CSE: 224 DATASTRUCTURE & ALGORITHMS

Lab 5 Report

|  |  |
| --- | --- |
| Names | IDs |
| Aaser Fawzy Zakaria Hassan | 19015403 |
| Mohamed Ezzat Saad El-Shazly | 19016441 |
| Ahmed Hamdy Ahmed Osman | 19017253 |

Contents

[Problem Statement: 2](#_Toc104067588)

[Requirements: 2](#_Toc104067589)

[Implementation details & design choices: 3](#_Toc104067590)

[B- Tree 3](#_Toc104067591)

[Search Engine 3](#_Toc104067592)

[Analysis: Graphs & Tables 4](#_Toc104067593)

[B-Tree: 4](#_Toc104067594)

[Tables: 4](#_Toc104067595)

[Graphs: 5](#_Toc104067596)

[Complexity: 6](#_Toc104067597)

[Search Engine: 7](#_Toc104067598)

[Tables: 7](#_Toc104067599)

[Graphs: 7](#_Toc104067600)

[Complexity: 8](#_Toc104067601)

# Problem Statement:

1. You are required to implement a generic B-Tree where each node stores key-value pairs and maintains the properties of the B-Trees use the provided interfaces.
2. You will be given a set of Wikipedia documents in the XML format, and you are required to parse them and maintain an index of these documents content using the B-Tree to be able to search them efficiently use the provided interfaces.

# Requirements:

1. A generic B-Tree data structure that implements the given interfaces.
2. A Search Engine that uses the B-tree to search for given words in the provided Wikipedia xml documents.

# Implementation details & design choices:

## B- Tree

1. using 3 lists to store keys, values and children’s in every node.
2. to enter any key or value to node it should be wrapped in a list
3. key enter first then value or child second in the node.
4. the operations (insertion & deletion) happens first and if there is error fixup take place second.
5. search return null if the object is not in the tree.
6. in order to make the mapping there is a control variable in the code the node where the key then value then value or the children second.
7. we use inline coding to make the code faster
8. transvers done using stack and iterative method no recursive

## Search Engine

* Based on the above implementation the Search engine first indexes the files/folders then the read values are then inserted with key of type Integer: id of the “doc” and with value: context of the “doc” into the webpage tree.
* The searching for one word traverses each node in the B-Tree using recursive in-order traversal to search each node’s list of values for the word then counting its frequency in this string. The result is then added to ISearchResult and pushed into the list.
* The searching of multiple words assuming they are separated by “ “ are split into multiple words and searched for individually and then for each result there will be a list, and for each “doc” searched in the word with the minimum frequency is picked. After this the results are added as above.
* The string for searching for multiple words should be “ “ separated.

# Analysis: Graphs & Tables

## B-Tree:

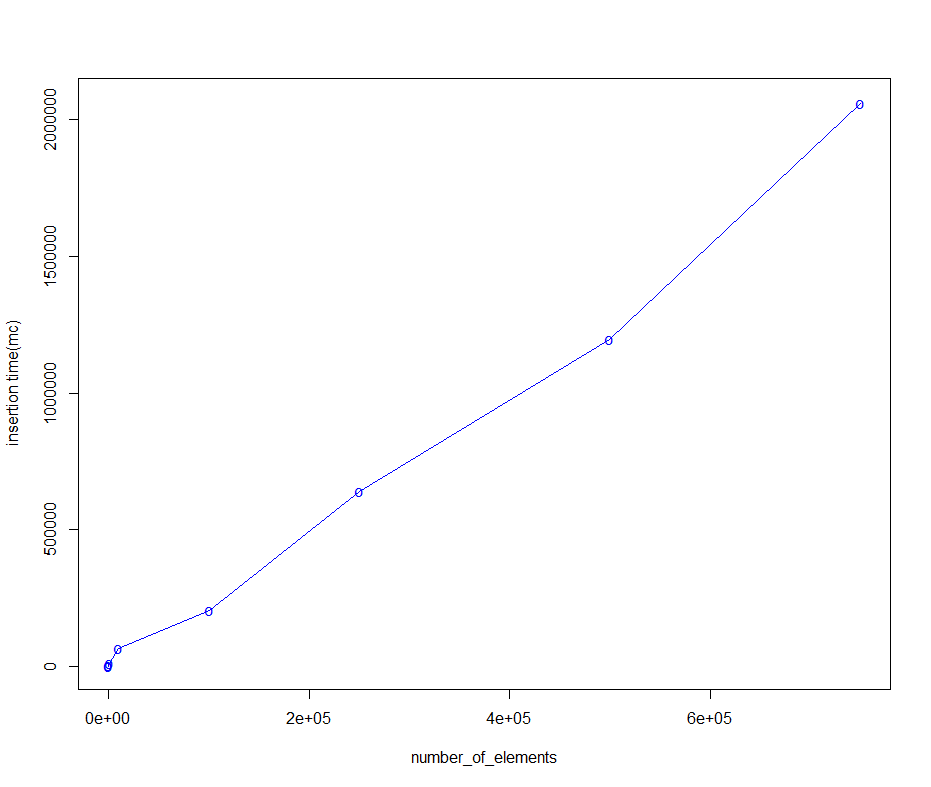
### Tables:

#### Insertion table

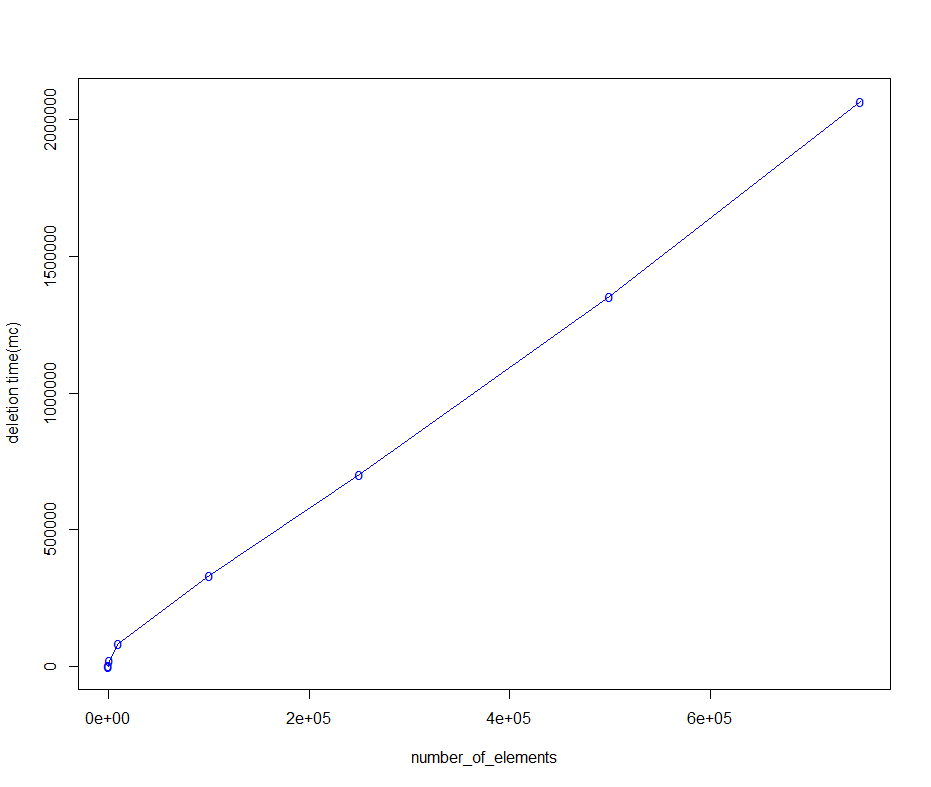
#### Deletion table

### Graphs:

#### Insertion Graph



#### Deletion Graph



### Complexity:

From the graph both insertion & deletion times in B-Tree are almost O(log n) and since space complexity of any tree is O(log n ) & each node has lists of size O(n) then space complexity is O(n log n).

## Search Engine:

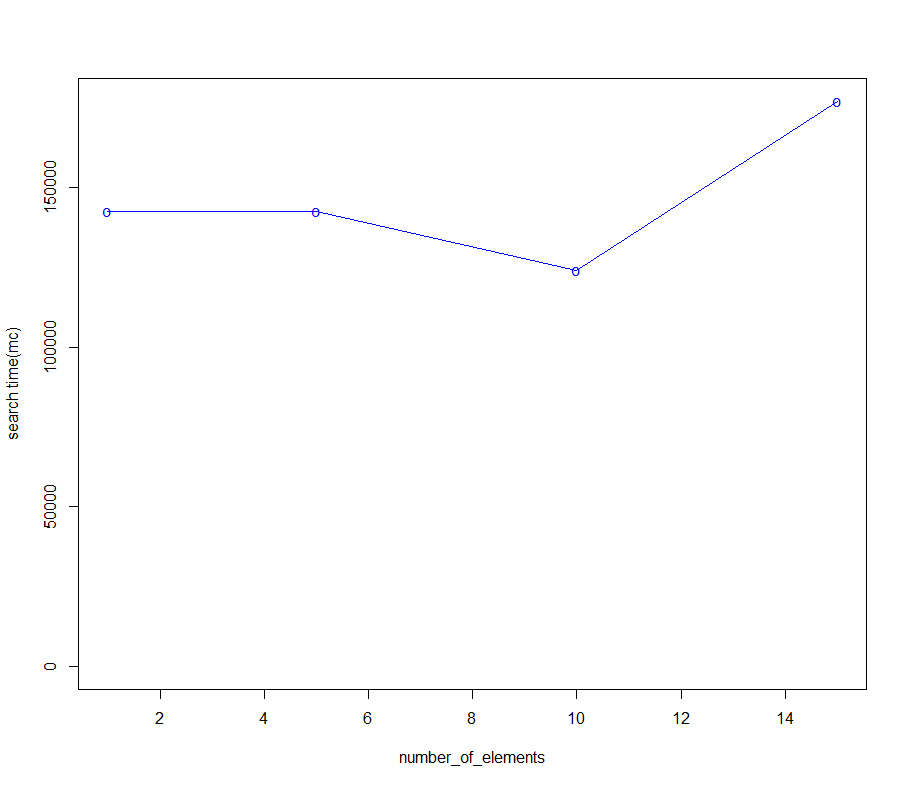
### Tables:

#### Look up multiple words table



### Graphs:

#### Look up multiple words Graph



### Complexity:

Search Engine searches in each node of the tree which takes O(log n) and then searches in each node’s list of values to find the proper occurrences with O(n) then the total is O(n log n) in terms of space the search engine has only one B-Tree data structure which takes O(n log n) space.