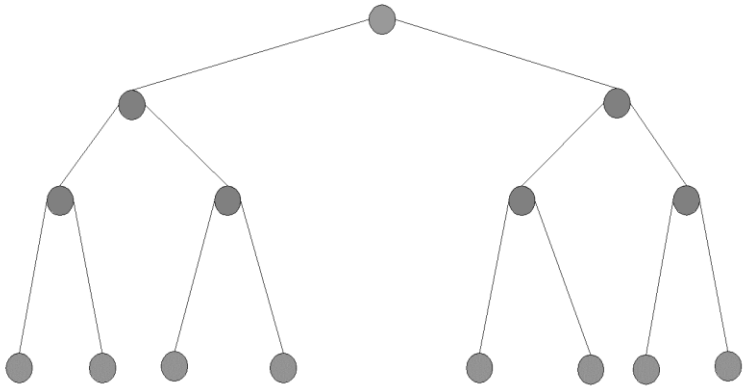


Assignment 2

AVL TREES

Daniel Vorster | CSC 2001F | VRSDAN004

Contents

The Problem

For this Assignment, we are given the task of creating an Electronic Telephone Directory. The program needs to make use of a AVL Tree as the core data-storage and searching method. Users need to be able to input a file containing a list of queries, with the expected output being either the details for each query or the text "Not found," should the entry not exist.

We are required to write one program:

- **SearchAVL**: This program reads the entries from a file and loads them into an AVL Tree. Like SearchIt (from the previous assignment), it uses a file with search terms, but uses an AVL Tree structure in place of a Binary Tree structure to search for these terms.

The purpose of this program is to help us compare the operation speed of the AVL Tree and the Binary Tree, using a few tests.

Application Design

The application consists of the following class files:

- **Entry.class**

Entry.class

Entry.class serves as the datatype for the data loaded from the file, or entries. This class has parameters for name, address and number. It also serves as a parser to deconstruct (split) the single-line string, as seen in the raw data file. This means an Entry object can be initialised with data read straight from the file.

AVLTree.class

AVLTree.class is the base class for the AVL Tree. I have chosen to use Hussein's code for ease and simplicity.

BinaryTreeNode.class

BinaryTreeNode.class is the class for the Binary Tree Node. This is still required as the AVL Tree still depends on a binary tree, so nodes are still essential.

BinaryTree.class

BinaryTree.class is the base class for the Binary Tree used by the AVL Tree structure. The AVLTree class performs "AVL" operations on a standard Binary Tree, making it an AVL Tree.

InsertAVL.class

InsertAVL.class is a class developed for testing. It is developed to only insert the query data items into the AVL Tree, so insertion speed can be measured. It inserts the entire data file and takes a parameter for repetitions, so the time can be accurately measured. It provides statistics after runtime.

InsertIt.class

As like InsertAVL.class, InsertIt.class is also developed for insertion speed testing. IT behaves in the same way, only differing in core data structure: It uses a Binary Tree instead of an AVL Tree. It also provides statistics after runtime.

SearchAVL.class

SearchAVL.class is the main class of the program. It uses an AVL Tree to store and find data. Data is first loaded into the AVL Tree: After each element is read from the file using a Buffered Reader, it then parses these to the Entry datatype. They are then inserted into the AVL Tree. Data is then loaded, one item at a time, from a query file. Once an element is loaded, it is then cast to an Entry datatype (With a key only, no other information) and the find() method of the AVL Tree is then used to find the Entry, by using the key.

Once a match is found, the contents of the found Entry object are printed to the screen. If nothing is found and null is returned, “Not found” is output to the screen.

I have also implemented timing and parameter input into this class for testing. The timing acts as a test timer for the class, testing from initialisation till the class has read and output the last data line. It then offers up statistics, namely “Time taken over total reps:” and “Total time:”

This is where the parameter comes in. This program accepts a single parameter that dictates the number of repetitions that must be taken. This ensures high time measurement accuracy.

