Varun Venkatesh Gowda(002126161)

Program Structures & Algorithms
Fall 2021
Assignment No. 2

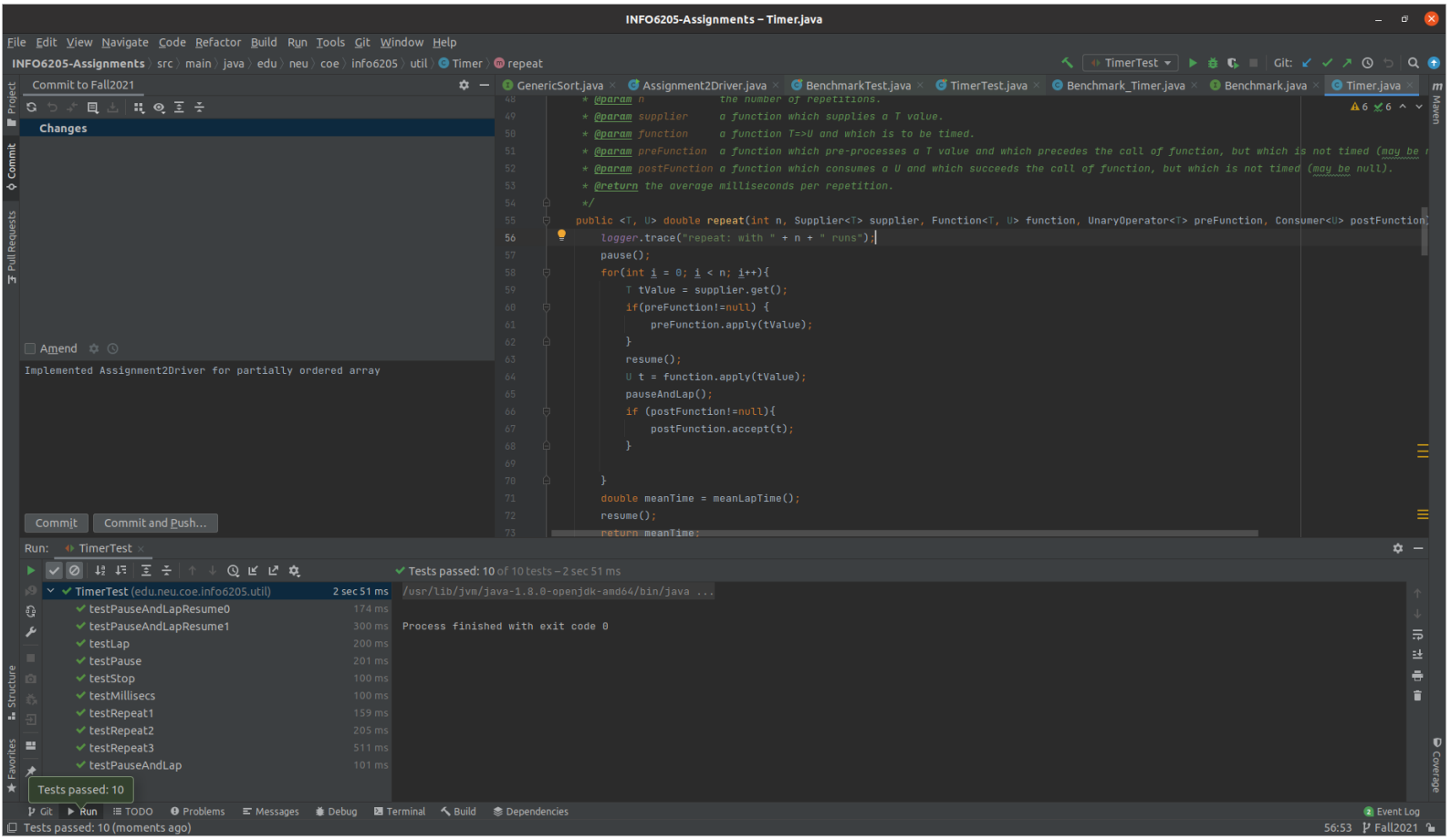
Tasks:

1. Implemented getClock( ) function in Timer.java to return System.nanoTime
2. Implemented toMillisecs( ) function in Timer.java to convert nanoseconds to milliseconds
3. Implemented repeat( ) to iterate over ‘n’ runs and to time the execution of ‘function’ and return the meanLapTime( ) returned value.
4. Ran TimerTest.java and BenchmarkTest.java unit tests to test implementation (Passed)
5. Implemented sort( ) function in InsertionSort.java using helper.swapStableConditional method
6. Ran InsertionSortTest.java unit tests to test the implementation (Passed)
7. Implemented a main program named Assignment2Driver.java in edu.neu.coe.info6205.assignment2 to be able to run the benchmarks with 4 required scenarios: random array, ordered array, reverse array, partially ordered array
8. Conducted experiments for each scenario after a warmup of 5-10 runs and recorded values
9. Tabulated readings and plotted relevant graphs

