**CSE 212 – Programming with Data Structures**

**W02 Prove – Response Document**

|  |  |
| --- | --- |
| **Name:** |  |
| **Date:** |  |
| **Teacher:** |  |

*It is a violation of BYU-Idaho Honor Code to post or share this document with others or to post it online. Storage into a personal and private repository (e.g. private GitHub repository, unshared Google Drive folder) is acceptable.*

**Question 1: From Part 1, what is the big O notation for the SortArray function? O(n)**

**Question 2: From Part 1, what is the big O notation for the StandardDeviation1 function? O(n)**

**Question 3: From Part 1, what is the big O notation for the StandardDeviation2 function? O(n^2)**

**Question 4: From Part 1, what is the big O notation for the StandardDeviation3 function? O(n^2)**

**Question 5: Put the following big O notations in order from best performance to worst performance: O(n^2), O(1), O(2^n), O(n log n), O(log n), O(n).**

O(1), O(log n), O(n), O(n log n), O(n^2), O(2^n)

**Question 6: From Part 2, what is the performance (using big O notation) for the SearchSorted1 function? O(log n)**

**Question 7: From Part 2, what is the performance (using big O notation) for the SearchSorted2 function? O(n^2)**

**Question 8: From Part 2, which function (SearchSorted1 or SearchSorted2) has the better performance? O(2^n)**

**Question 9: From Part 2, for both functions (SearchSorted1 and SearchSorted2), explain in detail how you determined the big O notation by just looking at the code without the benefit of observing actual execution results? I looked at the loops and decided if it ever got divided.**

**Question 10: From Part 2, it is possible in the best case for each of these functions (SearchSorted1 and SearchSorted2) to complete in O(1) time even if the size of the list was very large. What input scenarios would give this result for both functions? I would say that the input would be how many loops get put within each other or inside of each other = times them selves.**