Kan Zhang (001529338)

**Program Structures & Algorithms**

**Spring 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.
* **Output**

Part3 Result

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | N=5000 | N=10000 | N=20000 | N=40000 | N=80000 |
| Reversed | 0.06571897 | 0.18453579 | 0.19580160 | 0.29446817 | 0.74291393 |
| Random | 0.05124713 | 0.10692083 | 0.19352205 | 0.3588002 | 0.70911917 |
| Partially | 0.04585155 | 0.09960218 | 0.15763254 | 0.33111634 | 0.71412152 |
| Ordered | 0.03807066 | 0.06664922 | 0.13814873 | 0.25452497 | 0.50889162 |

* **Relationship Conclusion:**

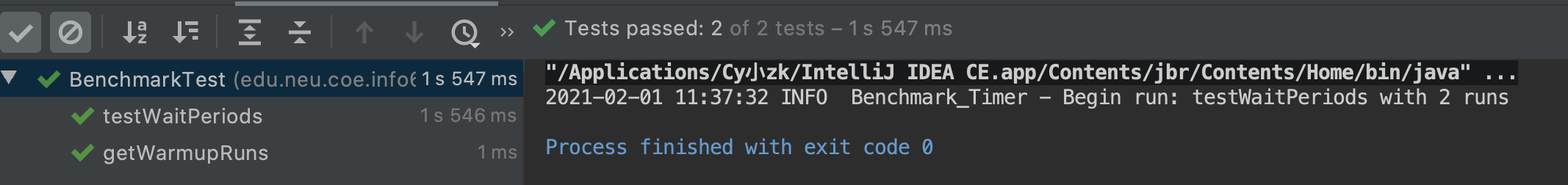
With N increase, the running time of sort goes up as well, nearly linear relationship based on “ordered” array. In most cases the running time of “reversed” > “random” > “partially” > “ordered”. But it could happen differently like when N = 40000.

* **Evidence to support the conclusion:**

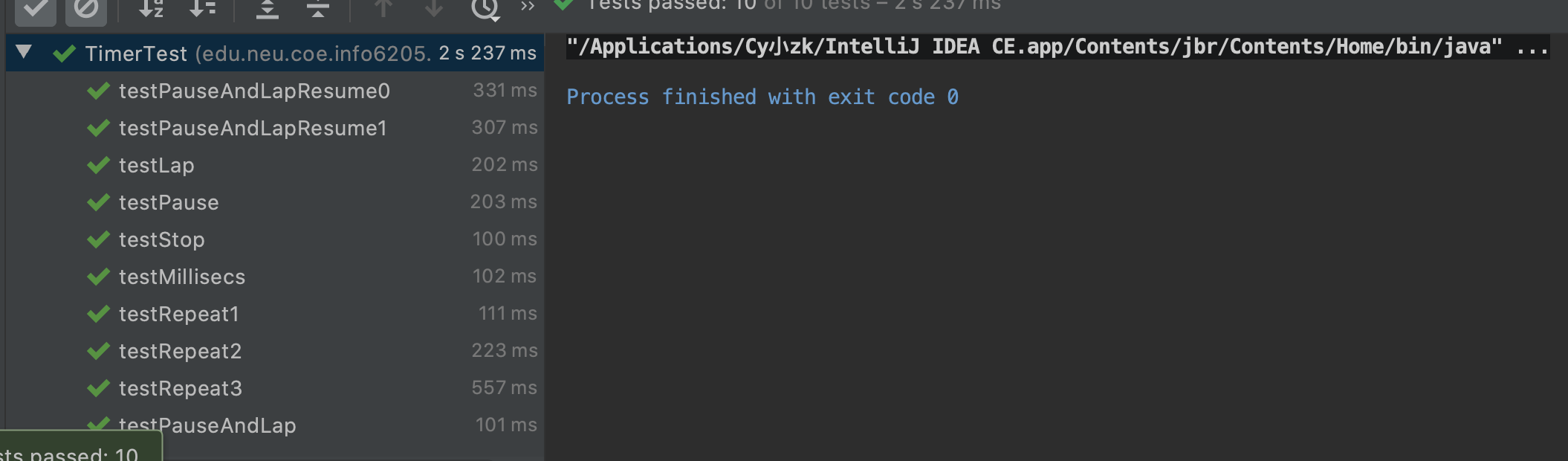
The output of running time is the evidence.

* **Unit tests result:**

BenchmarkTest



TimerTest



InsertionSortTest

