Brigham Campbell

Dr. Griffith

CS 2235

11 September 2019

Programming Assignment 01

PART 1

1. Note that $2n^3 - 7n^2 + 100n - 36 \leq (2 + 7 + 100 + 36)n^3 = cn^3$, for $c = 145$, when $n \geq n_0 = 1$. Therefore, the function is $O(n^3)$.

2. Note that $10n + 3log(n) \leq (10 + 3)n = cn$, for $c = 13$, when $n \geq n_0 = 1$. Therefore, the function is $O(n)$.

3. Note that $n/1000 \leq (1/1000)n = cn$, for $c = 1/1000$, when $n \geq n_0 = 1$. Therefore, the function is $O(n)$, not $O(1)$.

4. Note that $log(n)^2 + log(n)/30 \leq (1 + 1/30)log(n)^2$, for $c = 31/30$, when $n \geq n_0 = 2$. Therefore, the function is $O(log(n)^2)$.

5. Note that $n^2/log(n) + 3n \leq (1 + 3)n^2$, for $c = 4$, when $n \geq n_0 = 2$. Therefore, the function is $O(n^2)$.

6. The function is $O(n)$.

7. The function is $O(n^2)$.

8. The function is $O(n)$.

9. The function is $O(n)$.

10. The function is $O(1)$.

11. Function $m1FindLargest$ is $O(n)$ because its highest-order function is looping through each value in an array.

12. Function $m2PrintTriangle$ is $O(n)$ because its highest-order function is looping through each integer between 1 and the supplied integer.

13. Function *m3PrintBooks* is $O(n)$ because its highest-order function is looping through each value in an array.

PART 2

The following figure is the result of plotting the averages of the length of time it takes for four algorithms to complete. The averages have been computed by running each function 200,000 times per array size.
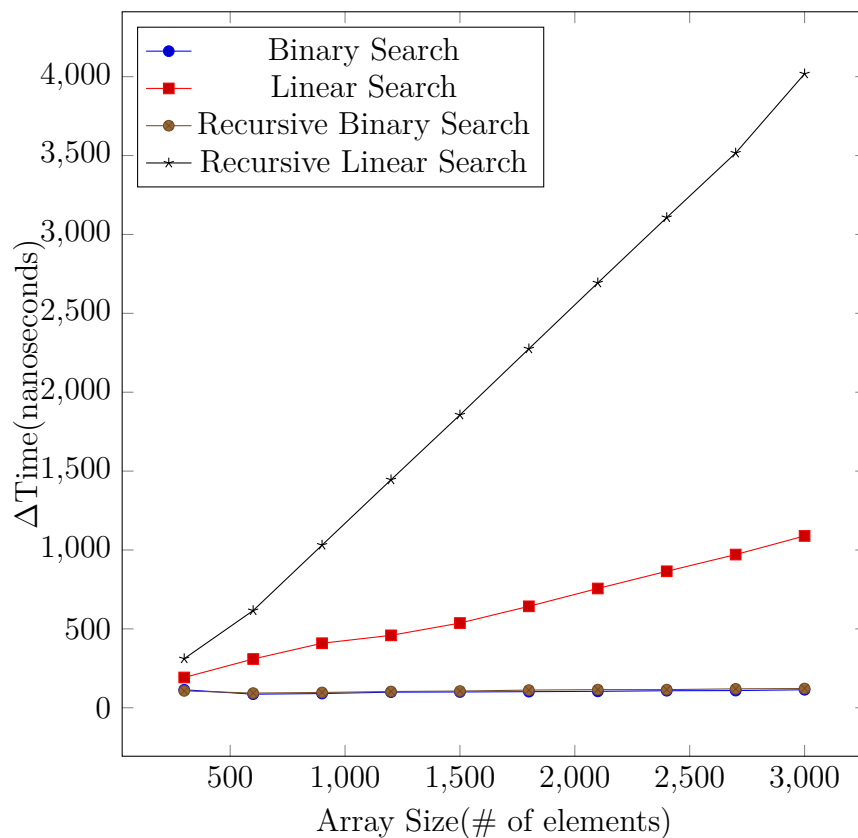
As the graph shows, Recursive Linear Search quickly becomes the slowest algorithm as its given larger arrays. By inspection, we see that Linear Search is in the same order as Recursive Linear Search, but is scaled such that it will take nearly one fourth the time that Recursive Linear Search will take for the same array size. This makes sense because rapid function calls impose significant overhead in Java.

Comparatively, both Binary Search and Recursive Binary Search take a very small amount of time to complete. This makes sense as Binary Search is $O(log(n))$ while Linear Search is

$O(n)$.

From these results, we can conclude that some recursive or linear implementation of Binary Search is more efficient than Linear Search when searching through sorted data.