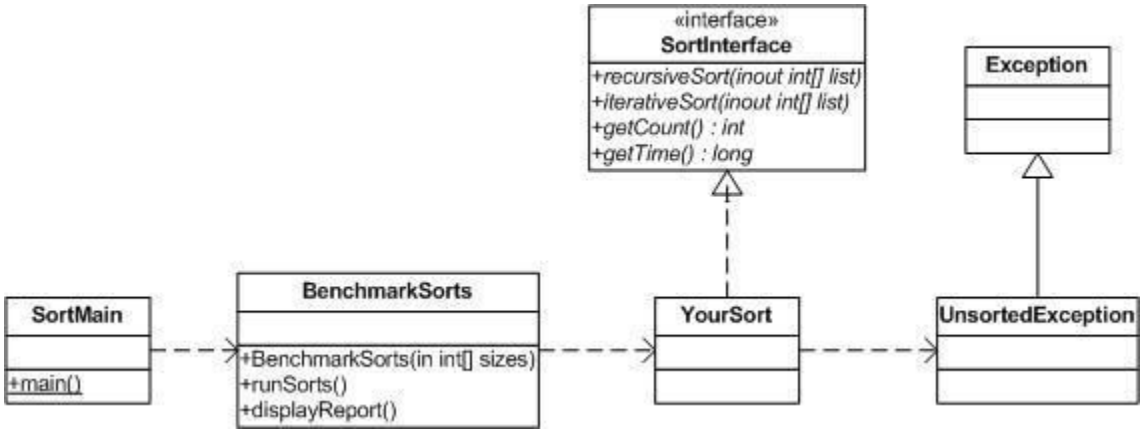
CMSC 451 Project 1

The project involves benchmarking the behavior of Java implementations of one of the following sorting algorithms, bubble sort, selection sort, insertion sort, Shell sort, merge sort, quick sort or heap sort. You must post your selection in the "Ask the Professor" conference. No more than five students may select any one algorithm.

Project 1 involves writing the code to perform the benchmarking of the algorithm you selected. Your program must include both an iterative and recursive version of the algorithm. You do not have to write them yourself, you may take them from some source, but you must reference your source. You must identify some critical operation to count that reflects the overall performance and modify each version so that it counts that operation. In addition to counting critical operations you must measure the actual run time. You are to write code to determine their efficiency based on the number of times that the critical operation is executed and actual time measurements. In addition, you should examine the result of each call to verify that the data has been properly sorted to verify the correctness of the algorithm. If the array is not sorted, an exception should be thrown. It should also randomly generate data to pass to the sorting methods. It should produce 50 data sets for each value of *n*, the size of the data set and average the result of those 50 runs. The exact same data must be used for the iterative and the recursive algorithms. It should also create 10 different sizes of data sets. Choose sizes that will clearly demonstrate the trend as *n* becomes large. You should also calculate the standard deviation of the critical operation counts and time measurement for the 50 runs of each data set size as a way to gauge the data sensitivity of the algorithm. Your program must be written to conform to the following design:



Your output should look at follows:

Data Set Size n	Iterative	Recursive

	Average Critical Operation Count	Standard Deviation of Count	Average Execution Time	Standard Deviation of Time	Average Critical Operation Count	Standard Deviation of Count	Average Execution Time	Standard Deviation of Time
100								
200								

The data set sizes above are examples. You are to select the actual data set sizes. On the due date for project 1, you are to submit a .zip file that includes the source code of your complete program. All the classes should be in the default package.

Include a small description of the following:

- A Big-Θ analysis of the two versions of the algorithm
- A discussion of the results of your study, which should include
  - graphs of your results
  - a comparison of the performance of the two versions of the algorithm
  - a comparison of the critical operation results and the actual execution time measurements
  - a discussion of the significance of the standard deviation results and how it reflects the data sensitivity of your algorithm
  - how your results compare to your Big-Θ analysis

Grading of the project will be based on the following items:

Adhered to the specified design
Produced the required output in the specified format Correctly calculated the statistics
Chose good test sizes and good random data Correctly implemented the sorting algorithm