Computer Science 2A Assignment 1.

Problem

The main idea of the assignment is to strengthen one's understanding of Data Structures. The introductory levels of data structures.
We are given about 10000 names that include the address and the telephone or cellphone number. We have to first sort the names and print them tio the screen in alphabetical order; we do that with the help of a binary search tree that has been pre-made for the assignment. The second Is to add the names to the tree again, but this time to search for items and print them to the screen – A binary search. The third thing we were required to do is to add the names to an unsorted array (I used an ArrayList) and search through this array or ArrayList for the name – a linear search.

After we make our applications, we then have to make an experiment to make a speed comparison between the binary search and the linear search. The experiment also works on the speed difference when there are different number of items in the tree or ArrayList.

Application Design

The applications printIt and searchIt both make use of the Binary Search Tree, That was written for us. A person object named humanBeing was created; the object stores name, phone number and the address. The name that is stored will then be used when searching through the tree.
The printIt class uses the person object and the binary search tree to sort the lines in alphabetical order. The searchIt uses the person object and binary search tree to search for names in the tree. The searchItLinear uses the person object and an ArrayList object to do what searchIt does.
The person object has a compareTo method, the compareTo method compares the names only, because sorting the names by alphabetical order and assuming there are not duplicates.

PrintIt Output.

Abbott Alec |489-848-7299 |03707 Botsford Fork, Lima
Abbott Alexandria |318.679.5603 x712 |44812 Wilderman Mountain, Vallejo
Abbott Alia |507.340.1186 |76400 Barton Fields #044, Cerritos
Abbott Brando |602.992.4016 |02519 Zackery Village, San Mateo
Abbott Elwyn |788.603.8604 |88126 Bruen Common, Beverly Hills
Abbott Hosea |1-035-079-0176 x61480 |51832 Bayer Pass, Simi Valley
Abbott Ima |823.283.2198 x7192 |87191 Suite Z, Selma
Abbott Josh |822.752.1004 |27010 Sanford Center, Stanton
Abbott Leann |516-835-0116 |17296 Elta Crossroad #362, Newport Beach
Abbott Meda |1-117-789-3061 |18565 Suite B, Fountain Valley
Abbott Murray |1-654-279-2374 |22345 Runte Garden, Steubenville
Abbott Novella |297-763-2822 |32763 Langosh Route, San Diego
Abbott Rahsaan |(681)856-6604 x642 |90282 Haag Keys, Garden Grove
Abbott Sadye |(961)238-9093 |52000 Marques Loaf #288, Placentia
Abbott Santina |1-515-459-1556 |78469 Renner Mill, Agoura Hills
Abernathy Amparo |1-052-394-1236 x29668 |96179 Feil Tunnel #352, Canton
Abernathy Austyn |1-486-893-0367 |98827 Gerlach Pike Apt. 743, Apple Valley
Abernathy Catalina |1-331-934-0147 |14576 Harber Knolls, Riverside
Abernathy Chadd |(552)753-8320 x85031 |23694 Pier F, Tempe
Abernathy Cicero |(637)882-6835 x72457 |36296 Batz Walk, San Francisco

Search Query

Thabo Kopane
Sauer Meredith
O'Reilly Mallie
Bartell Deontae
Graham Raleigh
Mayert Cathy
Bauch Asia
Schmidt Matilda
Hartmann Mandy
Schaden Vernon
Little Noelia
Kreiger Tad
Kuhlman Alia
Green Foster
Paucek Mose
Gerhold Cristobal
Keeling Lulu
O'Conner Nedra
Lind Seth
Hermann Sophie

Search Query Output for searchIt.

