Yang Song (001003647)

Program Structures & Algorithms

Fall 2021

Assignment No. 2

- **Task (List down the tasks performed in the Assignment)**
    1. Implement three methods of a class called Timer
    2. Implement InsertionSort.java
    3. Implement a main program 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
- **Relationship Conclusion:**

  f(ordered) = O(N)

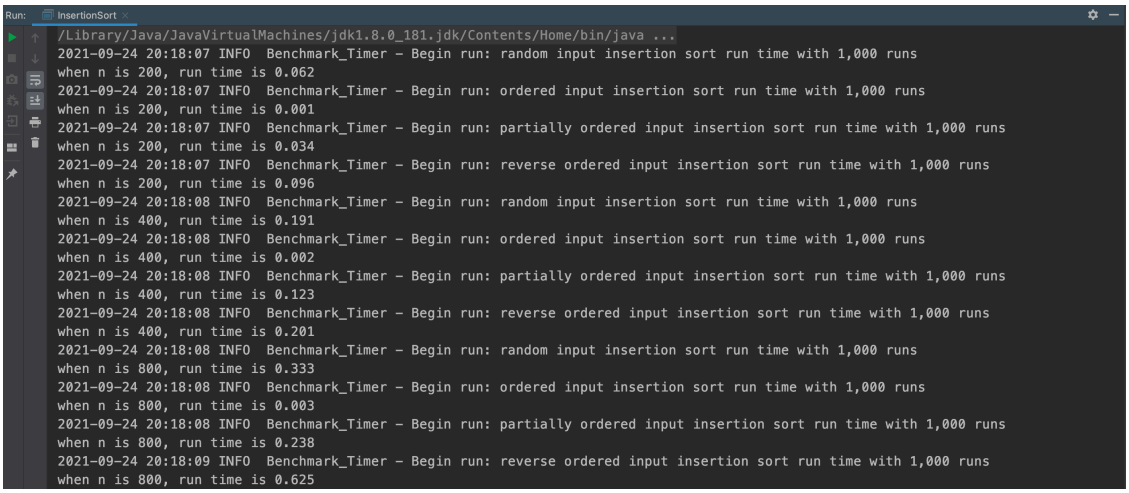
  f(reverse-ordered) = $O(N^2)$

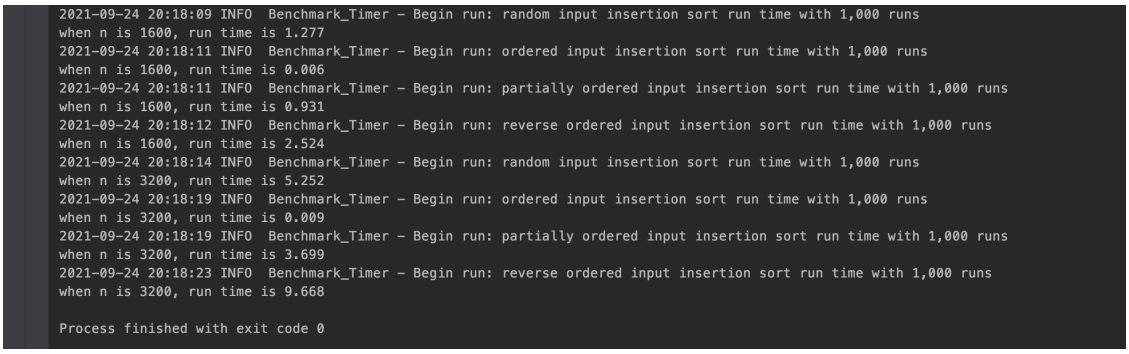
  f(ordered) < f(partially ordered) < f(random) < f(reverse ordered)

