Ripan Halder
NU ID: 001442993
Assignment 2

1. Code ->

https://github.com/RipanHalder/INFO6205

Asignment2.java File -> /src/main/java/edu/neu/coe/info6205/Assignment2.java

Results -> results/insertion_sort/results.csv

Insertion Sort Code
/src/main/java/edu/neu/coe/info6205/sort/simple/InsertionSort.java

Insertion Sort Basic Code
/src/main/java/edu/neu/coe/info6205/sort/simple/InsertionSortBasic.java

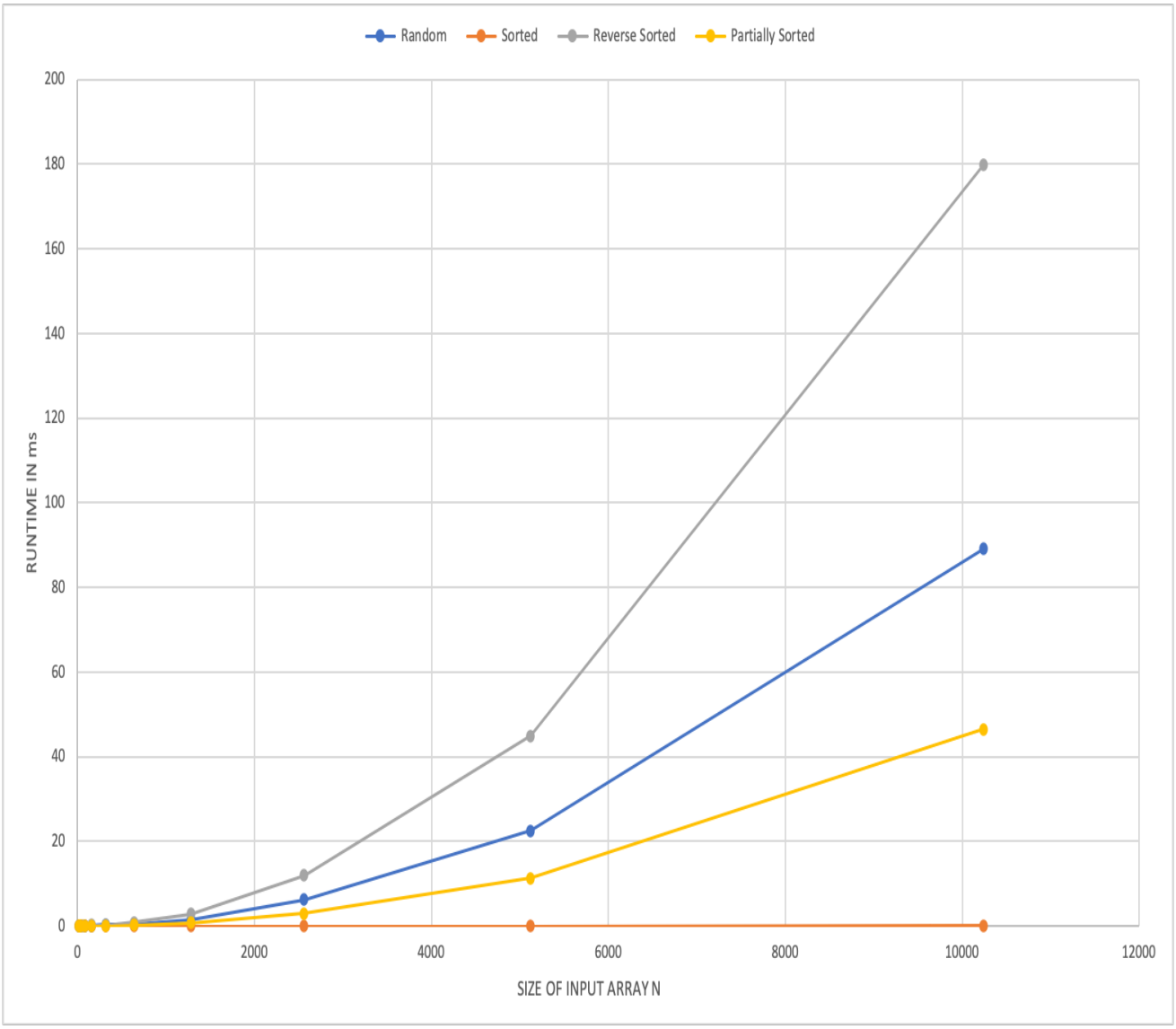
Timer Code
/src/main/java/edu/neu/coe/info6205/util/Timer.java

