Goal: Modify the 4 sorting algorithms presented in Chapter 23 so that they keep count of the number of swaps made to completely sort a given array. Then, write a new application that creates four identical arrays of 20 integers and calls the 4 sorting algorithms on them, still displaying the number of exchanges made by each algorithm

BubbleSort

* A bubble sort will perform a swap when the target/index value is greater than the value to the right of it, this is where the counter should be increment by one
  + If ( list[i] > list [i + 1])
  + \*arguments + swap\*
  + count++;

InsertionSort

* An insertion sort will perform a swap when the value to the left of the target/index value is greater than the target/index value itself. The counter should be implemented in the for loop where the comparison between these two values is made.
  + for(left index; left index >= 0 && array[left index] > target index; left index--)
    - swap
    - count++;

MergeSort

* A merge sort will perform a swap after it has broken the array down in to smaller portions and sorted those portions, bringing them back as two sorted halves. So, the counter should be implemented in the merge method which brings the two halves together. The program will use setter and getter methods to manage the count so the setter method should be called after the counter is implemented to update the value after the count increases.
  + If(half1[i1] < half2[i2])
    - Temp[i3++] = half1[i1++]
    - count++
    - setCount(count);

QuickSort

* A quick sort will perform a swap when the left/low value is greater than the right/high value. Because of this, the counter should be implemented within the if statement which compares the two values as well as within the if statement which finally places the pivot/target value in the correct placement in respect to the two previous values. The count MUST be implemented before the return statement when the pivot is being placed otherwise it will become unreachable.
  + If(high > low)
  + Swap
  + Count++;

And

* + If(pivot > high)
  + Swap
  + Count++;