Error !!Name not found!!
Sauer Meredith |1-155-932-2562 x738 |40327 Ocie Camp Apt. 117, Maywood
O'Reilly Mallie |(817)117-8903 x0797 |74173 Zakary Manors Suite 005, North Pole
Bartell Deontae |060-029-7336 x5402 |40854 Kenyon Tunnel, Selma
Graham Raleigh |235-082-6666 x166 |85277 Hackett Lake, Merced
Mayert Cathy |791-772-8120 x42168 |90125 Raven Circle #864, Downey
Bauch Asia |008.064.6393 x003 |11071 Murazik Spurs, Nogales
Schmidt Matilda |(619)315-9336 |80334 Walter Rue, Valdez
Hartmann Mandy |1-643-105-1210 |14519 Funk Station Suite 088, Columbus
Schaden Vernon |1-417-882-5517 x5439 |31370 Zula Freeway, Jeffersontown
Little Noelia |1-107-781-4467 x140 |97918 Ward Fords Suite 533, San Clemente
Kreiger Tad |322-975-3378 x706 |56941 Haylie Knolls, Sahuarita
Kuhlman Alia |063.398.9808 x87859 |16990 Schoen Extension, Carson
Green Foster |1-265-752-9982 x87443 |81530 Lazaro Pike, Lafayette
Paucek Mose |327.239.4036 x03681 |36902 Fay Harbor, San Juan Capistrano
Gerhold Cristobal |861-403-4036 x0562 |72476 Elroy Fort, Parma
Keeling Lulu |1-144-315-2499 |36647 Kenyon Corner Suite 450, Oxford
O'Conner Nedra |(876)957-2560 |38815 Lot 820, Downey
Lind Seth |676-583-2675 |19008 Bruen Way Suite 641, Rolling Hills Estates
Hermann Sophie |(731)262-5512 |65854 Runolfsdottir Parkways, Ontario

Search Query Output for searchItLinear

Name not found
Sauer Meredith |1-155-932-2562 x738 |40327 Ocie Camp Apt. 117, Maywood
O'Reilly Mallie |(817)117-8903 x0797 |74173 Zakary Manors Suite 005, North Pole
Bartell Deontae |060-029-7336 x5402 |40854 Kenyon Tunnel, Selma
Graham Raleigh |235-082-6666 x166 |85277 Hackett Lake, Merced
Mayert Cathy |791-772-8120 x42168 |90125 Raven Circle #864, Downey
Bauch Asia |008.064.6393 x003 |11071 Murazik Spurs, Nogales
Schmidt Matilda |(619)315-9336 |80334 Walter Rue, Valdez
Hartmann Mandy |1-643-105-1210 |14519 Funk Station Suite 088, Columbus
Schaden Vernon |1-417-882-5517 x5439 |31370 Zula Freeway, Jeffersontown
Little Noelia |1-107-781-4467 x140 |97918 Ward Fords Suite 533, San Clemente
Kreiger Tad |322-975-3378 x706 |56941 Haylie Knolls, Sahuarita
Kuhlman Alia |063.398.9808 x87859 |16990 Schoen Extension, Carson
Green Foster |1-265-752-9982 x87443 |81530 Lazaro Pike, Lafayette
Paucek Mose |327.239.4036 x03681 |36902 Fay Harbor, San Juan Capistrano
Gerhold Cristobal |861-403-4036 x0562 |72476 Elroy Fort, Parma
Gerhold Cristobal |1-334-679-6851 x50935 |56912 Ledner Plains, Thousand Oaks
Keeling Lulu |1-144-315-2499 |36647 Kenyon Corner Suite 450, Oxford
O'Conner Nedra |(876)957-2560 |38815 Lot 820, Downey
Lind Seth |676-583-2675 |19008 Bruen Way Suite 641, Rolling Hills Estates
Hermann Sophie |(731)262-5512 |65854 Runolfsdottir Parkways, Ontario

Design experiment.

1. The first experiment was to measure the general efficiency of the the Binary Search Tree in comparison to the Array or ArrayList. There is a file that is the search query, this acts as the input for searching the name.
   The system's internal time is used to measure the time taken to complete the query, the speed of each program in order words.
   I then tabled the average of 4 items 11 times. I ran the program 4 times and took the average 11 times. Then I drew a graph in excel to show the speed difference visually.

2. The second experiment was determining whether there is significant speed difference when searching through bigger trees in comparison to a smaller tree.
   I made 6 files, I extracted names to those file from the tesdata file – which I also used as the 7^{th} file. The names in the files are mutually exclusive and one of the files doesn't have any found search queries. My files had -:
   a. 20 lines to search from.
   b. 100 lines to search from
   c. 280 lines to search from
   d. 400 lines to search from
   e. 2000 lines to search from
   f. 2500 lines to search from.
   g. 10000, search testdata.