- **Evidence to support the conclusion:**

1. **Output**

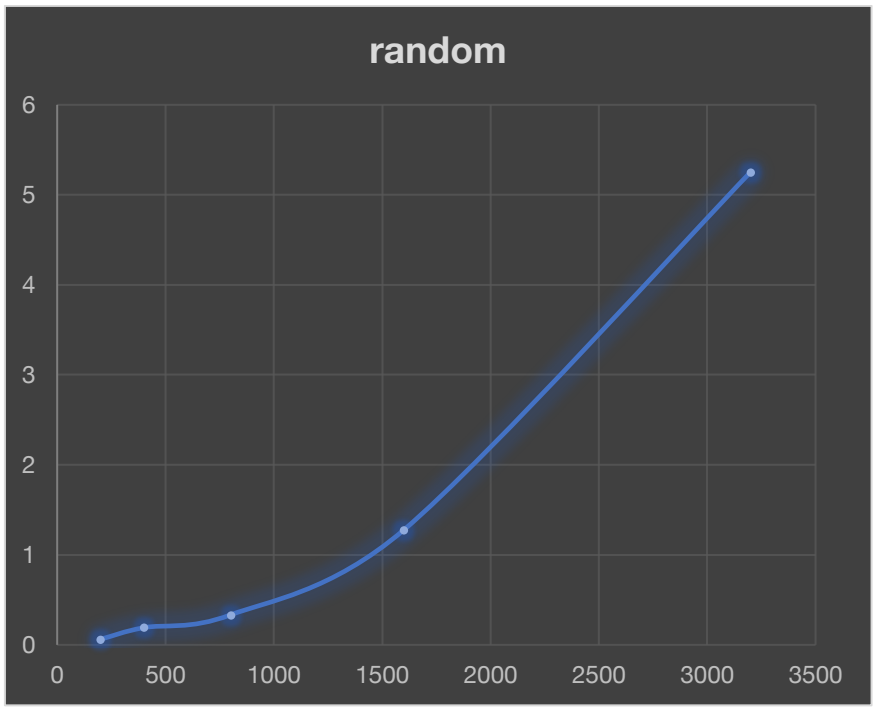
   Run main program in InsertionSort.java and get mean run time of different type of input arrays with doubling method