Searching

Input: Query File

Abbott Alec

Lueilwitz Candelario

Emard Sister

Jacobs Jake

Jake Jacobs

Zulauf Tyler

Howe Duane

Yundt Robb

Arnoldo Testing

Daniel Vorster

Keeling Antonia

Stamm Samir

Lehner Alberta

Kihn Nola

Nocola Tesla

Howe Abigayle

Keebler Tent

Kertzmann Cassandre

Abbott Meda

Williamson Murphy

Output: SearchAVL

Abbott Alec || 489-848-7299 || 03707 Botsford Fork, Lima

Lueilwitz Candelario || 1-077-306-6380 x65575 || 69026 Upper, Mission Viejo

Emard Sister || 1-734-468-6031 x836 || 28366 Jovanny Circle Apt. 524, Fullerton

Jacobs Jake || 311.582.0721 || 21176 Abernathy Passage, Hawaiian Gardens

Not found

Zulauf Tyler || 195-358-6930 x523 || 29262 Monahan Rest, Newport Beach

Howe Duane || 023.594.9084 x23601 || 59562 Rear, Riverside

Yundt Robb || 851-253-7017 x086 || 96145 Runolfsson Crescent, San Rafael

Not found

Not found

Keeling Antonia || 960.415.9539 x202 || 59282 Harvey Gardens, City of Industry

Stamm Samir || 906-736-1118 x68420 || 52019 McCullough Lake, West Hollywood

Lehner Alberta || (460)301-1274 x351 || 67100 Schumm Pines, Barrow

Kihn Nola || 1-595-066-0797 || 02200 Abbigail Inlet, Montgomery

Not found

Howe Abigayle || 682.826.1348 x2616 || 47833 Office, Long Beach

Not found

Kertzmann Cassandre || 768.834.5941 x102 || 40643 Nienow Plain, Beverly Hills

Abbott Meda || 1-117-789-3061 || 18565 Suite B, Fountain Valley

Williamson Murphy || 1-315-726-5402 x71144 || 14846 Crooks Spring Apt. 480, West Memphis

Comparison Experiment

Method

This experiment was made easy thanks to the addition of testing and timekeeping operations build into the application files. The programs, when run, take in a parameter. This parameter dictates the amount of times the given operations should be run. This allows us to easily repeat tasks hundreds of times to get extremely accurate readings.

The applications also keep their own time, and a report is displayed once the application has run, displaying total runtime and the average runtime for a single the n repetitions.

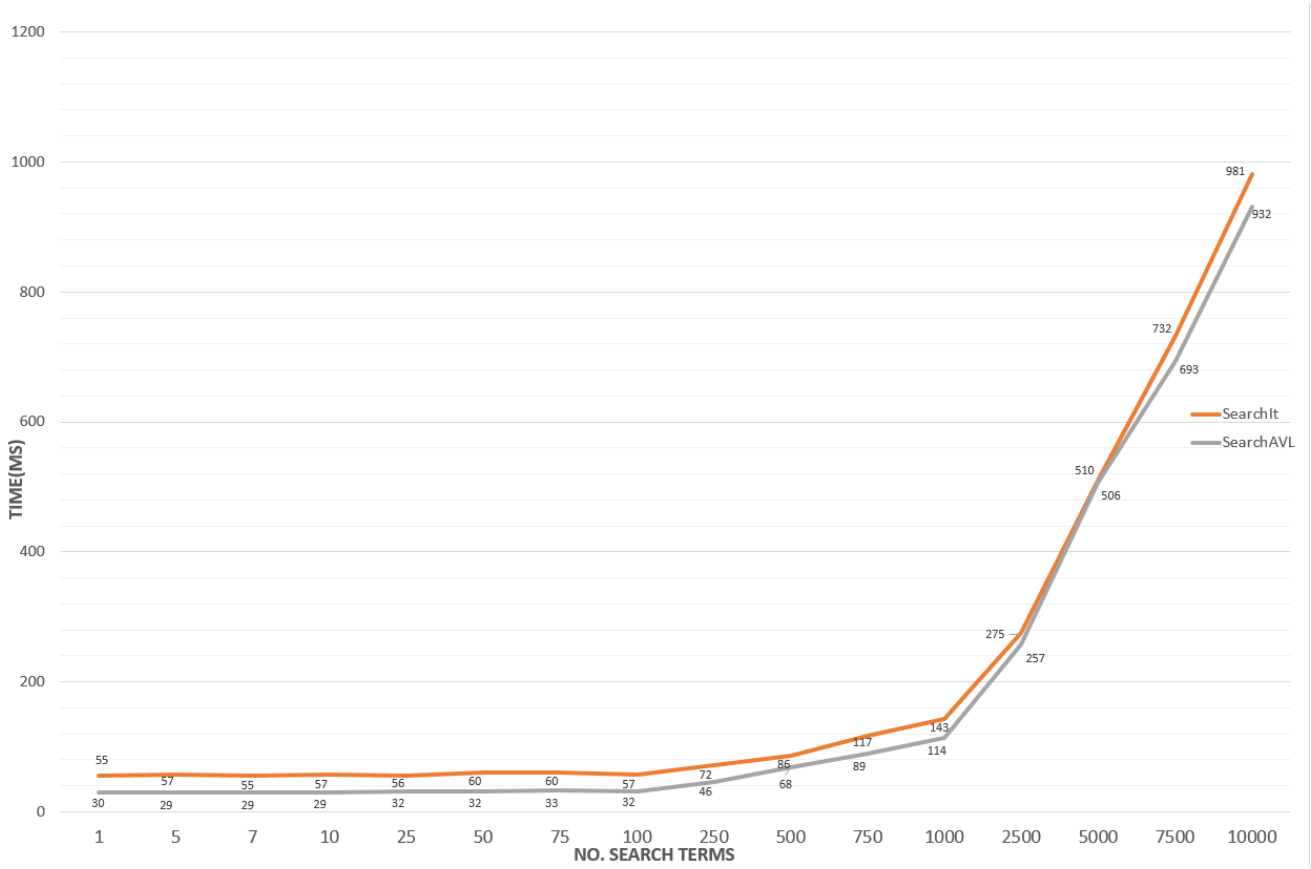
The application QueryFileGenerator was used to generate query files of given length, then both SearchIt and SearchAVL were run 100 times using the same query file and the average values were recorded.

