CSC2001F 2024 Data Structures Assignment 2 Report – MRRJOS007

Object Orientated Design and Interaction:

GenericsKbAVLApp:

- This class represents the `main()` application for managing a knowledge base using an AVL tree
- It creates an object of type GUI to run methods to generate a GUI for the examiner
- Reads data from files and stores queries in an ArrayList and the knowledge base in an AVL tree
- Has methods to search through the AVL tree and find any query by looking if they share the same key

GUI:

- This class creates a graphical user interface for creativity and ease of use for the examiner
- It has methods start(), part1(), part2() and a lot of accessor methods for retrieving the various buttons to be used in GenericsKbAVLApp.

AVLTree:

- Implements the AVL tree data structure with methods for insertion, searching, balancing and traversal.
- AVL tree maintains balance by updating node heights and performing rotations when necessary during insertion.

Entry:

- This class represents an entry in the knowledge base.
- It contains fields for storing the term, statement and confidence score of each entry.
- It provides methods for accessing and modifying these fields (`getTerm()`, `getStatement()`, `getScore()`, `setStatement()`, `setScore()`).

Goal and Execution:

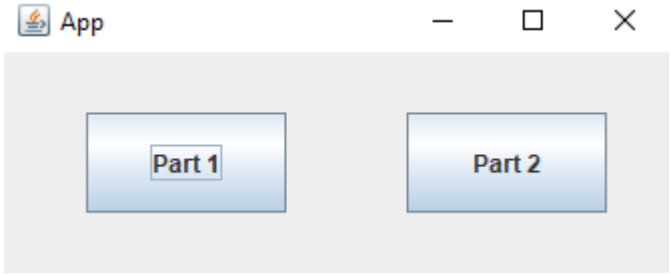
The goal of this assignment is to analyse the performance of AVL Trees by implementing an application that stores and retrieves data using this data structure. The experiment aims to compare the actual performance with theoretical expectations based on the size of the dataset. This report outlines the design, implementation, experiment execution, results and analysis.

I conducted experiments to measure the number of comparison operations (insert and search) for varying dataset sizes (n). The dataset used was provided in the file GenericsKB.txt. I selected 10 values of n (5, 50, 500, 5000, 50000) spaced logarithmically.

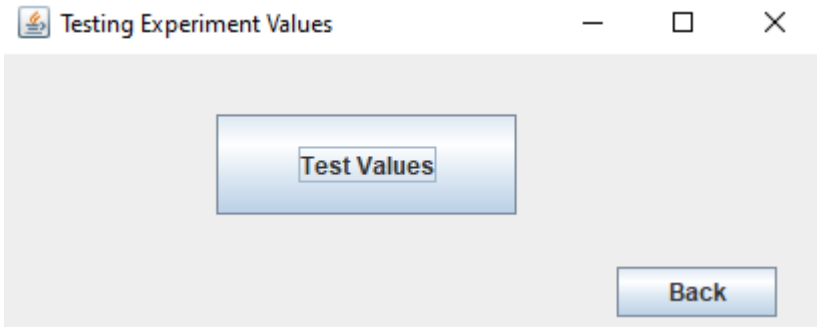
I instrumented the code to count the number of comparison operations during insertions and searches. This involved adding counters in the AVL tree implementation and updating them appropriately during comparisons.

Results:

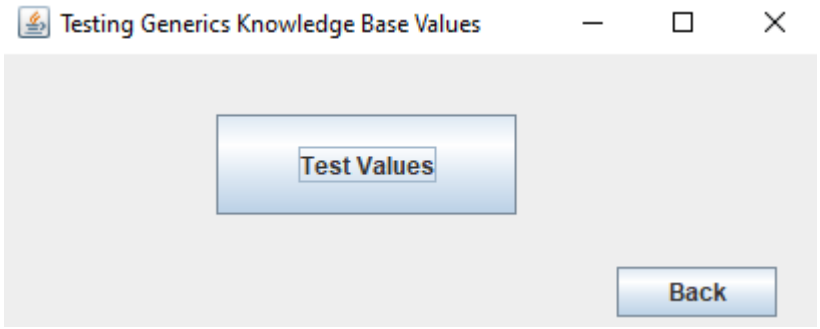
When the program is run a GUI pops up and prompts user to choose Part 1 or Part 2



This is the output GUI for part 1



This is the output for part 2



Part 1:

For the purpose of the report I will show all the terminal output but if command ‘make run’ is typed all output uses unix redirection and stores this in output.txt. The following was the manual txt file with 10 entries. I copied 7 keys from the GenericsKB-queries and created 3 (random, test, help) that are not in the knowledge base to test if my program was able to work with terms within the knowledge base and with terms outside.