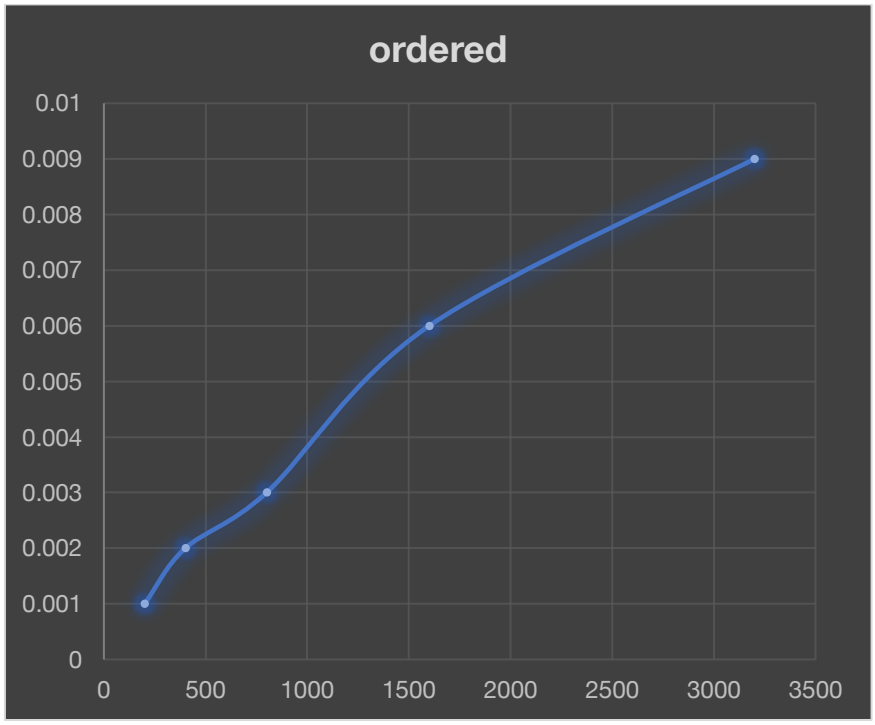


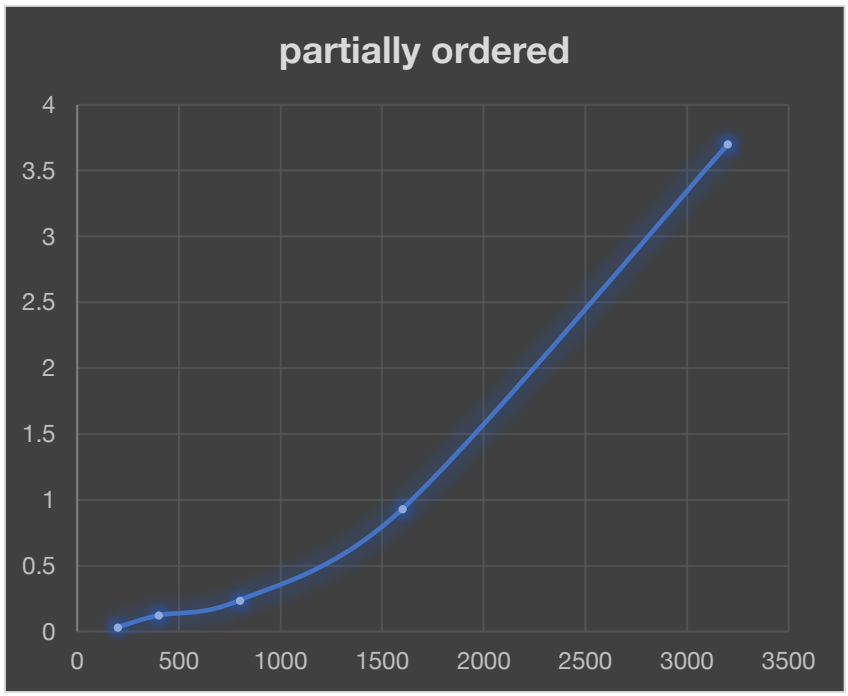


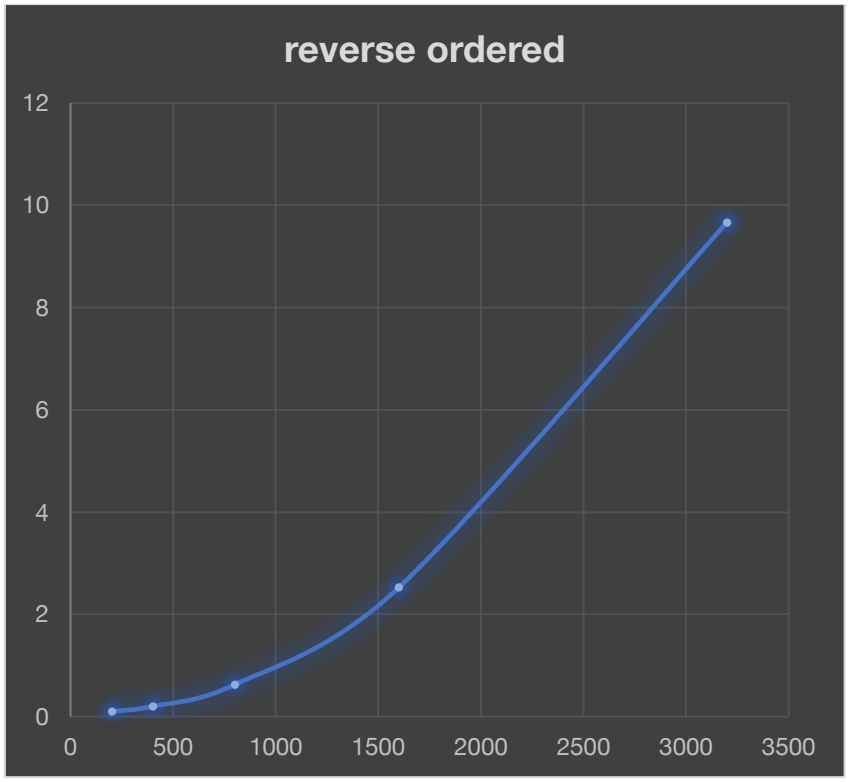
2. **Graphical Representation**

A	B	C	D	E
type / n	random	ordered	partially ordered	reverse ordered
200	0.062	0.001	0.034	0.096
400	0.191	0.002	0.123	0.201
800	0.333	0.003	0.238	0.625
1600	1.277	0.006	0.931	2.524
3200	5.252	0.009	3.699	9.668

