David Tran

CS 3310

Prof. Davarpanah

Due: 10/07/20

Elapsed run time of Bubble Sort:

Elapsed time to sort array of 10 elements:

3400 ns or 0.0034 ms

Elapsed time to sort array of 100 elements:

131600 ns or 0.1316 ms

Elapsed time to sort array of 1,000 elements:

3205200 ns or 3.2052 ms

Elapsed time to sort array of 10,000 elements:

137640000 ns or 137.64 ms

Elapsed time to sort array of 100,000 elements:

11006038100 ns or 11006.0381 ms

Analysis of Time Complexity of Bubble Sort:

Average case:

In Bubble Sort, n - 1 comparisons are done in the first pass, n - 2 in the second pass, n - 3 in the third pass, and so on until the array is fully sorted.

The total number of comparisons is:

(n - 1) + (n - 2) + (n - 3) + … + 3 + 2 + 1

= □ $n(n-1)/2$, where n starts at 0

This results in O(n^2).

Therefore, the average case time complexity is O(n^2).

Best case:

The best case for Bubble Sort is when the array is already sorted.

In this case, the algorithm checks N number of elements to see if any swaps of adjacent elements are necessary.
Therefore, the best case time complexity of Bubble Sort is O(n).

Worst case:
The worst case for Bubble sort is when the array is sorted in reverse order.
The same number of operations is done as the average case
(n - 1) + (n - 2) + (n - 3) + … + 3 + 2 + 1
= □ $n(n-1)/2$, where n starts at 0
This results in O(n^2).
Therefore, the worst case time complexity is O(n^2).

Analysis of Space Complexity (Memory Usage):

In Bubble Sort, all storing and swapping operations done to the elements in the array are done on the original data set. Regardless of the size of the original array, the amount of memory overhead is constant as additional memory allocation is not needed. The original data set has a space complexity of O(n). In Bubble Sort, only a single additional memory space is required for a temporary variable in the swapping process. Therefore, the space complexity of Bubble Sort is O(1).