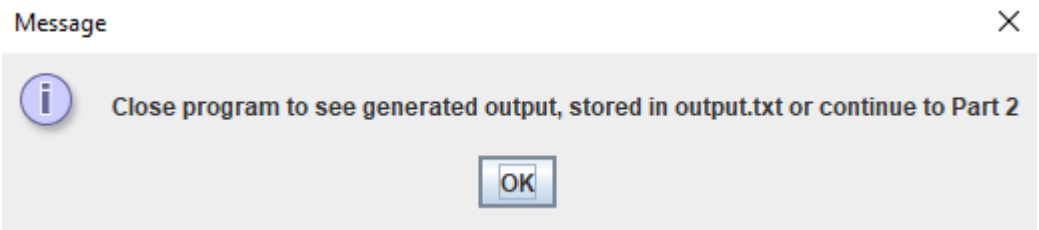
Experiment_Testing_Values.txt

```
competitor
funny
random
petroleum coke
generalized epilepsy
test
another random
help
wild garlic
medical receptionist
```

```
Knowledge base loaded successfully.

competitor: Competitors are located in sporting events. (1.0)
funny: Funnies are located in television. (1.0)
Term not found: random
petroleum coke: Petroleum coke is a mixture of carbon and hydrogen joined by nature in many different ways. (0.7167034149169922)
generalized epilepsy: Generalized epilepsy is epilepsy (1.0)
test: Tests involve analyses. (1.0)
Term not found: another random
Term not found: help
wild garlic: A wild garlic is a bulbous plant (1.0)
Term not found: medical receptionist

Comparisions = 244
Inserts = 50000
```

As you can see the AVL tree has been populated with 50000 entries and it made 244 comparisons from the query file that contain only 10 queries.

The following message pops up to help guide the user.



Part 2:

The user can now select the back button which opens the first frame and can now select Part 2.

```
Term not found: splayed leg
Term not found: amphidiploid
Term not found: sponger
Term not found: moon jelly
Term not found: follicular cast
Term not found: riptide

Comparisions = 12600
Inserts = 5

Average case for inserts: 5
Average case for comparisons: 12600
```

In the first instance the tree has been inserted with 5 entries and has 12600 comparisons totalling to 5 average inserts and 12600 average comparisons.

```
Term not found: sponger
Term not found: moon jelly
Term not found: follicular cast
Term not found: riptide

Comparisions = 29565
Inserts = 50

Average case for inserts: 27
Average case for comparisons: 21082
```

In the second instance the tree has been inserted with 50 entries and has 29565 comparisons totalling to 27 average inserts and 21082 average comparisons.

```
Term not found: amphidiploid
Term not found: sponger
Term not found: moon jelly
Term not found: follicular cast
Term not found: riptide

Comparisions = 46782
Inserts = 500

Average case for inserts: 184
Average case for comparisons: 29649
```

In the third instance the tree has been inserted with 500 entries and has 46782 comparisons totalling to 184 average inserts and 29649 average comparisons. In this instance we finally start seeing that some of the terms have been found.

```
Term not found: amphidiploid
Term not found: sponger
moon jelly: Moon jellies catch small plankton with tentacles, covered with stinging cells, called nematocysts. (0.8074753880500793)
follicular cast: Follicular casts are tightly adherent scales around the hair shaft. (0.7716030478477478)
Term not found: riptide

Comparisions = 67781
Inserts = 5000

Average case for inserts: 1388
Average case for comparisons: 39182
```