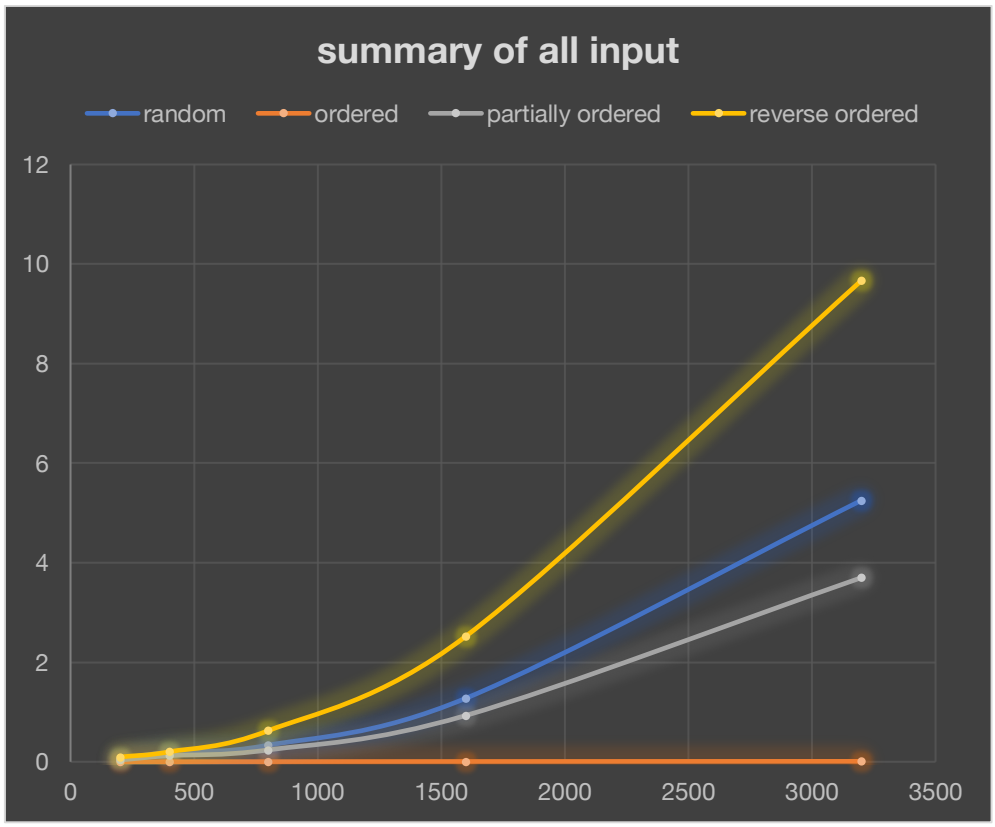

random
6
5
4
3
2
1
0
0
500
1000
1500
2000
2500
3000
3500


ordered
0.01
0.009
0.008
0.007
0.006
0.005
0.004
0.003
0.002
0.001
0
0
500
1000
1500
2000
2500
3000
3500

