Linear and Binary Search Copy

Homework #1

By:

John Kiyak

CS303

September 4, 2018

1. Problem Specification

The goal of this assignment was to write a program that will allow the user to search a given array using both binary and linear search, and be able to compare the difference in speed between the two of them. The program first reads a text file that has 1000 random integers, select one of those integers as a “key”, and compares it to the arraylist compiled from the binary and linear methods. It then goes through the array, using the key, and compares the key to the arrays that are created. If any of the keys match one of the arrays, it lets the user know the location of the array that matches the key, as well as how long each search took. If not, the program prints out “Element not found”.

2. Program Design

The design of this program wasn’t too terribly complicated. On my lab, I used only two methods, one being my linearsearch method, and the other being my binarysearch method, both of which we worked on and tested during class. Afterwards, we create new objects from those classes, and instaniate them. Then, we make a scanner class which holds the (“input\_1000.txt”) that is placed in the project directory. I created another scanner class for user input, during which asks the user how big they want their array to be. The user enters how big they want their array to be, and that is used to generate an array that is catered to the user’s specified size, from 1 to whatever they entered. Afterwards, an ArrayList is created to hold the integers from the “Input\_1000.txt” document. A random method is used to pick a random integer from the text file that was placed into the array, and that number is outputted onto the console. At the end of the project, the methods for both linear and binary searched are activated, along with the times that they took to complete.

3. Testing Plan

For testing, I used the input for reach of the arrays that needed testing, from 16 all the way to 1000. By using different inputs and differing how big the arrays were, we are able to see which search method is faster as the arrays get bigger.

4. Test Cases

Here are some sample test cases:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Array Input Size | Linear Search | | Binary Search | |
|  | Value found: | Time Ellapsed: | Value Found: | Time Ellapsed: |
| 16 | Not present | 0 | Not Present | 0 |
| 32 | Not present | 0 | Not present | 0 |
| 64 | Not present | 0 | Not present | 0 |
| 128 | Not present | 0 | Not present | 0 |
| 256 | Not present | 0 | Not Present | 0 |
| 512 | Not present | 0 | Not Present | 0 |
| 800 | 136 | 0 | 136 | 0 |

5. References

For the linear and binary search methods, I used examples from geeksforgeeks.org to model my own methods. For generating a set of random numbers, writing a text file to an array list, and a refresher on the scanner classes, I used some examples from stackoverflow, which are listed below.

<https://stackoverflow.com/questions/26831340/generate-10-random-integers-storing-them-in-an-array-and-then-calling-a-method-t>

<https://stackoverflow.com/questions/19341594/reading-numbers-from-a-text-file-into-an-arraylist-in-java>

<https://stackoverflow.com/questions/19346965/get-random-number-from-arraylist>

<https://www.geeksforgeeks.org/linear-search/>