Zhijie Li (001563872)

**Program Structures & Algorithms**

**Fall 2021**

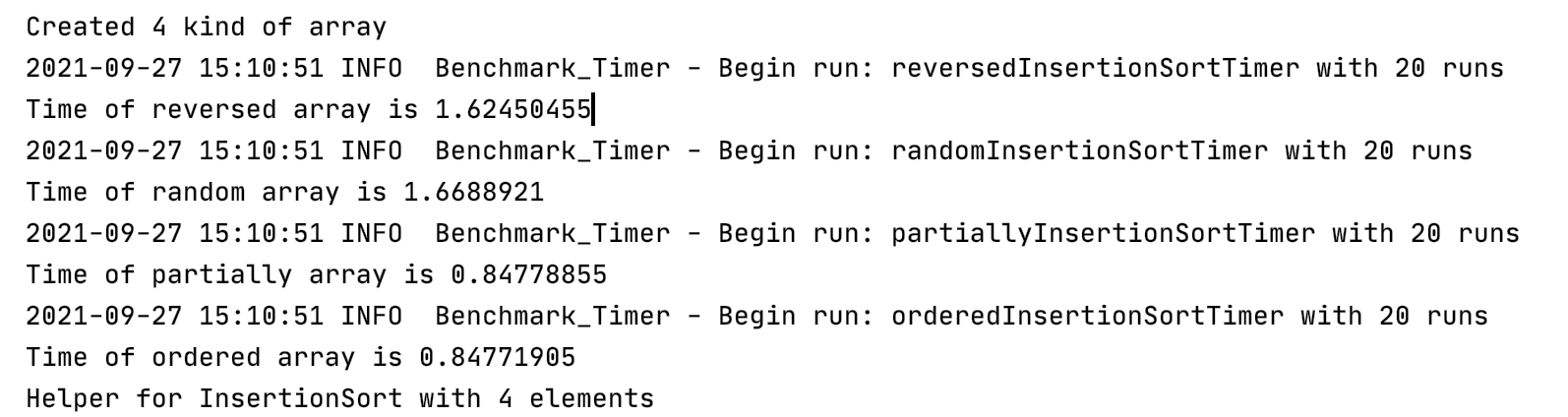
**Assignment No. 2**

* **Task**
* (Part 1) You are to implement three methods of a class called *Timer*. Please see the skeleton class that I created in the repository. *Timer* is invoked from a class called *Benchmark\_Timer* which implements the *Benchmark* interface.
* (Part 2) Implement *InsertionSort*(in the *InsertionSort* class) by simply looking up the insertion code used by*Arrays.sort.* You should use the *helper.swap* method although you could also just copy that from the same source code. You should of course run the unit tests in *InsertionSortTest*.
* (Part 3) Implement a main program (or you could do it via your own unit tests) to actually run the following benchmarks: measure the running times of this sort, using four different initial array ordering situations: random, ordered, partially-ordered and reverse-ordered. I suggest that your arrays to be sorted are of type *Integer*. Use the doubling method for choosing *n*and test for at least five values of *n.*Draw any conclusions from your observations regarding the order of growth.
* **Relationship Conclusion: (For ex : z = a \* b)**

In different N-elements experiments, the running time of “reverse” > “random”>”partially” ~ to “ordered”.

* **Evidence to support the conclusion:**

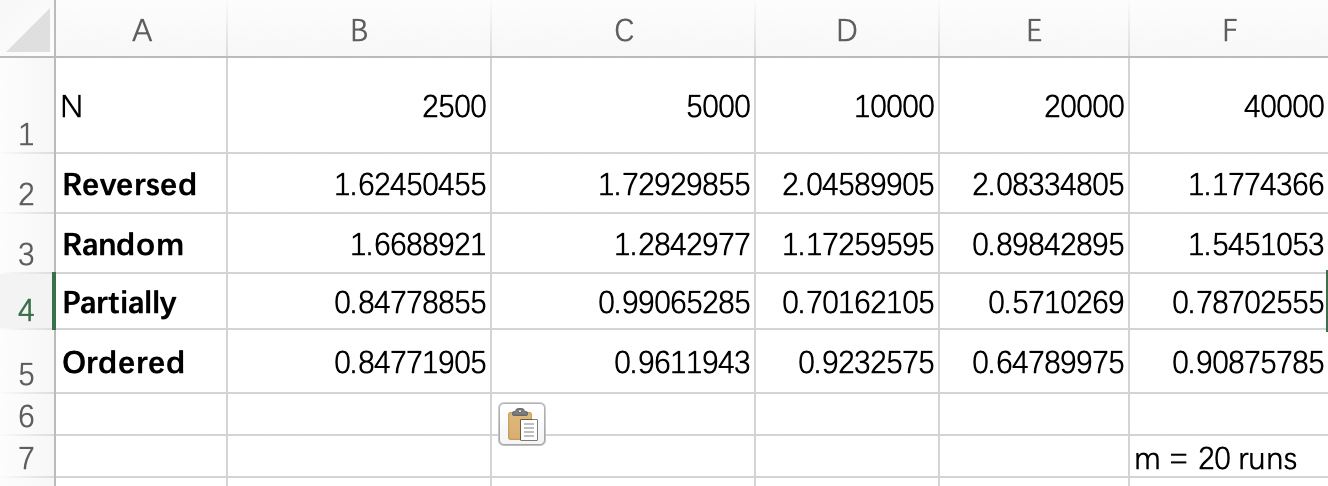
1. **Output (Snapshot of Code output in the terminal)**

