Manuel Melendez – 80542914

Week 7 Lab

Performance of sortOfSort method

The method sortOfSort receives an array as input and sorts it as specified in the lab instructions. The following is the number of instructions performed then the method is called (to make things simple, I did not copy the whole instruction, but I will refer the line number and the instruction performed):

line 5: if to check length of array = 1

line 6 return nothing = 1

lines 9,12,13,16, and19: initialize and declare counter,

startIndex, endIndex, max, and lastIndex = 5

line 21: for loop = n

initialize and declare i = 1

line 24 initialize and declare max with a call to findMax = n

line 26 if lastIndex = 1

line 28 call to swapIndex = 3

line 30 endIndex -- = 1

line 34 else

line 35 call to swapIndex = 3

line 36 startIndex++ = 1

line 40 counter++ = 1

line 44 if counter > 1 = 1

line 45 counter = 0 = 1

line 46 lastIndex = !lastIndex = 1

Helper methods:

findMax

line 54: if checking startIndex == endIndex = 1

line 55: return startIndex = 1

line 56: initialize and declare max = 1

line 58: for loop = (endIndex-startIndex times)

\*Inside for loop\*

line 58 if checking arr[i] > max[0] = 1

lines 59 and 60 assigning values to max[0] and max[1] = 2

line 63 return max[1] = 1

swapIndex

line 68 initialize and declare temp = 1

lines 69 and 70 assign valies to arr[a] and arr[b] = 2

The best cases for this method are when it receives at empty array or an array of length 1 as there is nothing to sort or what is in the array is technically already sorted, respectively. In both of these cases, the only instructions executed are the if and return from lines 5 and 6, meaning that there are only two instructions executed. Therefore, the time complexity in these two cases is O(1) as it is constant and will always perform only these two instructions in these two cases.

There is no worst case for this method as it will always go through the whole array if its length is larger than 2