WordSearch Algorithm

Milan H Patel
Hossein Alishah
Robin Malhotra

Data Structures Used

- Arrays
- Linear Search
- Recursive Linear Search

Arrays allow random access and require less memory per element (do not need space for pointers) while lacking efficiency for insertion/deletion operations and memory allocation. On the contrary, linked lists are dynamic and have faster insertion/deletion time complexities. However, linked list have a slower search time and pointers require additional memory per element in the list. Hence we chose to work with arrays over linked lists since the main aim of this project was to just search for the element, and that does not involve insertion and deletion.

Run Time and Space Complexities

Run Time Complexity of searching an element in the array is :

linearly searching through an array via its index, has a complexity of **O(n).**

Recursive linear searching through an array via its index, has a complexity of **O(n).**

Space Complexity:

Space Complexity of Linear Search: O(1)

Space Complexity of Recursive Linear Search: O(n)

Linear Search Algorithm

Element/ filename	1000	10000	100000	1000000	10000000	10000000
Last Element	159ms	1145ms	12301ms	126341ms	1.23896e+06ms	2.01354e+07ms

The above table displays the amount of time taken to search for the given word checked for the last word in all six files it means we checked for the worst case complexity run time for different files.

Recursive Linear Search Algorithm

Element/ filename	1000	10000	100000	1000000	10000000	10000000
Last Element	91ms	575ms	5489ms	67973ms	665689ms	2.07714e+07ms

The above table displays the amount of time taken to search for the given word checked for the last word in all six files it means we checked for the worst case complexity run time for different files.

Results:

1000.txt

```
Enter the full name of the file: 1000.txt
Enter the word to find in the file: ezrzop

========================================================================
Linear Search: Element 'ezrzop' is found at line: 1000
Linear Search time: 159 ms (Milliseconds)

****************************************
Linear Recursice: Element found at index 1000
Linear Recursive Search time: 91 ms (Milliseconds)

========================================================================
sh: PAUSE: command not found
Program ended with exit code: 0
```

100000.txt

```
Enter the full name of the file: 100000.txt
Enter the word to find in the file: ypetjk

========================================================================
Linear Search: Element 'ypetjk' is found at line: 100000
Linear Search time: 12301 ms (Milliseconds)

****************************************
Linear Recursice: Element found at index 100000
Linear Recursive Search time: 5489 ms (Milliseconds)

========================================================================
sh: PAUSE: command not found
Program ended with exit code: 0
```

10000000.txt

```
Enter the full name of the file: 10000000.txt
Enter the word to find in the file: fslvee

========================================================================
Linear Search: Element 'fslvee' is found at line: 10000000
Linear Search time: 1.23896e+06 ms (Milliseconds)

****************************************
Linear Recursice: Element found at index 10000000
Linear Recursive Search time: 665698 ms (Milliseconds)

========================================================================
sh: PAUSE: command not found
Program ended with exit code: 0
```

10000.txt

```
Enter the full name of the file: 10000.txt
Enter the word to find in the file: veajkf

========================================================================
Linear Search: Element 'veajkf' is found at line: 10000
Linear Search time: 1145 ms (Milliseconds)

****************************************
Linear Recursice: Element found at index 10000
Linear Recursive Search time: 575 ms (Milliseconds)

========================================================================
sh: PAUSE: command not found
Program ended with exit code: 0
```

1000000.txt

```
Enter the full name of the file: 1000000.txt
Enter the word to find in the file: aqlhcd

========================================================================
Linear Search: Element 'aqlhcd' is found at line: 1000000
Linear Search time: 126341 ms (Milliseconds)

****************************************
Linear Recursice: Element found at index 1000000
Linear Recursive Search time: 67973 ms (Milliseconds)

========================================================================
sh: PAUSE: command not found
Program ended with exit code: 0
```

100000000.txt

```
Enter the full name of the file: 100000000.txt
Enter the word to find in the file: faphrt

========================================================================
Linear Search: Element 'faphrt' is found at line: 100000000
Linear Search time: 2.01354e+07 ms (Milliseconds)

****************************************
Linear Recursice: Element found at index 100000000
Linear Recursive Search time: 2.07714e+07 ms (Milliseconds)

========================================================================
sh: PAUSE: command not found
```

