to utilize coding skills, how to pay attention to what’s more important and what things need to be done for the completion of group task. We also learned about application of the data structures which we have studied in this course and that we cannot just simply implement any data structure that we know but we have to carefully research a single most efficient and better data structure which saves our time and memory.