Charmi Dalal (001582441)

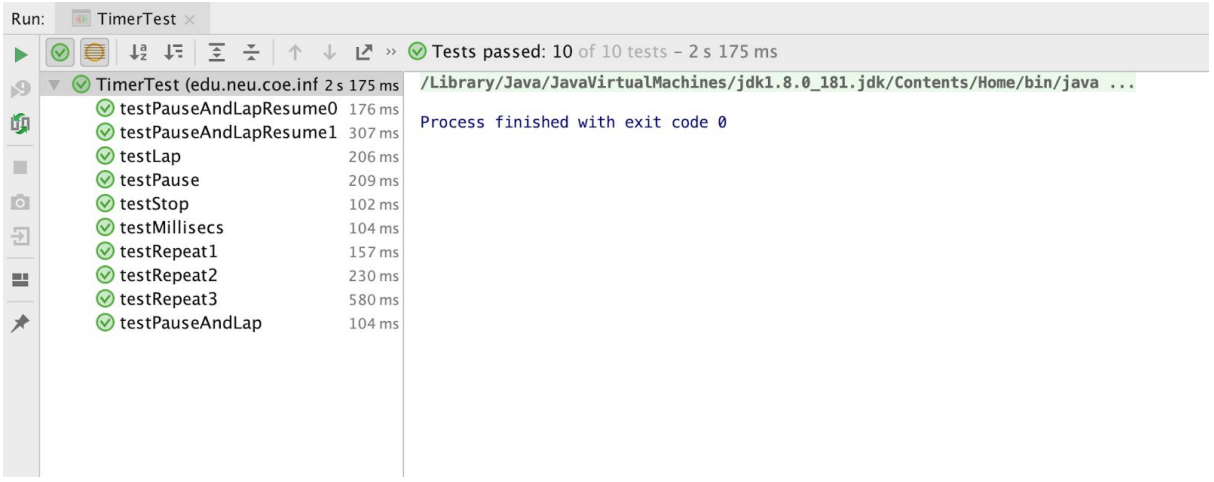
INFO 6205 Program Structure & Algorithms
Spring 2021
Assignment 2

Task1:

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.

Output:

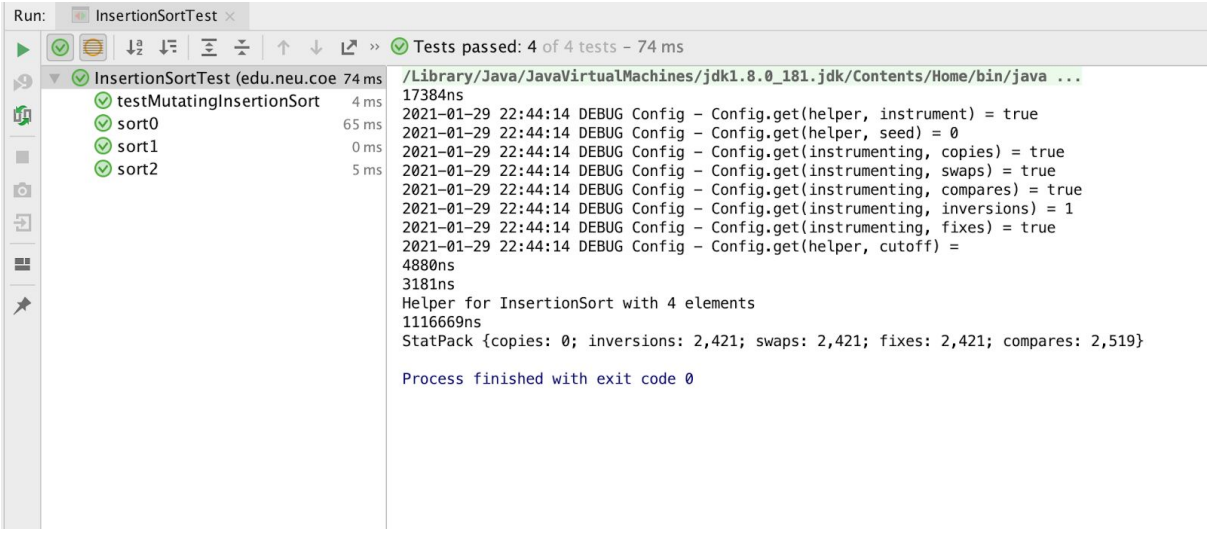
1.1 Benchmark Test Results



1.2 Timer Test Results

Task2:

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.