Code Snippet

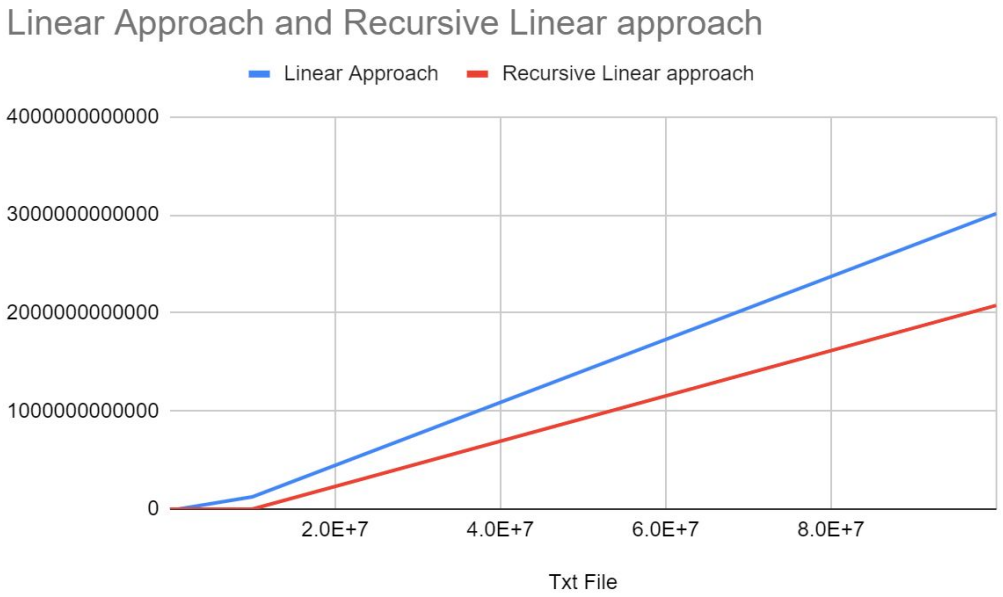
WordSearch

```cpp
#include <iostream>
#include <string>
#include <fstream>
#include <ctype.h>
#include<vector>
#include<ctime>
//#include <conio.h>

using namespace std;

int LinearSearch(vector<string> arr, string x) {

    int result = -1;

    for (int i = 0; i < arr.size(); i++) {

        if (arr.at(i) == x)
            result = i;

    }

    return result;

}

// Returns index of x if it is present in arr[],
    // else return -1
int linearRecursive(vector<string> arr, string x, int size)
{
    size--;
    int rec;
    if (size >= 0) {
        if (arr.at(size) == x)
            return size;
        else
            rec = linearRecursive(arr, x, size);
    }
    else
        return -1;
    return rec;
}
```

```cpp
int main()
{
    clock_t search_start, search_end;
    float searchruntime = 0.0;

    clock_t search_start1, search_end1;
    float searchruntime1 = 0.0;

    string wordToFind, fileName;


    cout << "Enter the full name of the file: ";
    cin >> fileName;

    cout << "Enter the word to find in the file: ";
    cin >> wordToFind;

    cout << endl << "================================================================================" << endl;


    vector<string> arr;
    string line;
    ifstream myfile(fileName);
    if (myfile.is_open())
    {
        while (!myfile.eof())
        {
            getline(myfile, line);
            arr.push_back(line);
        }
        myfile.close();
        search_start = clock();
        int result = LinearSearch(arr, wordToFind);
        search_end = clock();
        if (result == -1)
            cout << ("Linear Search: Element not present") << endl;
        else
            cout << "Linear Search: Element '" << wordToFind << "' is found at line: " << result + 1 << endl;

        searchruntime = ((float)search_end - (float)search_start);
        cout << "Linear Search time: " << searchruntime << " ms (Milliseconds)" << endl;

        cout << endl << "****************************************" << endl;

        int n = arr.size();

        search_start1 = clock();
        int result1 = linearRecursive(arr, wordToFind, n);
        search_end1 = clock();

        if (result1 == -1)
            cout << ("Linear Recursice: Element not present");
        else
            cout << ("Linear Recursice: Element found at index ") << result + 1 << endl;
        searchruntime1 = ((float)search_end1 - (float)search_start1);
        cout << "Linear Recursive Search time: " << searchruntime1 << " ms (Milliseconds)" << endl;
    }
    else cout << "can't open the file";
    cout << endl << "================================================================================" << endl;
    system("PAUSE");
```

No issues found

Graph:



The above graph compares the linear search vs recursive linear search, with X axis being the text file being used, and y axis consists of time taken to search the last element in that file.

Thank You!