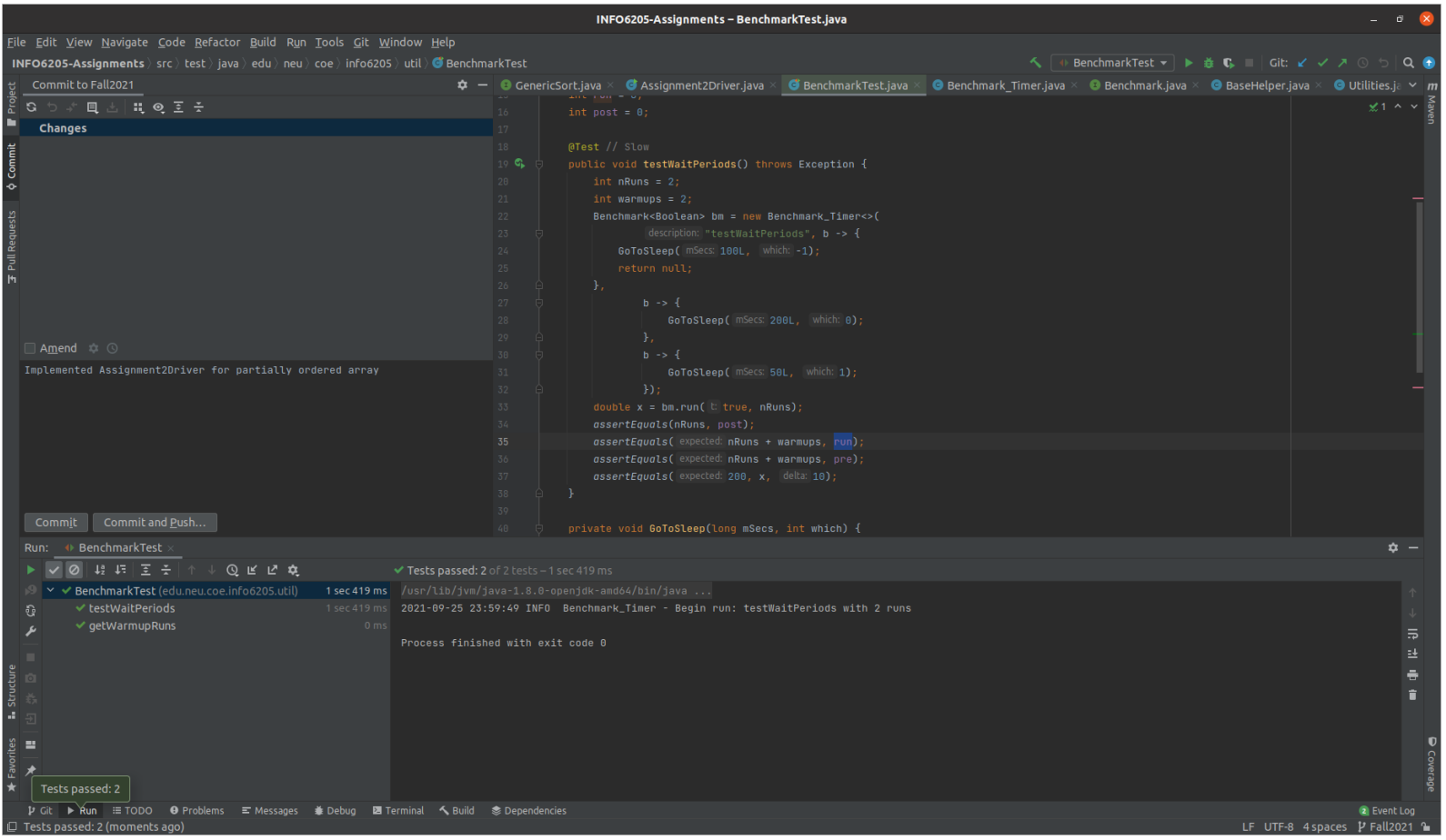
Screenshots:

Unit tests:

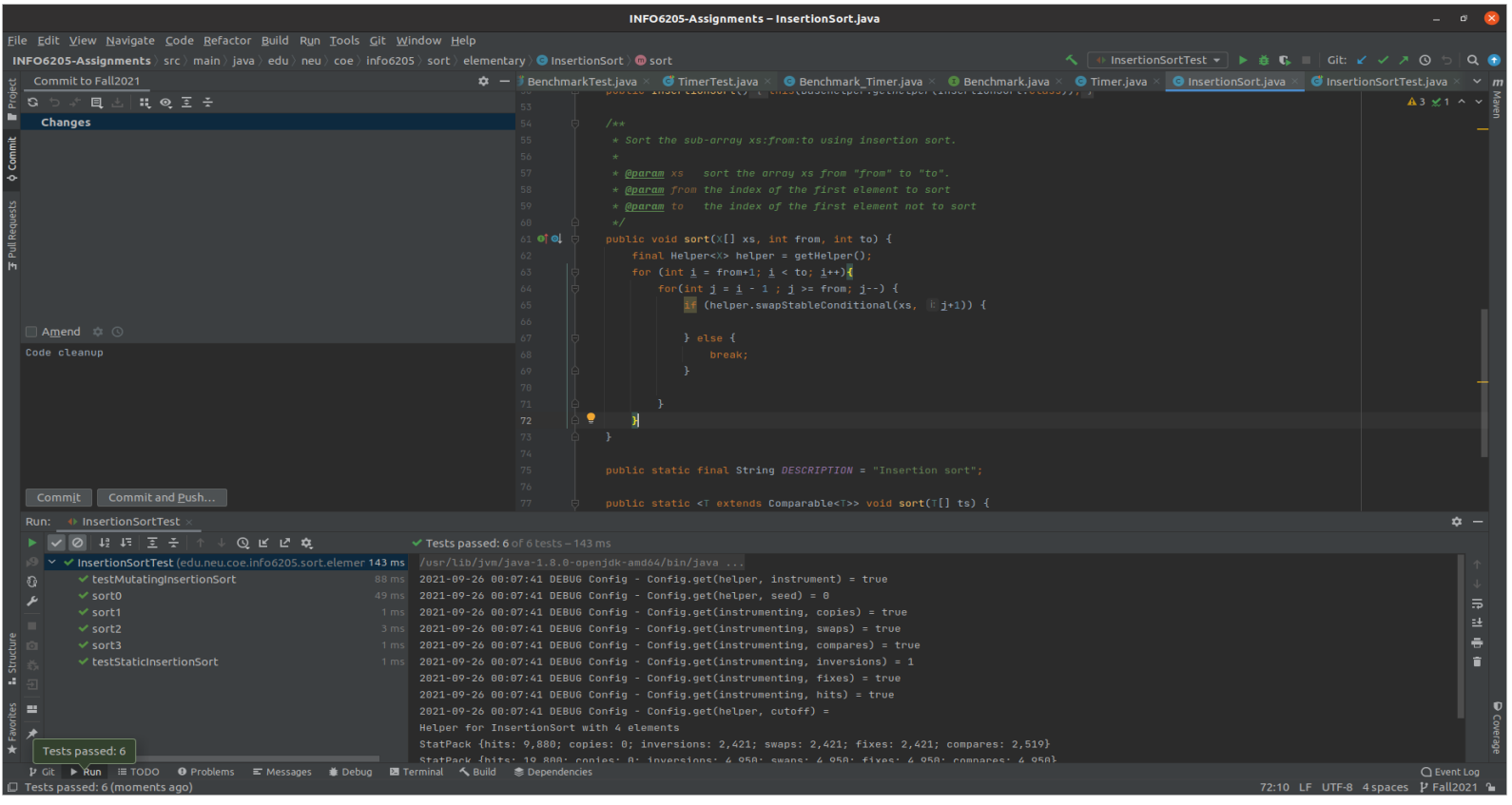
1. TimerTest



2. BenchmarkTest

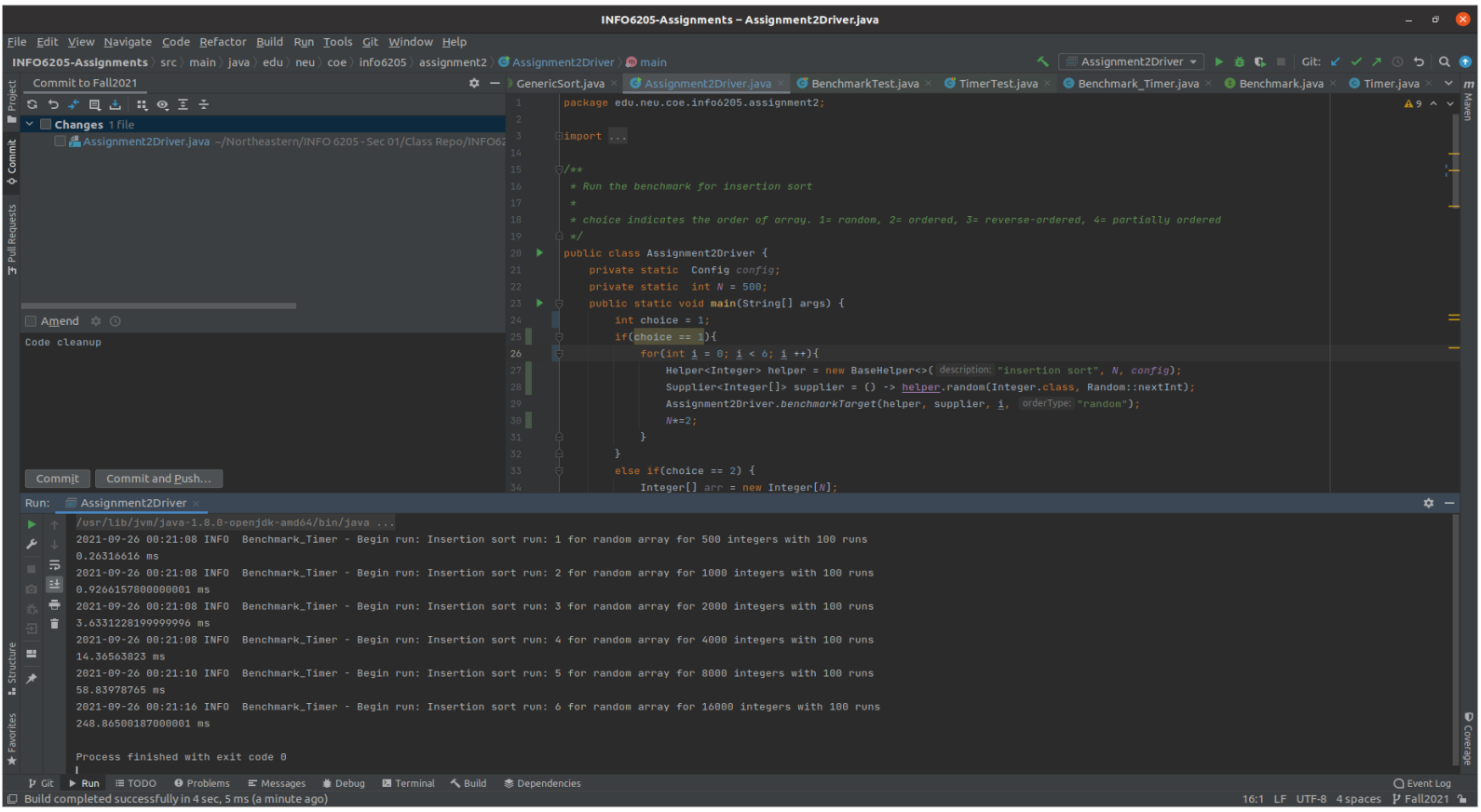


