CS 335: ALGORITHMS
FALL 2022
QUIZ 2

Assigned: Tuesday, October 4, 2022 @ 12:30 pm
Due: Tuesday, October 4, 2022 @ 11:59 pm (No Extensions)
Max Score: 50 points

Student Name: _____

1) Complete a detailed Asymptotic Analysis of below-shown program.
   Compute the space complexity.
   Compute the best, average, and worst run-time complexity.
   What data scenario would respectively be needed for the best, average, and worst run-time complexity?

   Show all your work clearly and in a step-by-step manner, instead of just showing your final answer. I should clearly understand the flow of how you obtained your solution.

   Provide all your work in this Word document.

   Space complexity is O(n), as the memory usage increases linearly with the size of the array someMethod takes as a parameter. (Looking at the array as an input, being added to auxiliary space to find space.) Ignoring the array as input, space is O(1), as the algorithm itself does not utilize more space.

   The algorithm has 2 for loops, 1 of them being nested. This leads to a worst-case run-time complexity of O(n^2) if there were a theoretically infinite number of elements in the array. This would be effectively achieved through big-data scenarios, or near-infinite elements.
   Similarly, in the average case, the complexity is O(n^2). This can be calculated by taking all random inputs and taking the computation time for those random inputs with All random cases / Total # of random cases. A scenario similar to your database would represent the average.

   The best case complexity is O(1), running the algorithm on only a single element. (1^2 is 1). As a scenario, you are [for some reason] attempting to sort a single element.

```
class SomeClass {
   void someMethod(int arr[])
   {
      int n = arr.length;
```

```
    for (int i = 0; i < n - 1; i++){

        for (int j = 0; j < n - i - 1; j++){

            if (arr[j] > arr[j + 1]) {
                int temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }

        } // end of inner loop

    } // end of outer loop
}
```