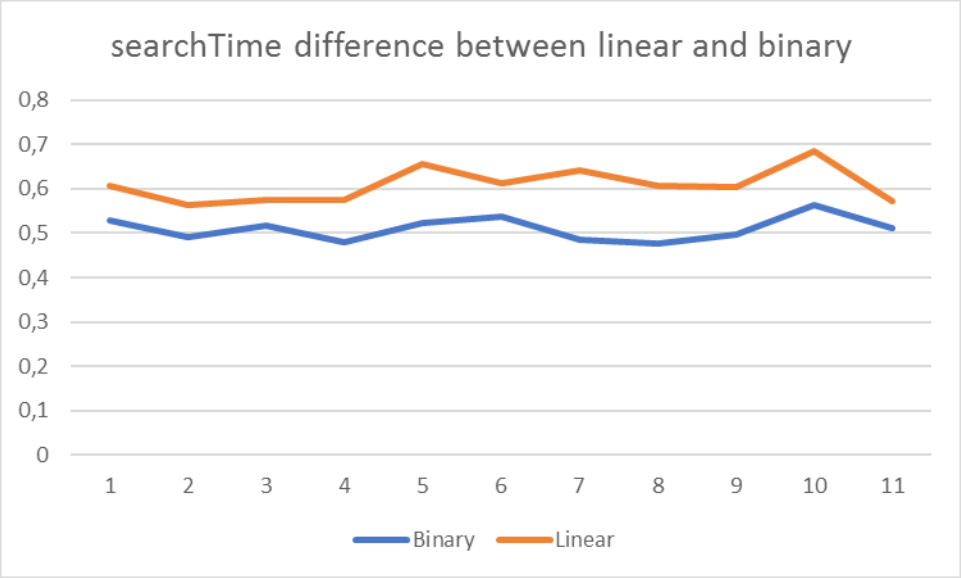
The null hypothesis was that the speed of either the search tree or the ArrayList is independent of the size of the array.

The alternative hypothesis was that the speed of the ArrayList is dependent of the size of the array.

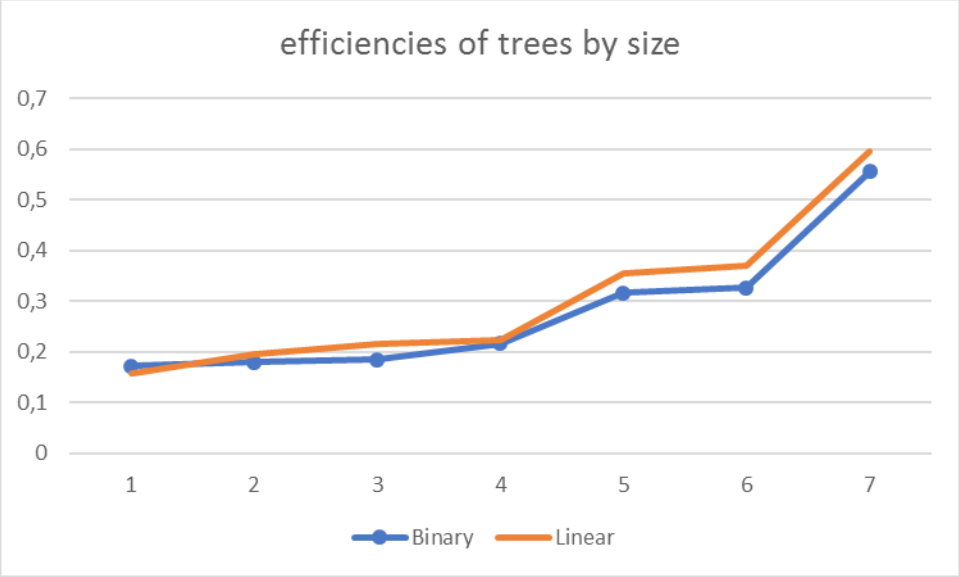
To test each hypothesis, the program was ran 4 times for each number of lines to 10000; the average was taken and the results were put into a graph. The graph helps us with visual display of the relative performance of the data structures individually and against each other.

Experiment Results

Experiment 1.



The one with lesser time is the fastest.
So with this experiment we can conclude that the binary search tree is faster than searching linearly.
The Binary Search Tree is consistently faster than linear search.



Experiment 2.

1. **We can safely reject the null hypothesis here, because the graph shows that there is significant speed difference when searching between a smaller number of items and a larger number regardless of the data structure.**

   This graph also shows that if there is a small data to search through, the speed difference is not significant. A faster searching data structure such as a Binary Search Tree is only required when dealing a large number of items. For example, when storing details of many individuals in a company.

Junit

I tested the person class.
JUnit version 4.12
.....
Time: 0.008

OK (7 tests)

Git