3. InsertionSortTest

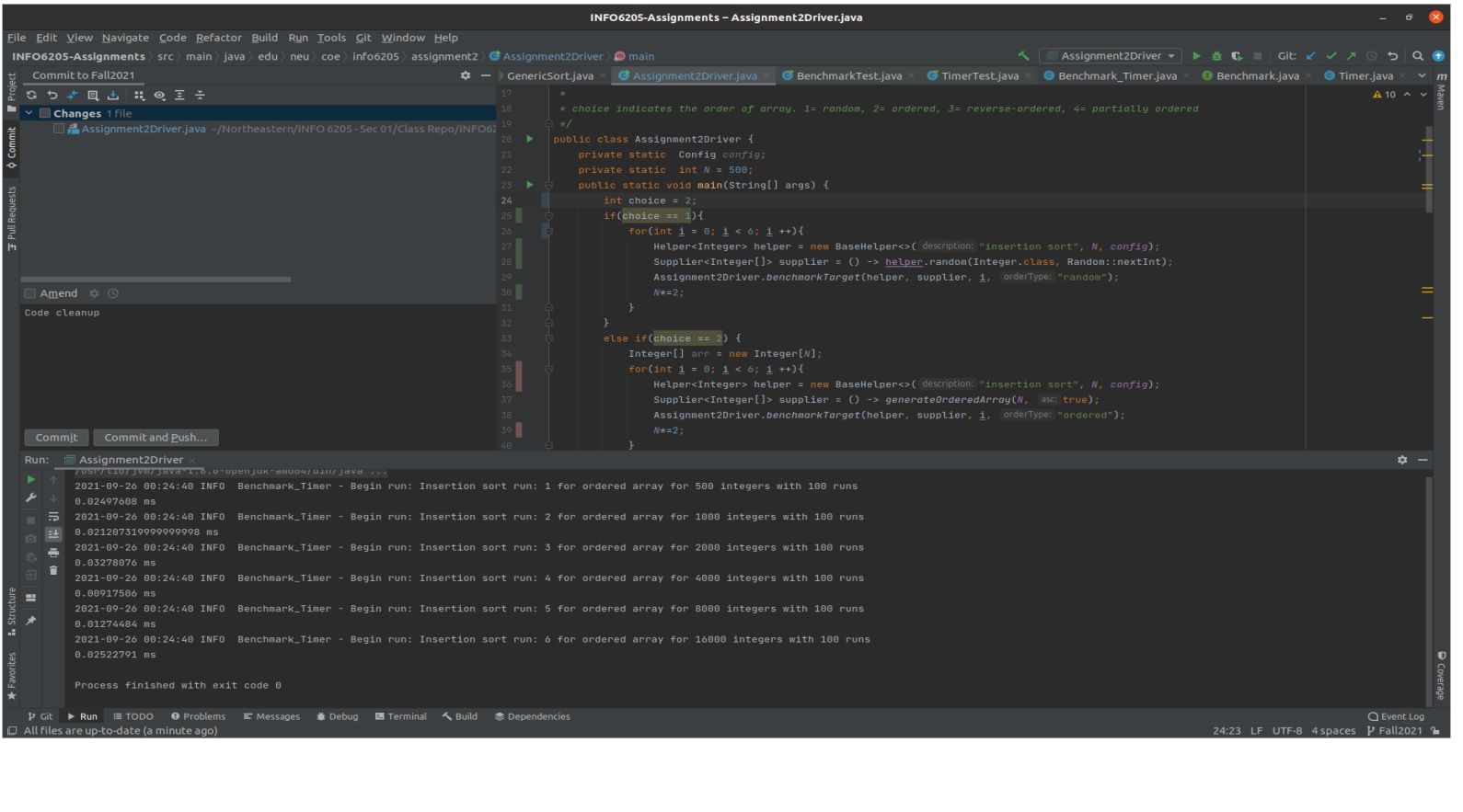


Outputs

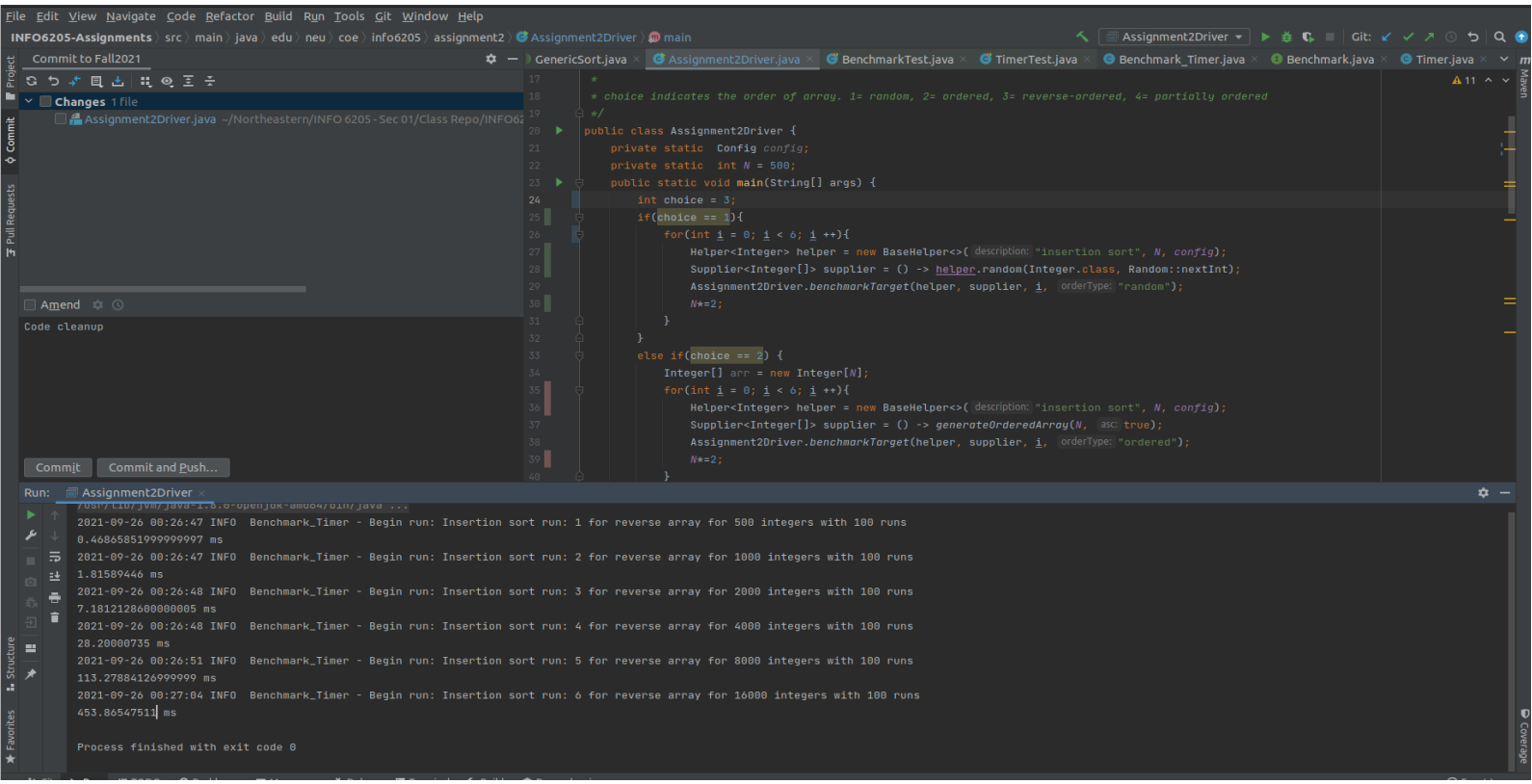
1. Random Array



