Giselle Martel ID# 2652936
COMP 352
Dr. Stuart Thiel
May 15, 2018

Assignment 1

Question 2

The insertion sort algorithm was tested 2000 times on 100 unique arrays (incrementally increasing in length from 10 to 1000). Analysis of average runtimes from this test indicates an operation cost of $O(n^2)$ as expected. The quadratic trend lines in figure 1&2 below suggest a complexity of $O(n^2)$ in the worst case (reverse sorted arrays) and $\theta(n^2)$ in the average case (unsorted randomized arrays). The linear trend line in figure 3 suggests a complexity of $\Omega(n)$ in the best case (sorted array).

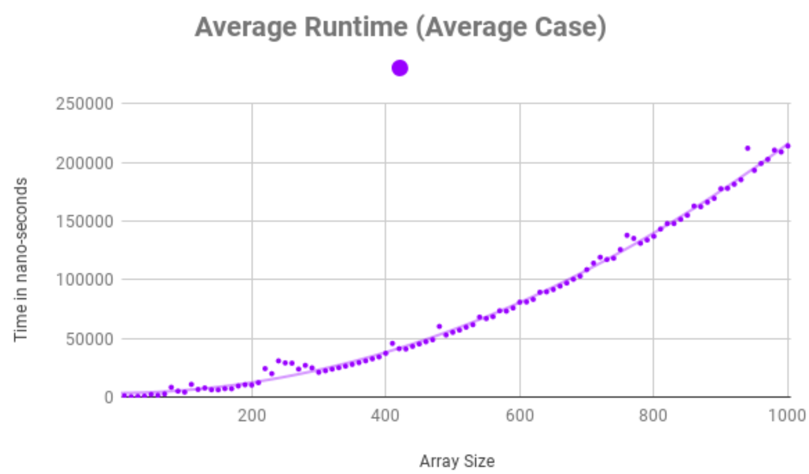

figure 1*: random values (unsorted)*

figure 2*: reverse sorted values*

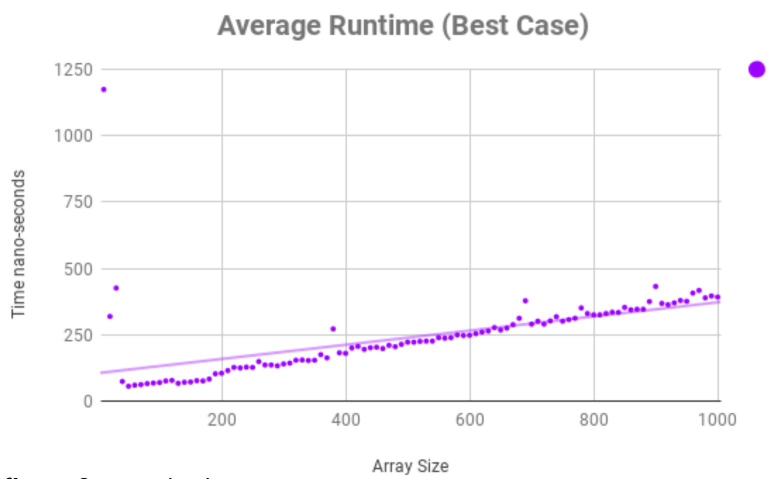

figure 3: *sorted values*

Data Table

Insertion Sort Runtime Average Case

Size of Array Tested	Average Execution Time (ns)	# of Runtimes
100	4174	2000
200	10204	2000
300	21089	2000
400	37291	2000
500	55188	2000
600	80891	2000
700	108606	2000
800	137023	2000
900	177600	2000
1000	214088	2000

figure 4*: Data table of average runtimes for the average case test (only data from 10 arrays shown for simplicity)*

Question 3

With binary search, complexity is logarithmic where the algorithm searches for the location that the value being compared should be placed. With traditional insertion sort the complexity is linear when finding this location, meaning a higher operation cost. However, because the inner loop in binary search would have a complexity of O(n) due to shifting of remaining elements, it would not be any more efficient than insertion sort since total operation cost would also be quadratic.