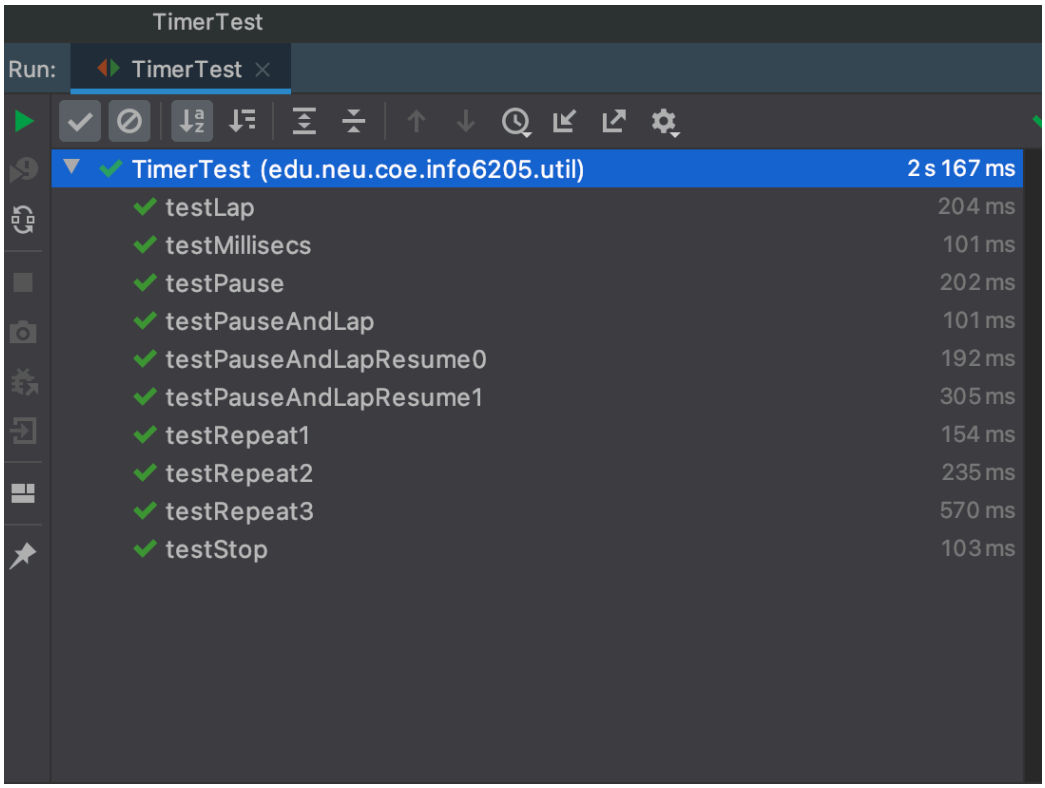


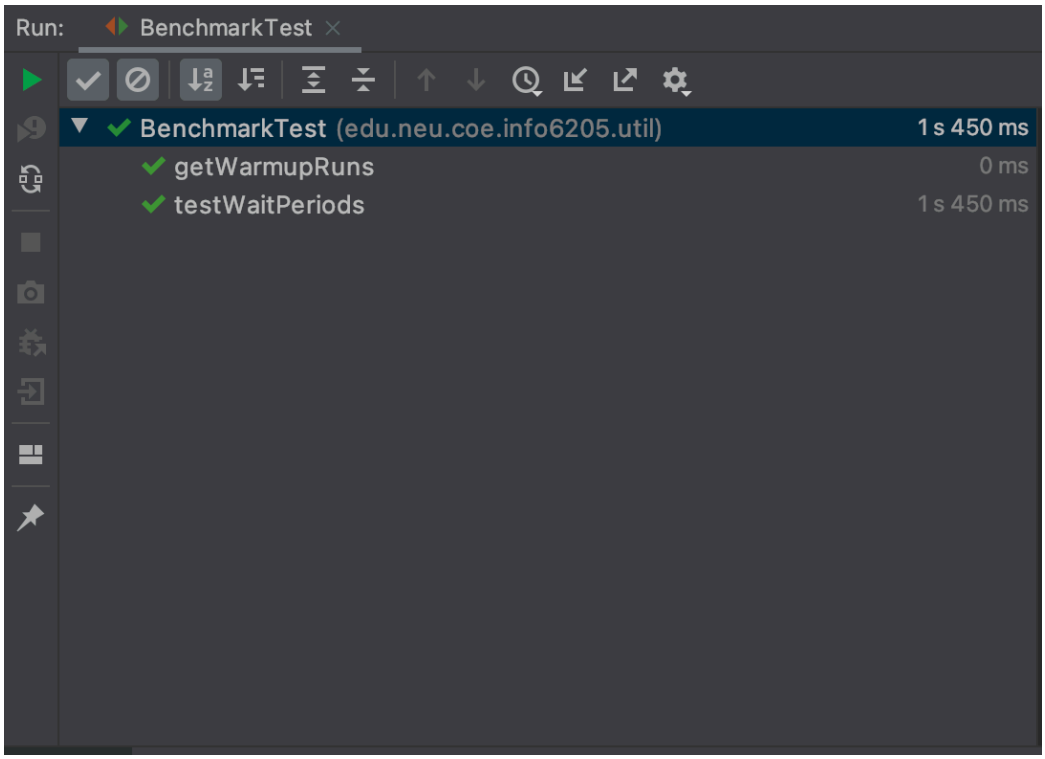


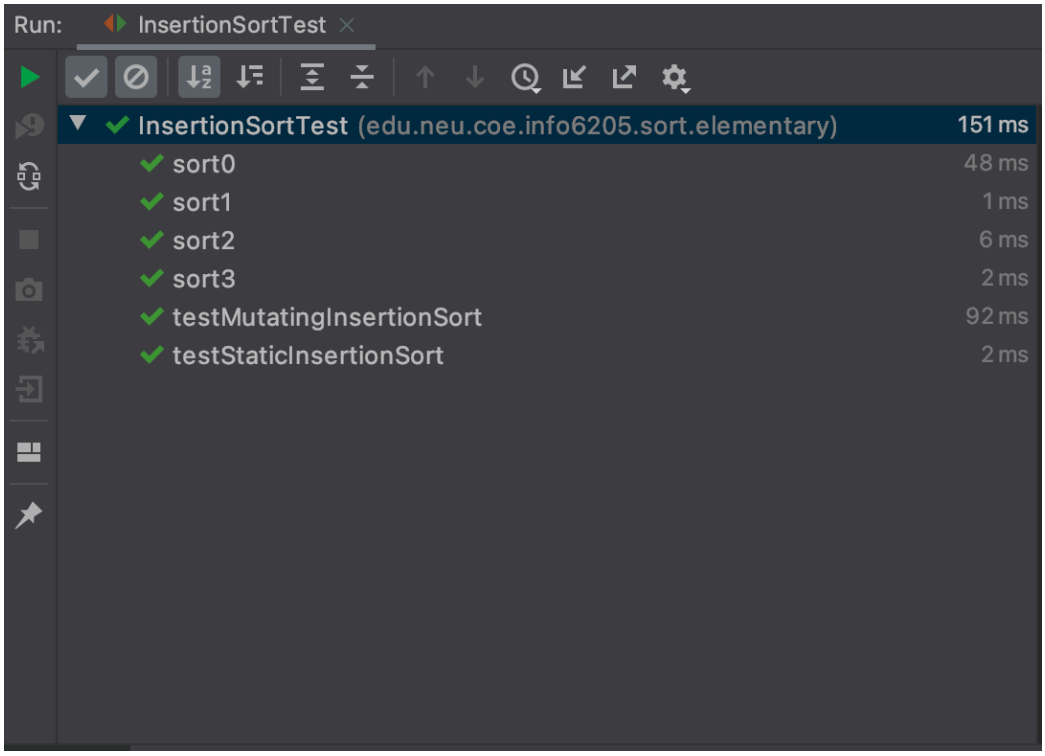
x-axis demonstrates n (size of array), y-axis demonstrates mean run time of the sort with each type of input. We can merge them into one graph to see the differences.



- **Unit tests result:**




Run:
BenchmarkTest
BenchmarkTest (edu.neu.coe.info6205.util) 1 s 450 ms
getWarmupRuns 0 ms
testWaitPeriods 1 s 450 ms


Run:
InsertionSortTest
InsertionSortTest (edu.neu.coe.info6205.sort.elementary) 151 ms
sort0 48 ms
sort1 1 ms
sort2 6 ms
sort3 2 ms
testMutatingInsertionSort 92 ms
testStaticInsertionSort 2 ms