

APCS – Mr. Ascione

Sort Analysis Assignment:

Spec: In this project we will be analyzing and comparing the relative efficiency of the Sequential Sort, Insertion Sort and Merge Sort algorithms. We need to carefully consider what the initial data looks like – the number of items and how they are sorted / unsorted can definitely make an impact. Our goal is to analyze these algorithms – and print the results in a neatly formatted table.

1. On Google classroom you will find 6 different sets of test data to be sorted.

Important: *****Don't forget that once you sort an array you have modified the array itself, we need to give each sort its own copy of the data to work on from scratch. In this case, I'm going to give you the client so you won't have to worry about it.

2. Next, we want to do some analysis on each sort to compare how effective each sort is on different sets of data. We'll take the same type of approach we did for Searching - create a new class to keep track of comparison data. In particular we are looking to evaluate millis, comparisons, and swaps. Create a class called SortResult that stores this data. Modify your APSorts class so that each sort returns a SortResult object.

3. Look at the TimeExample class and use the same approach to time each sort. We will use counters to each sort to keep track of comparisons and swaps, consider this carefully.







4. For each set of data obtain the following:

Sort	Type of Data	Number of Elements	Time(millis)	Comparisons	Swaps
Bubble	Ordered	100
...					

9. Complete this analysis for each sort on each set of data. Again, you need to make sure that each sort gets a new copy of the original data set – not an already sorted copy, that would not be fair!

10. After you have collected all of your data, show your findings in a neatly formatted table at the bottom of your code and write a summary analysis of your results. Which sorts were most efficient and on which data. What are your conclusions?

Files you will need:

`100RandomInts`
`100ReversedInts`
`100SortedInts`
`10000RandomInts`
`10000ReversedInts`
`10000SortedInts``APSorts.java`
`SortClient.java`
`SortResult.java`