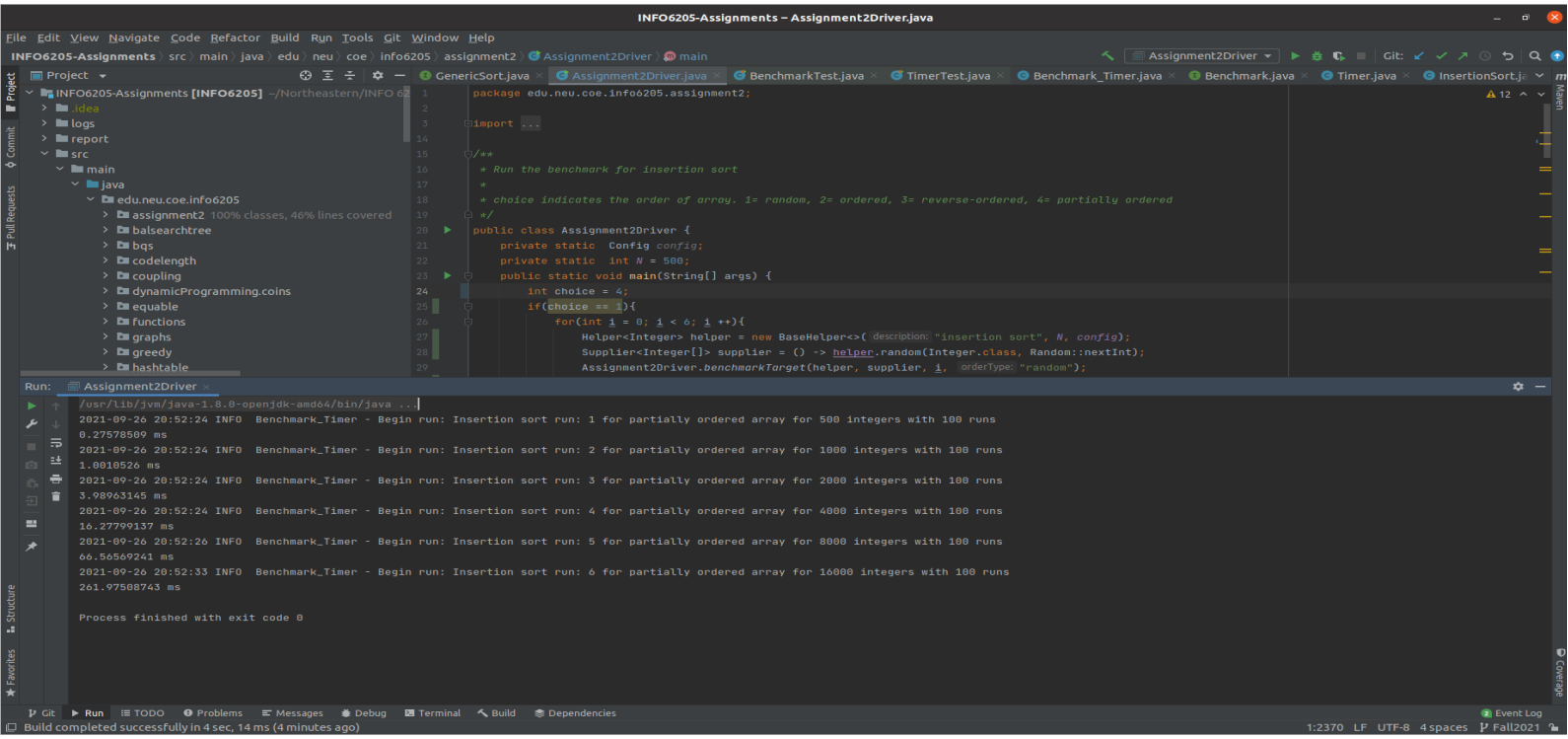
2. Ordered Array



3. Reverse Array



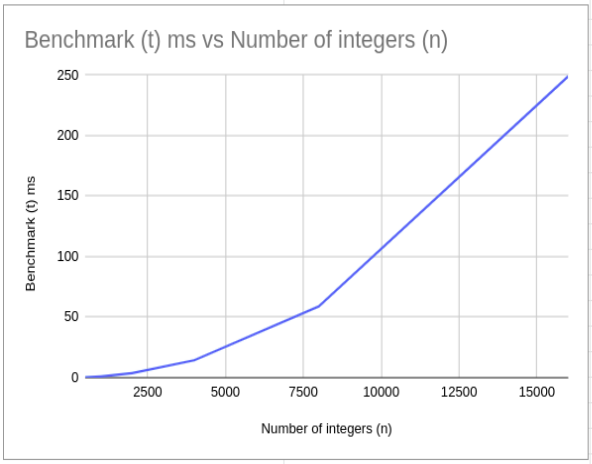
4. Partially Ordered Array

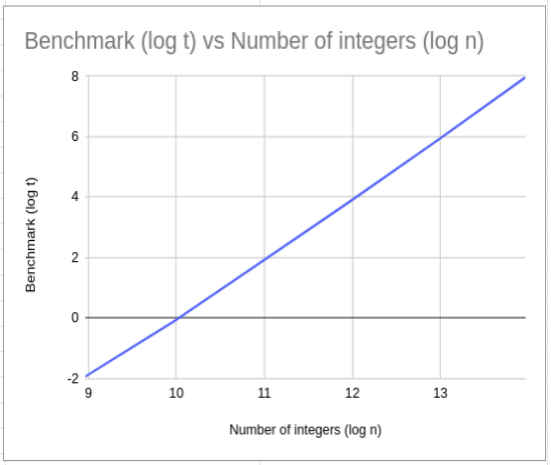


Tabulation and graphs: (t vs n[left] & log t vs log n[right])

