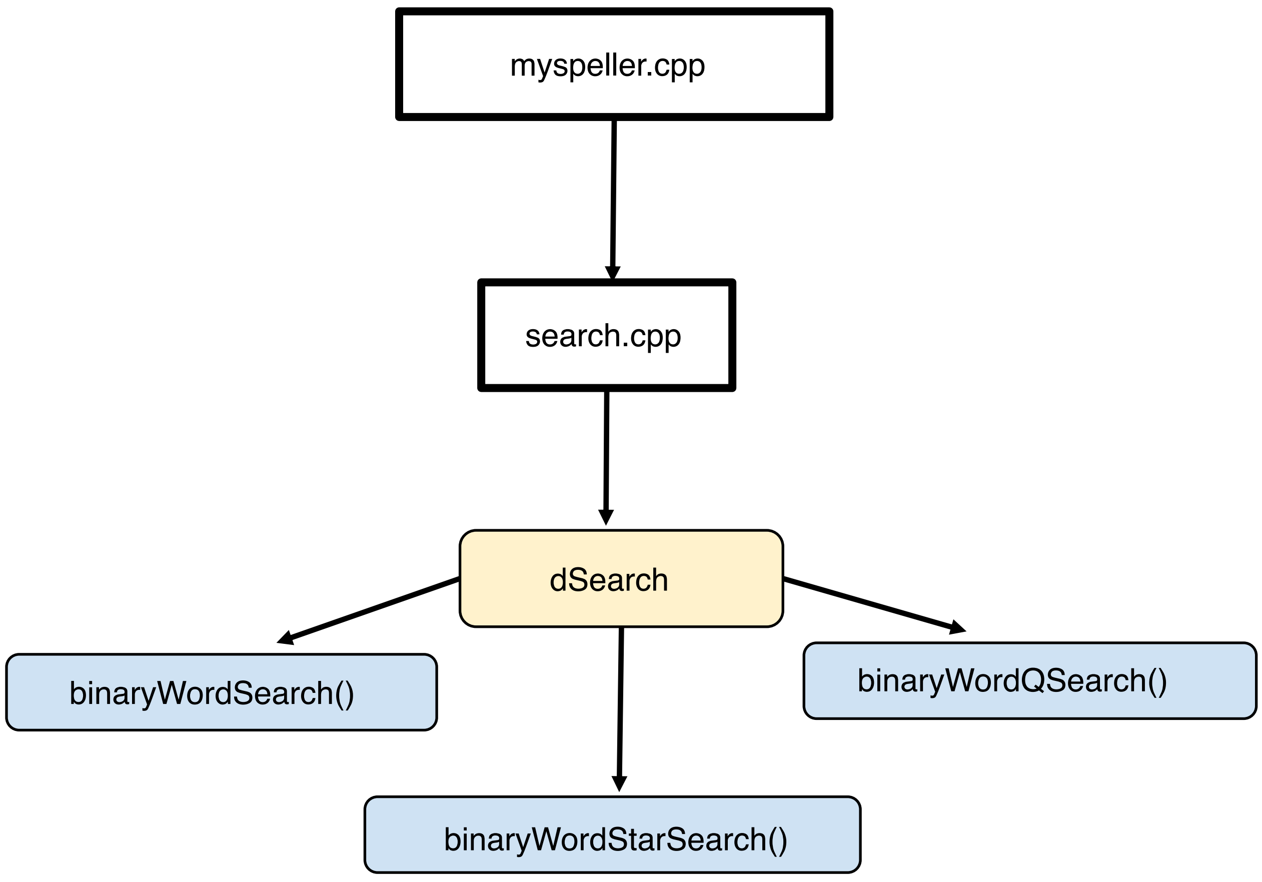
Data-Structure-Project-1

This is a speller application termed myspeller, that can help a user lookup English words in a provided dictionary. This program is created using the elementary data structures (mostly arrays) and the Linux/FreeBSD C++ environment.
Once the program starts, it presents the user with a prompt. Every time, the user types in a query, the application generates and prints out the result(s) and finally, it gives the prompt again to indicate that it expects the next query. The program terminates once the user types at the prompt exit.

Structure

The program takes the input of the search string from the user and calls one of 3 functions depending on the type of search.
○ If there is a * mark in the string, the program would call the binaryWordStarSearch function.
○ If there is a ? mark in the string, the program would call the binaryWordQSearch function.
○ If neither of these two marks is present in the string, the program would call the normal binaryWordSearch function.



Prerequisites

All the files including Makefile, myspeller.cpp, search.cpp, search.h, and dictionary files should be in the same directory
Sorted words in dictionary files, each word per line.

Getting Started

This instruction will get you a copy of the project up and running on your local machine for development and testing purposes.
./myspeller -d dictionaryfilename -l numberOfWordsSearch
The following is an example of running the program via command line.

```
./myspeller -d Dictionary141words.txt -l 2 //user input
filename = Dictionary141words.txt
WordCount = 2
Dictionary size is 141
Which word are you looking for? Type 'exit' to stop :
college //user input
word found
7 word comparisons carried out
Which word are you looking for? Type 'exit' to stop :
co* //user input
word found
10 word comparisons carried out
college
company
Which word are you looking for? Type 'exit' to stop :
col?ege
college
word found
8 word comparisons carried out
Which word are you looking for? Type 'exit' to stop :
exit  //user input
```

Algorithmic Analysis

In the Main.() function, reading from the dictionary file and storing the data into an array has O(n) running time.
In the Seaching class, binary search has O(n logn) running time.
binaryWordStarSearch and binaryWordQSearch functions have additional search of linear complexity on part of the dictionary array, which does not affect the overall complexity.

Authors

* **Julie Liu**
* **Romeno Wenogk Fernando**

Acknowledgments

* The project is finished under the guidance of Professor Stavros Kolliopoulos and instructor Khalid Mengal in New York University Abu Dhabi Computer Science Department