****

**In the case of N=2500**

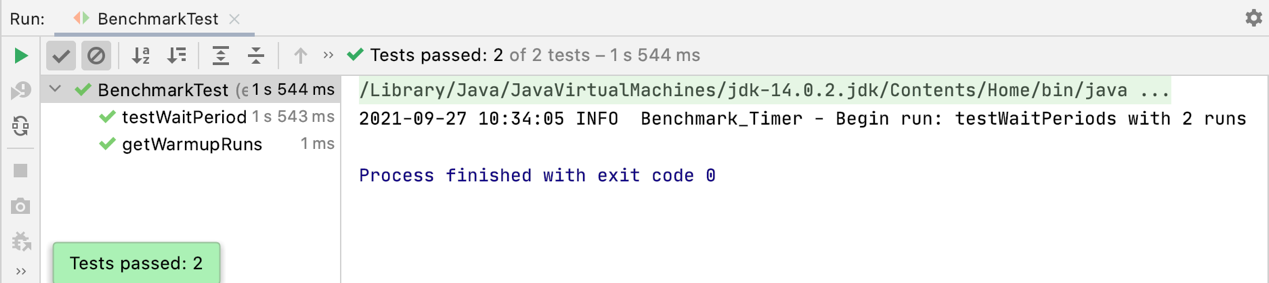
1. **Graphical Representation(Observations from experiments should be tabulated and analyzed by plotting graphs(usually in excel) to arrive on the relationship conclusion)**

Part3

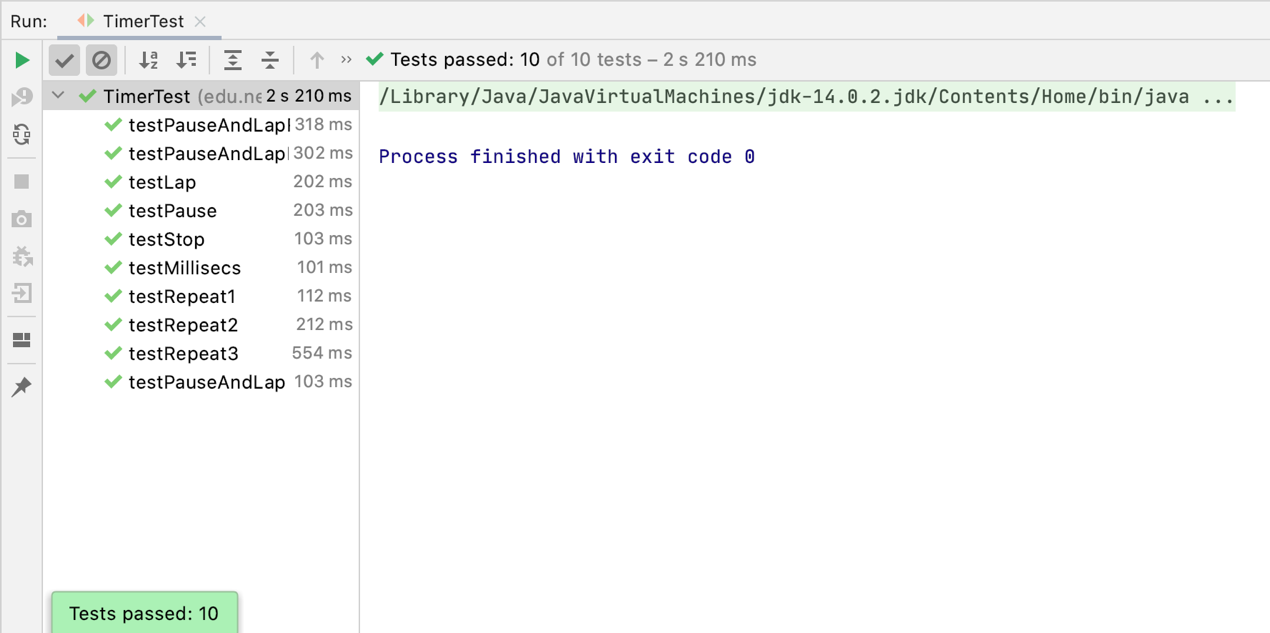


* **Unit tests result:(Snapshot of successful unit test run)**

BenchmarkTest



TimerTest



InsertionSortTest