1. RandomArray

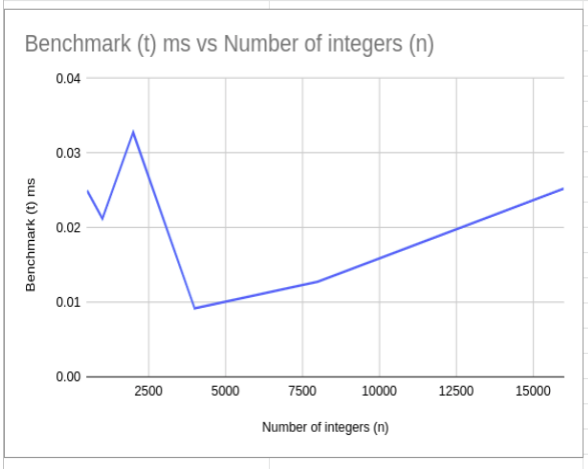
Number of integers (n)	Benchmark (t) ms	Number of integers (log n)	Benchmark (log t)
500	0.26316616	8.965784285	-1.925954107
1000	0.92661578	9.965784285	-0.1099568436
2000	3.63312282	10.96578428	1.861210138
4000	14.36563823	11.96578428	3.844550185
8000	58.83978765	12.96578428	5.878720135
16000	248.8650019	13.96578428	7.959219547

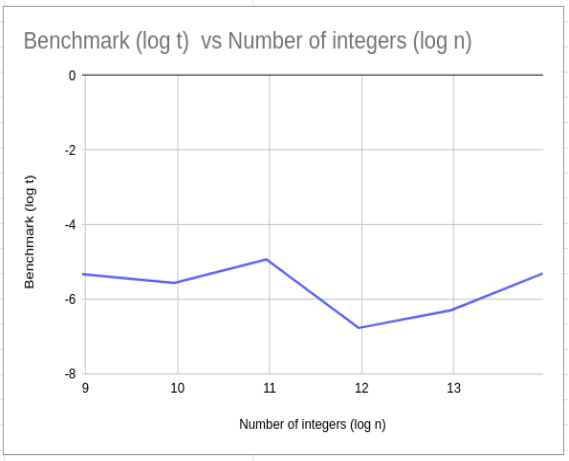




2. Ordered Array

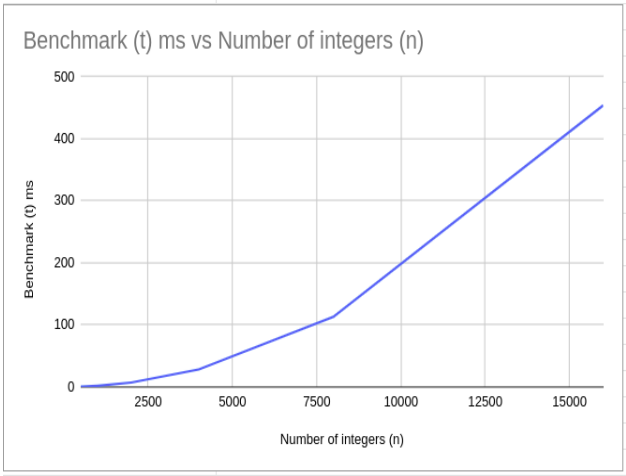
A	B	C	D	E
Number of integers (n)	Benchmark (t) ms		Number of integers (log n)	Benchmark (log t)
500	0.02497608		8.965784285	-5.323309126
1000	0.02120732		9.965784285	-5.559293873
2000	0.03278076		10.96578428	-4.931006887
4000	0.00917506		11.96578428	-6.768066692
8000	0.01274484		12.96578428	-6.293942928
16000	0.02522791		13.96578428	-5.308835498

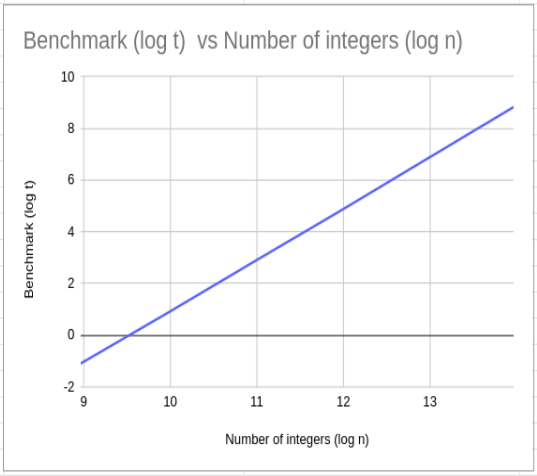




3. Reverse Array

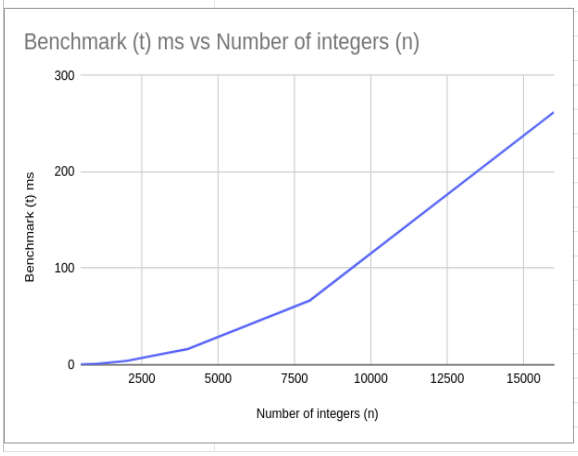
A	B	C	D	E
Number of integers (n)	Benchmark (t) ms		Number of integers (log n)	Benchmark (log t)
500	0.46865852		8.965784285	-1.093390984
1000	1.81589446		9.965784285	0.8606803555
2000	7.18121286		10.96578428	2.844227526
4000	28.20000735		11.96578428	4.817623634
8000	113.2788413		12.96578428	6.823734603
16000	453.8654751		13.96578428	8.826120939

