CSC2001F Assignment 2

RBLCAM001

OO Design

My OOP designs is based around the central object type of Student. This class creates an object that will be used to store the individual data of each of the students in the text file. The student object consists of the parameters of a name and a student number. This class also then contains getter and setter methods for the variables as well as a toString and a compareTo method that compares the student number of students. The next class is a StudentAVL object that uses an AVL tree implementation to fill an AVL tree with the data from the oklist text document in the form of student objects. The class also has an accessor method that will return the AVL tree with all the students populated in it. There is also a helper class that is used for encapsulation and information hiding from the user as it takes away the methods that search for students in the tree from the main user interface class. There is also a host of classes that are used to create an AVL tree and use object types of Binary trees and various other support classes. Finally, there is the AccessAVLApp class that is used to run the actual program that only consists of a min class.

Experiment

The goal of this experiment is to determine the speed at which a program can search through an AVL tree. The experiment uses instrumentation to keep track of the number of comparisons that are made while searching for desired result. The experiment is conducted by using a bash file to create subsets of the oklist data by randomly choosing batches of multiples of 500 entries i.e., 500 random entries then 1000 ransom entries then 1500 and so on. Each time a batch is made in the for loop – each Student number is then sent to the AccessAVLApp where the data structure then looks through its data and counts the number of comparisons it makes until they find the corresponding data in the entire list that they are storing. The program then stores the number of comparisons it used in order to find that corresponding piece of data in a temporary text file. This data is then used to determine the best case, worst case, and average case of each of the subsets for each of the data structures and outputs this to the terminal. This data can then be graphed.

Trials

AccessAVLApp

Input:

Valid 1: DYNJUN011

Valid 2: VNXOLE007

Valid 3: KMHPHE006

Invalid 1: RBLCAM001

Invalid 2: WLTGEO002

Invalid 3: ATTRAE001

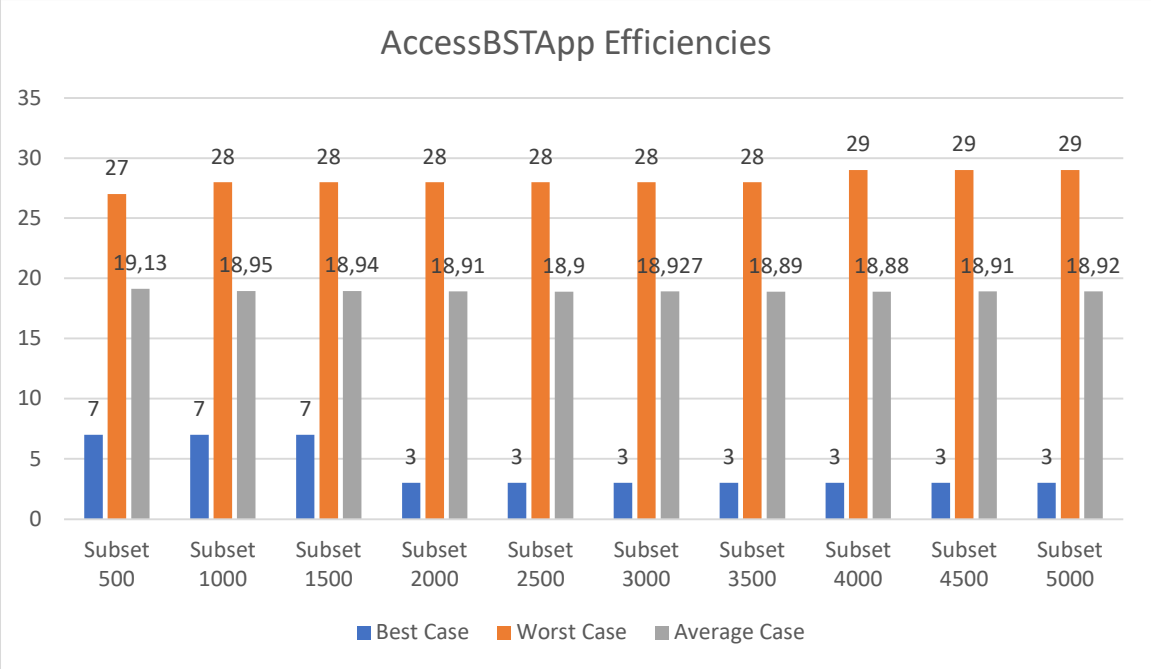
```
cameron@cameronlaptop-VirtualBox:~/Assignment2/data$ more AVLValid1.txt
Phenyo Khumalo
cameron@cameronlaptop-VirtualBox:~/Assignment2/data$ more AVLValid2.txt
Olerato Van
cameron@cameronlaptop-VirtualBox:~/Assignment2/data$ more AVLValid3.txt
Junior Dyantyi
cameron@cameronlaptop-VirtualBox:~/Assignment2/data$ more AVLInvalid1.txt
Access denied!
cameron@cameronlaptop-VirtualBox:~/Assignment2/data$ more AVLInvalid2.txt
Access denied!
cameron@cameronlaptop-VirtualBox:~/Assignment2/data$ more AVLInvalid3.txt
Access denied!
cameron@cameronlaptop-VirtualBox:~/Assignment2/data$ more AVLNoParameters.txt
```

Input:

No parameters

```
cameron@cameronlaptop-VirtualBox:~/Assignment2/data$ cat AVLNoParameters.txt |
head -5
Noah Maluleke MLLNOA014
 Omaatla Khoza KHZOMA010
  Melokuhle Adams DMSMEL001
   Bonolo Booi BXXBON021
    Katleho Abrahams BRHKAT012
cameron@cameronlaptop-VirtualBox:~/Assignment2/data$ cat AVLNoParameters.txt |
tail -5
           Thato Witbooi WTBTHA010
            Siyabonga Witbooi WTBSIY016
          Tshegofatso Witbooi WTBTSH028
           Tshegofatso Witbooi WTBTSH025
           Warona Witbooi WTBWAR001
cameron@cameronlaptop-VirtualBox:~/Assignment2/data$
```

Results



From the data presented in the graph, we can see that the data stored in the AVL tree is more efficient than the data structures that we saw previously. The search function of the AVL is far more efficient at searching through the dataset than a linear search. The tree can efficiently search for data with the worst case not being very high at all. The data set contains 5000 entries and so an average case of 20 comparisons is very low. This dataset is more complex because of the rotate operations that are needed in order to insert the data into the tree, but the graph shows that the higher complexity is more beneficial for the overall efficiency of sorting and searching through the data.

Creativity

During this assignment I applied my own method and procedures largely based on the creativity from our previous assignment. This assignment was almost the same as the last assignment apart from the change of data structure and so there was much need for adding new features between the two assignments. My creativity was that my code was so modular that it was easy to adapt it to this project with only a few problems and bugs to be fixed.

Git