2. Screenshots and Evidences:

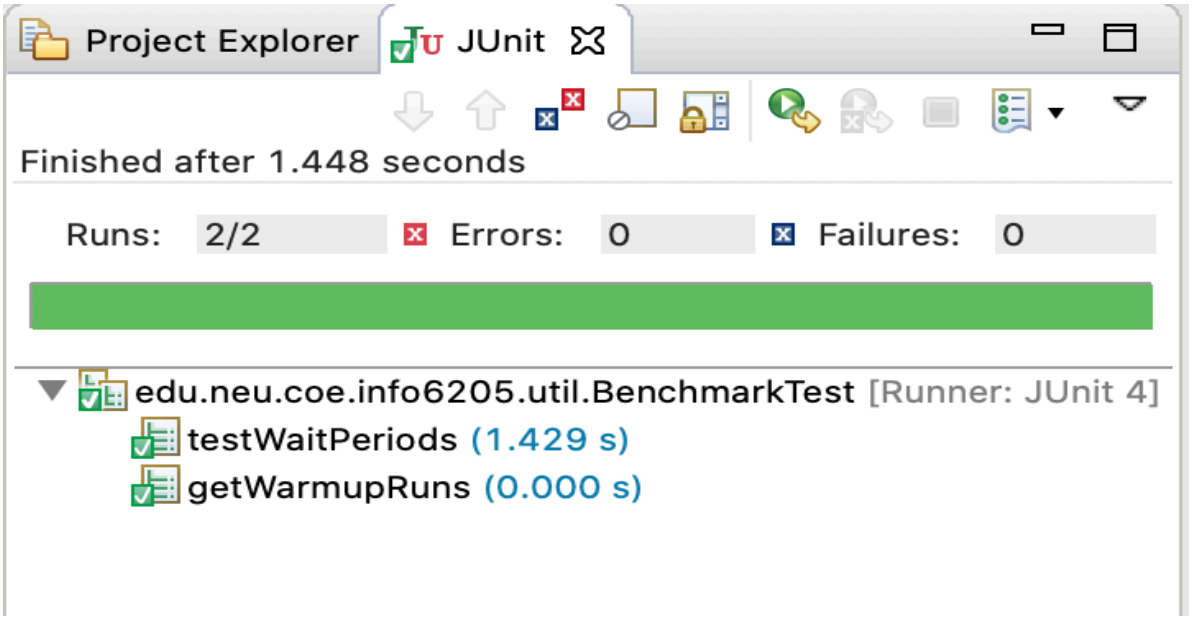
The following table shows the mean time taken by N size of input arrays when they are random, sorted, reverse or partially sorted.

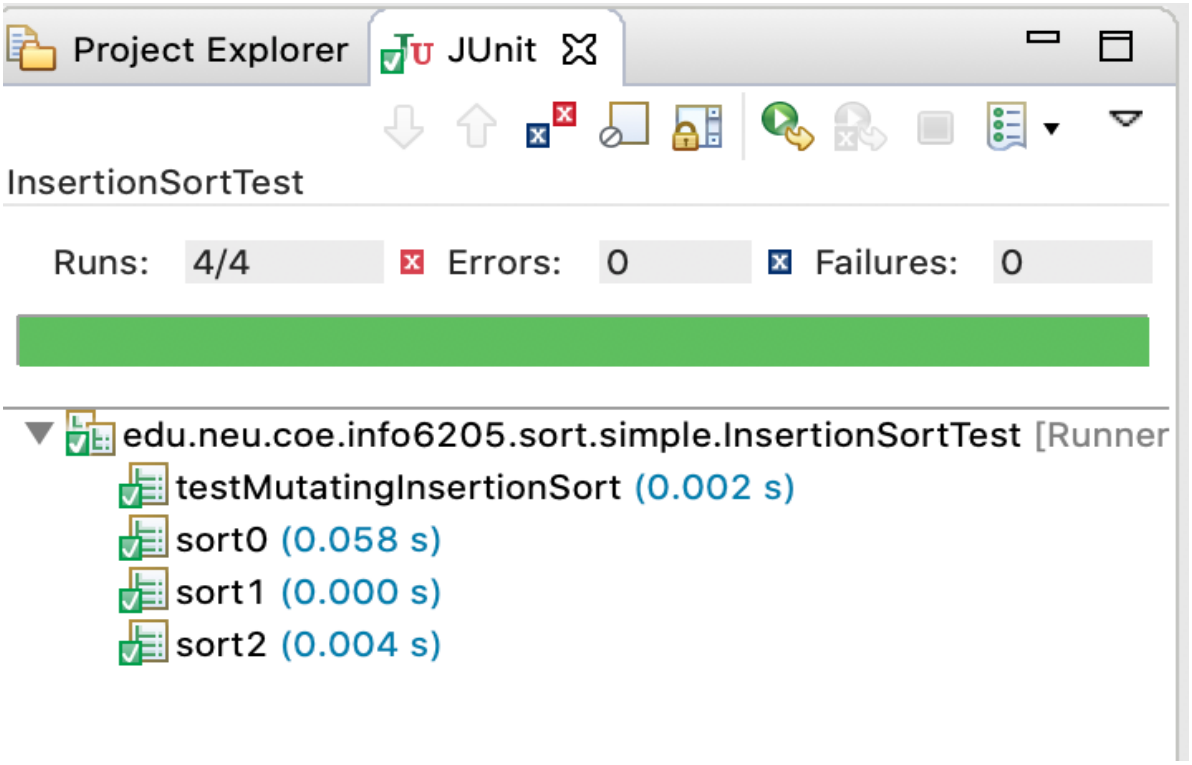
n	Random	Sorted	Reverse Sorted	Partially Sorted
10	0.015448857	0.001973929	0.0039735	0.003087857
20	0.005463571	0.002356571	0.017868143	0.006771643
40	0.017085429	0.001794643	0.075909786	0.004672214
80	0.032771143	0.0039495	0.081150929	0.019795786
160	0.076768714	0.003671571	0.185389071	0.033305643
320	0.316140786	0.0039525	0.286326143	0.047583857
640	0.381167071	0.004704143	0.858183286	0.311149786
1280	1.504971643	0.007723571	2.863168643	0.779422857
2560	6.228537	0.013355071	11.915798	2.970112
5120	22.57178071	0.025591143	44.99579636	11.28264129
10240	89.13674029	0.055204786	179.827852	46.44845193

The chart which we can understand from the above result of my code is:



Unit Tests Results of Benchmark Test and Insertion Sort Test





3. Conclusion:

- Reverse sorted grow exponentially and takes most time to execute. It is of the order of N^2.
- Random sorted array also grows exponentially but not as much as that of reverse sorted.
- Partially sorted also grows exponentially to the order of N^2, but depends its curve depends upon the number of initial elements sorted. The more percentage of elements sorted initially; the less duration taken to sort.
- Sorting a sorting array takes negligible amount of time, even if size of input array increases rapidly.