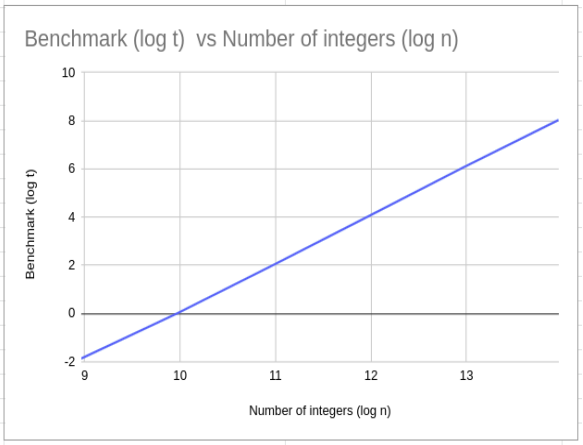




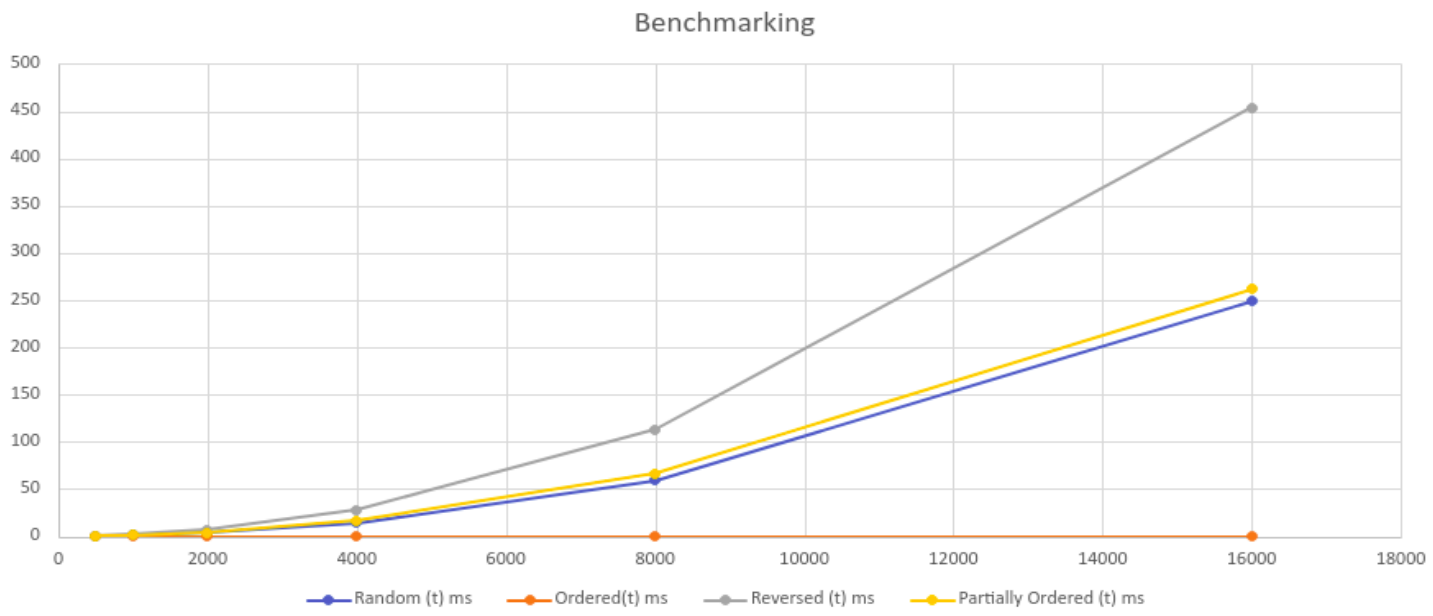
4. Partially Ordered Array

A	B	C	D	E
Number of integers (n)	Benchmark (t) ms		Number of integers (log n)	Benchmark (log t)
500	0.27578509		8.965784285	-1.858383634
1000	1.0010526		9.965784285	0.001517782131
2000	3.98963145		10.96578428	1.996255481
4000	16.27799137		11.96578428	4.024850783
8000	66.56569241		12.96578428	6.056706907
16000	261.9750874		13.96578428	8.033285815





Consolidated Graph:



Conclusions:

1. An ordered array has the fastest benchmark time by a long margin.
2. The general trend of the log graphs indicates the order of growth for Partially Ordered array and Random Array is closer to O(n log n).
3. The trend for Reverse ordered array indicates order of growth is closer to O (n^2)
4. The trend for Ordered Array indicates order of growth is closer to O(1)
5. The benchmarks in ascending order of time are Ordered, Partially Ordered, Random and Reverse order