Kyle Stoltzfus

4.16 O(n)

The addition statement inside the for loop is executed for all the elements of the array of size *n*. Therefore, the time required for the algorithm to execute is based on the size of the array. As a result, the algorithm is linear.

4.17 O(n)

Although the addition statement inside the for loop only executes for even cells of the array, this algorithm is still linear. For example, an array of 400 elements will execute the addition statement twice as often as an array with 200 elements. Therefore, it will take twice as long to run this algorithm on an array of 400 elements as it will on an array of 200 elements.

4.18 O(n²)

This algorithm is quadratic. For its worst case, the addition statements in both for loops are executed. Since the second loop is nested in the first one, the time the outer loop takes to execute must be multiplied by the time the inner loop takes to execute to determine the running time of the entire expression.

4.19 O(n)

Both addition statements execute for every repetition of the for loop. As a result, the time to run this algorithm is dependent on the size of the array it is being used on. Therefore, this algorithm is linear.

4.20 O(n³)

During every repetition of the outer for loop, the two nested for loops also are executed. For this algorithm’s worst case, each for loop executes a statement which takes a certain amount of time. These three times must be multiplied together to get the total running time of the algorithm.