Raw Data - Results

The following data was collected during the testing:

n	SearchIt (ms)	SearchAVL (ms)
1	55	30
5	57	29
7	55	29
10	57	29
25	56	32
50	60	32
75	60	33
100	57	32
250	72	46
500	86	68
750	117	89
1000	143	114
2500	275	257
5000	510	506
7500	732	693
10000	981	932

Summary – Results and Conclusion



The above chart is a summary of the test data. In conclusion, we can say that the search operation on an AVL Tree is faster than that of a Binary Tree. However, the difference is only slight, with an average difference of 26.375ms. This was run on a fast computer, but multiple repetitions were taken to ensure they were accurate.

The experiment was a success and the AVL Tree has proven to be a better datatype for this usage.

GitHub Log

See the below GitHub log for usage statistics and info.

```
52758b8e5cb83e5bbc467254c808ab10d2620111 danarent007
<danielvorster2009@gmail.com> 1492891894 +0200   commit: Added
param. input for repeated testing of vals.
52758b8e5cb83e5bbc467254c808ab10d2620111
e45e588906f24dc3239469df2e70deb599f1f9aa danarent007
<danielvorster2009@gmail.com> 1492891955 +0200   commit: Code
will not sout if param > 1
e45e588906f24dc3239469df2e70deb599f1f9aa
7dd431c24d216d0d3484b94722ddd8c36ea921fb danarent007
<danielvorster2009@gmail.com> 1492891989 +0200   commit: Added
old custom tree... For old times sake. Oh, and for the
SearchIt...
7dd431c24d216d0d3484b94722ddd8c36ea921fb
b0ed82aa6a354d4f778ca2ad79e4432b4cc8c75e danarent007
<danielvorster2009@gmail.com> 1492892104 +0200   commit:
InsertAVL now only inserts from file. Repeats according to param.
b0ed82aa6a354d4f778ca2ad79e4432b4cc8c75e
76f27edfd2fb9ca18e1f908f1da24942cf8897b9 danarent007
<danielvorster2009@gmail.com> 1492892188 +0200   commit: Query
file now accepts param, for automation.
76f27edfd2fb9ca18e1f908f1da24942cf8897b9
4ebb5b17bd1285635d29676a15de12beba67f491 danarent007
<danielvorster2009@gmail.com> 1492892747 +0200   commit: Added
timing.
4ebb5b17bd1285635d29676a15de12beba67f491
d4ec35d5b7b18134f7463a9dc8f642b51173d44b danarent007
<danielvorster2009@gmail.com> 1492892768 +0200   commit: Added
timing.
d4ec35d5b7b18134f7463a9dc8f642b51173d44b
bb293dd1b64e921d5dbb68c3ab8dc9edb7052875 danarent007
<danielvorster2009@gmail.com> 1492892834 +0200   commit: Added
timing.
bb293dd1b64e921d5dbb68c3ab8dc9edb7052875
f57407295c6f42de6d934b1823e25a1e7adf51c1 danarent007
<danielvorster2009@gmail.com> 1492892905 +0200   commit: Modified
SearchIt to InsertIt.