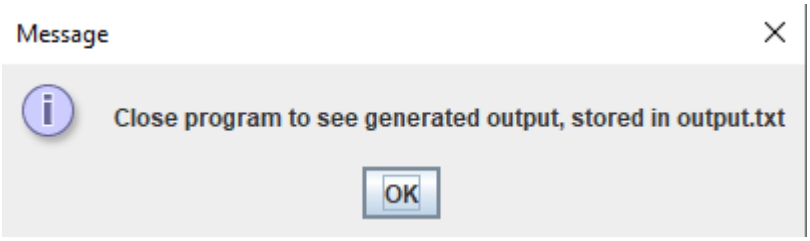
In the fourth instance the tree has been inserted with 5000 entries and has 67781 comparisons totalling to 1388 average inserts and 39182 average comparisons.

```
sponger: A sponger is a follower (1.0)
moon jelly: Moon jellies catch small plankton with tentacles, covered with stinging cells, called nematocysts. (0.8074753880500793)
follicular cast: Follicular casts are tightly adherent scales around the hair shaft. (0.7716030478477478)
riptide: Riptides are current. (1.0)

Comparisions = 141245
Inserts = 50000

Average case for inserts: 11110
Average case for comparisons: 59594
```

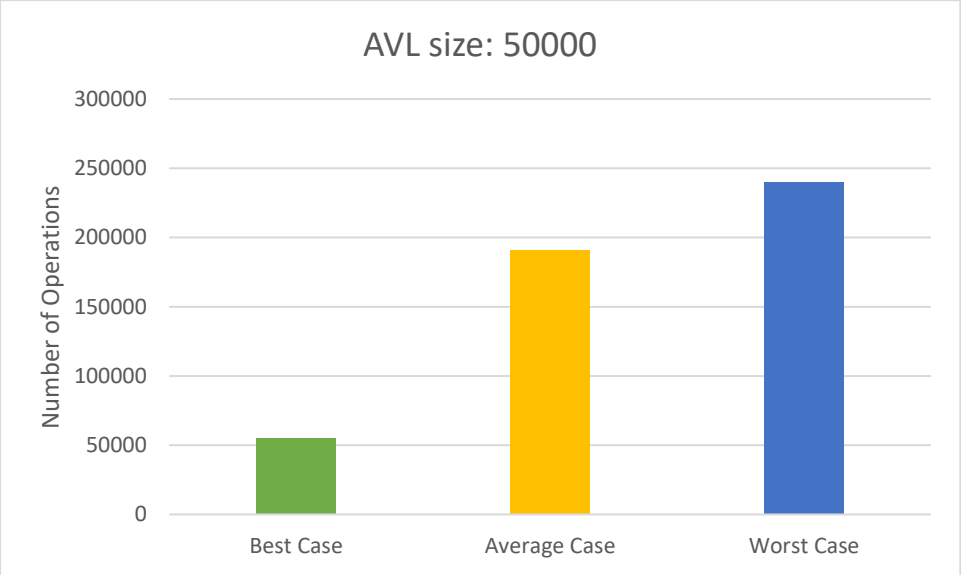
And in the final instance we see when the AVL tree has 50000 inserts there were 141245 comparisons. The total average inserts were 11110 and total average comparisons were 59594.

As you can see it prints all the terms that were found and terms that werent found in the 5000 query file named, GenericsKB-queris.txt. There is too many terms outputed so you can look in the output.txt to see all 5000.

The following message pops up indicating the file was successfully stored all the outputted data.



Graphs:



Creativity:

I created a GUI for the user to navigate through with minimalistic effort that has a well-constructed and clean design. I created my own GUI class that creates the GUI and then implemented the buttons in the main method of the GnericsKbAVLApp to run code whenever a button is pressed.

Git Summary:

```
0: commit cfdd56311888300d2c29802cc1627e5321c2099e
1: Author: Josh Murray <mrrjos007@myuct.ac.za>
2: Date: Fri Mar 22 13:32:38 2024 +0200
3:
4: Finished the searchTerm method to work for any AVL size
5:
6: commit a05f3bf81b271fbccb2ce96b23e12c750d572fa4
7: Author: Masterjosh3107 <masterjosh3107.murray@gmail.com>
8: Date: Fri Mar 22 11:29:00 2024 +0200
9:
...
37: Author: Josh Murray <mrrjos007@myuct.ac.za>
38: Date: Mon Mar 11 10:18:49 2024 +0200
39:
40: Created GenericsKbAVLApp
41:
42: commit 98f972e653e3f93cfed940f0c39c4ac28140979a
43: Author: Josh Murray <mrrjos007@myuct.ac.za>
44: Date: Mon Mar 11 08:45:29 2024 +0200
45:
46: Initial git commit
```