1.3 Insertion Sort Test Results

Task3:

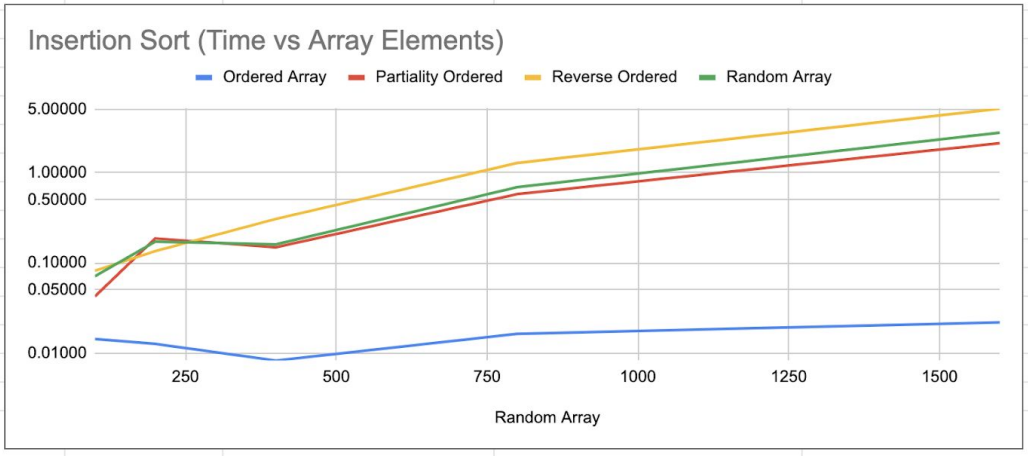
Measure the running times of this sort, using four different initial array ordering situations: random, ordered, partially-ordered and reverse-ordered. I suggest that your arrays to be sorted are of type *Integer*. Use the doubling method for choosing *n* and test for at least five values of *n.* Draw any conclusions from your observations regarding the order of growth.

```
Run:  InsertionSort
Random Sort
For 100 array element insertionsort takes 0.07329489ms
For 200 array element insertionsort takes 0.23886416000000002ms
For 400 array element insertionsort takes 0.14635824ms
For 800 array element insertionsort takes 0.553727ms
For 1600 array element insertionsort takes 2.26991167ms
****************************************
Ordered Sort
For 100 array element insertionsort takes 0.02287953ms
For 200 array element insertionsort takes 0.014534ms
For 400 array element insertionsort takes 0.0074246699999999995ms
For 800 array element insertionsort takes 0.01112649ms
For 1600 array element insertionsort takes 0.020361749999999998ms
****************************************
Partially Ordered Sort
For 100 array element insertionsort takes 0.0434228ms
For 200 array element insertionsort takes 0.13767158ms
For 400 array element insertionsort takes 0.12540572ms
For 800 array element insertionsort takes 0.45723079ms
For 1600 array element insertionsort takes 1.7072877999999996ms
****************************************
Reverse Ordered Sort
For 100 array element insertionsort takes 0.08124005ms
For 200 array element insertionsort takes 0.14905725999999997ms
For 400 array element insertionsort takes 0.28484448999999995ms
For 800 array element insertionsort takes 1.1605449ms
For 1600 array element insertionsort takes 4.037837309999999ms

Process finished with exit code 0
```

1.4 Insertion Sort RunResults

	Time(ms)			
No. elements (n)	**Random Array**	**Ordered Array**	**Partiality Ordered**	**Reverse Ordered**
100	0.07088	0.01437	0.04242	0.08185
200	0.17119	0.01266	0.18614	0.13473
400	0.16023	0.00827	0.14883	0.30607
800	0.68905	0.01634	0.57709	1.27565
1600	2.75972	0.02183	2.11491	5.09236



1.5 Insertion Sort Observation
(Note: Here y axis is showing logarithmic value of time)

Conclusion:

After doing sorting on 5 different sized arrays and observing time taken for each sort I conclude the following observation.

1. Reverse Ordered array takes maximum time to sort the array among all four types of arrays.
2. Ordered array takes minimum time to sort the array among all four types of arrays.
3. Random array and Partial Ordered array takes almost the same amount of time to sort the array.
4. As you double the size of the array, Time taken to sort the arrays becomes quadruple.