Lab 6 Big-O analysis

1) Set n = manyItems. This analysis will find the amount of operations for the method as a function of n.

Line 3 takes about 2 operations, finding data.length then comparing it to manyItems. Line 5 takes one operation, setting array. At Line 7 takes about 3 operations for multiplying and adding manyItems then setting it to biggerArray.

The amount of operations depends on the manyItems or n. The loop will iterate n times.

For each iteration there are approximately 3 operations iterating each time, comparing manyItems, incrementing I, and setting biggerArray.

After the for loop there are 3 more operations leaving the total number of operations to be

6 + 3n + 3 = 3n + 9 as the worst case with Big-O being O(n).

2) N is set to manyItems to count the iterations of the for loop. The main operations will be through Lines 7- 9.

The first operations begin with creating int variables, 1 operation for line 3, 1 for line 4, and one for line 6.

For every iteration of the for loop(lines 7-9) there are between 2-5 operations. Finding the worst case will consider the 5 operations.

After the loop is finished there is one operating, return answer. The total operation count will end up being 3 + 5n + 1 = 5n + 4 for the worst case with Big-O as O(n).

3) This method runs through a lined list, so n will be the amount of iterations through the list. The loop begins at lines 10 with operations before and after it.

Lines 3 and 4 both take one operation for creating variables IntNode and int.

Lines 6 and 7 take one operation each as well, one for comparing position to zero and the other for throwing an argument. Line 7 may not always iterate but will be considered for the worst case run time.

Line 9 takes one operation for setting cursor to equal head.

Line 10 and 11 will take 4 operations for each iteration of the for loop, 1 to compare I, 1 to compare cursor, 1 to increment I, and one to set cursor to next link.

After the for loop one more operation is taken by returning cursor. This leaves the worst case run time to be 5 + 4n + 1 = 4n + 6 the Big-O being O(n).

4) This method also runs through each number in the linked list leaving n to be the number of iterations that are taken to get to the end of the list. The for loop begins at line 6.

Lines 3 and 4 take one operation each for creating variables cursor and answer

The amount of operations for the for loop at line 6 is n and will generate 3 operations each iteration leaving to be 3n. One operation for comparing cursor, 1 for setting cursor to next link, and one for incrementing answer.

After the for loop one operation takes place, return answer. This leaves the worst case run time to be

2 + 3n + 1 = 3n + 3. with Big-O being O(n).