f57407295c6f42de6d934b1823e25a1e7adf51c1
97ad8aa0abee256f1f0d85a00de4891842a3b9a6 danarent007
<danielvorster2009@gmail.com> 1492892926 +0200   commit: Updated
Node.
97ad8aa0abee256f1f0d85a00de4891842a3b9a6
10a18699b51d63dd6470b5f52a7c63a03a280085 danarent007
<danielvorster2009@gmail.com> 1492927716 +0200   commit: Added /
updated header.
```

```
10a18699b51d63dd6470b5f52a7c63a03a280085
4396922ef8a09a461033f431f0576da488791e59 danarent007
<danielvorster2009@gmail.com> 1492927857 +0200   commit: Modified
code - now prints regardless of param (More accurate times).
4396922ef8a09a461033f431f0576da488791e59
1562c5e4e896523e48a2727c1bf09747ab46df06 danarent007
<danielvorster2009@gmail.com> 1492927958 +0200   commit: Updated
headers.
1562c5e4e896523e48a2727c1bf09747ab46df06
2da18f051f09c4c921d05753504f9b3fd9acbcab danarent007
<danielvorster2009@gmail.com> 1492930264 +0200   commit: Added
time output and counter for insertions + time taken for 1.
2da18f051f09c4c921d05753504f9b3fd9acbcab
2cb1cbe53288819749513305c5baefc9ddd0c861 danarent007
<danielvorster2009@gmail.com> 1492930269 +0200   commit: Added
time output and counter for insertions + time taken for 1.
2cb1cbe53288819749513305c5baefc9ddd0c861
7577ede104c2e8267e62a3b07d4f9bcfb2d6eb9f danarent007
<danielvorster2009@gmail.com> 1492930281 +0200   commit: Added
time output and counter for insertions + time taken for 1.
7577ede104c2e8267e62a3b07d4f9bcfb2d6eb9f
97060104d10ff34a3158e104dc38aaf85625276d danarent007
<danielvorster2009@gmail.com> 1492936209 +0200   commit: Casted
root to BinaryTreeNode
97060104d10ff34a3158e104dc38aaf85625276d
0b79c31a050b38f95b9606544a7c9cb7b8176487 danarent007
<danielvorster2009@gmail.com> 1492937601 +0200   commit: Trying
to fix compareTo...
0b79c31a050b38f95b9606544a7c9cb7b8176487
ffbdc71f1d28ed79cb8e9d1dc1c612fef535983c danarent007
<danielvorster2009@gmail.com> 1492937996 +0200   commit: Trying
to fix compareTo...
ffbdc71f1d28ed79cb8e9d1dc1c612fef535983c
d24832b91390df368146bd518ae3d21b2d703180 danarent007
<danielvorster2009@gmail.com> 1492948523 +0200   commit: Updated
driver class to do EVERYTHING.
d24832b91390df368146bd518ae3d21b2d703180
dfade0842f5c8b432cccc1a823a5e75d1b68d719 danarent007
<danielvorster2009@gmail.com> 1492948568 +0200   merge
origin/master: Merge made by recursive.

dfade0842f5c8b432cccc1a823a5e75d1b68d719
47a1a9a9a1676c331165110a0c02c49e9a07710b danarent007
<danielvorster2009@gmail.com> 1492949189 +0200   commit: Added
data files.
```