Implementing Linear and Binary Search
Lab #1

By
Joshua Matthews

CS 303 Algorithms and Data Structures
August 31, 2015

1. Problem Specification

The goal of this assignment was to implement the linear and binary search algorithms and test the time differences in each of their performances. We were given input files to test these algorithms with.

2. Program Design

This program required two functions, one for linear search and one for binary search. It also required a way to retrieve the data from the files and a way to test out both of the algorithms.

The following steps were required to develop this program:

a. I needed to implement the code to read in whatever file that was passed into the program.
b. I then setup the linear and binary search functions.
c. I implemented the logic to track time each algorithm would take.
d. I setup the program to take user input for looking for certain values.
e. I setup the program to execute both algorithms for the user when input is given.

The StdLib for ruby supplied all of the classes need to read in the files content and take in and respond to user input.

3. Testing Plan

The Lab01.rb ruby file tests both linear and binary algorithms. The file takes in an input file and parses out the numbers inside the file and pushes them onto an array to be used for testing in the algorithms. The initial values we retrieve in the file are random numbers from 0 to the number of elements that were in the file. The user would be prompt to enter a number to look for inside the array. The linear search is tested first, when it is done it will print out the value that you were looking for, the index it was found at, and it will display how long the search took in seconds. Once the linear search is done it starts the binary search with a new start and end timer to measure how long the binary search will take. It searches for the same number to stay consistent with what number both algorithms are looking for. Once done the program outputs what number was being looked for, what index it was found at, and how long the search took in seconds. If the user enters a value that does not exist in the array, both searches will output that the value was not found when the finish.

4. Test Cases

The test cases are shown below:

Test Case Number	Input File	Input Value	Output
1	input100.txt	24	Search for what number? 24 Linear search for 24 24 was found in array at index 41 Linear search took this long in seconds: 0.000111 Binary search for 24 24 was found in array at index 11 Binary search took this long in seconds: 8.6e-5
2	input_1000.txt	2	Search for what number? 2 Linear search for 2 2 was found in array at index 436 Linear search took this long in seconds: 0.000282 Binary search for 2 2 was found in array at index 2 Binary search took this long in seconds: 9.8e-5
3	input_5000.txt	84	Search for what number? 84 Linear search for 84 84 was found in array at index 2683 Linear search took this long in seconds: 0.000873 Binary search for 84 84 was found in array at index 79 Binary search took this long in seconds: 2.8e-5
4	input_10000.txt	811	Search for what number? 811 Linear search for 811 811 was found in array at index 6289 Linear search took this long in seconds: 0.002169 Binary search for 811 811 was found in array at index 814 Binary search took this long in seconds: 3.9e-5
5	input_50000.txt	34637	Search for what number? 34637 Linear search for 34637 34637 was found in array at index 23299 Linear search took this long in seconds: 0.005574 Binary search for 34637 34637 was found in array at index 34508 Binary search took this long in seconds: 2.7e-5

Test Case Number	Input File	Input Value	Output
6	input_100000.txt	63122	Search for what number? 63122 Linear search for 63122 63122 was found in array at index 26459 Linear search took this long in seconds: 0.006169 Binary search for 63122 63122 was found in array at index 63024 Binary search took this long in seconds: 2.1e-5
7	input_500000.txt	4051	Search for what number? 4051 Linear search for 4051 4051 was found in array at index 24712 Linear search took this long in seconds: 0.005972 Binary search for 4051 4051 was found in array at index 4060 Binary search took this long in seconds: 2.0e-5
8	input_1000000.txt	147772	Search for what number? 147772 Linear search for 147772 147772 was found in array at index 20771 Linear search took this long in seconds: 0.004323 Binary search for 147772 147772 was found in array at index 147903 Binary search took this long in seconds: 0.004586
9	input_5000000.txt	1076604	Search for what number? 1076604 Linear search for 1076604 1076604 was found in array at index 17374 Linear search took this long in seconds: 0.217432 Binary search for 1076604 1076604 was found in array at index 1075957 Binary search took this long in seconds: 0.005879
10	input_5000000.txt	0	Search for what number? 0 Linear search for 0 Linear search took this long in seconds: 0.754356 Item was not found Binary search for 0 Item not found Binary search took this long in seconds: 2.9e-5

5. Analysis and Conclusion

From what the test cases show, Binary search seems to be the quickest search algorithm in most cases. The linear search is quick if the element is towards the front of the array, but if not it could take as long as the array length to find it. The Binary search will keep cutting the search results of the array in half until it finds the element it needs which makes it really fast when searching through millions of values or even just thousands of values.

6. References

The www.ruby-doc.org website contains all documentation on how to use the objects from the StdLib. Input files were supplied by Abhishek Anand.