Varun Venkatesh Gowda(002126161)

Program Structures & Algorithms
Fall 2021
Assignment No. 2

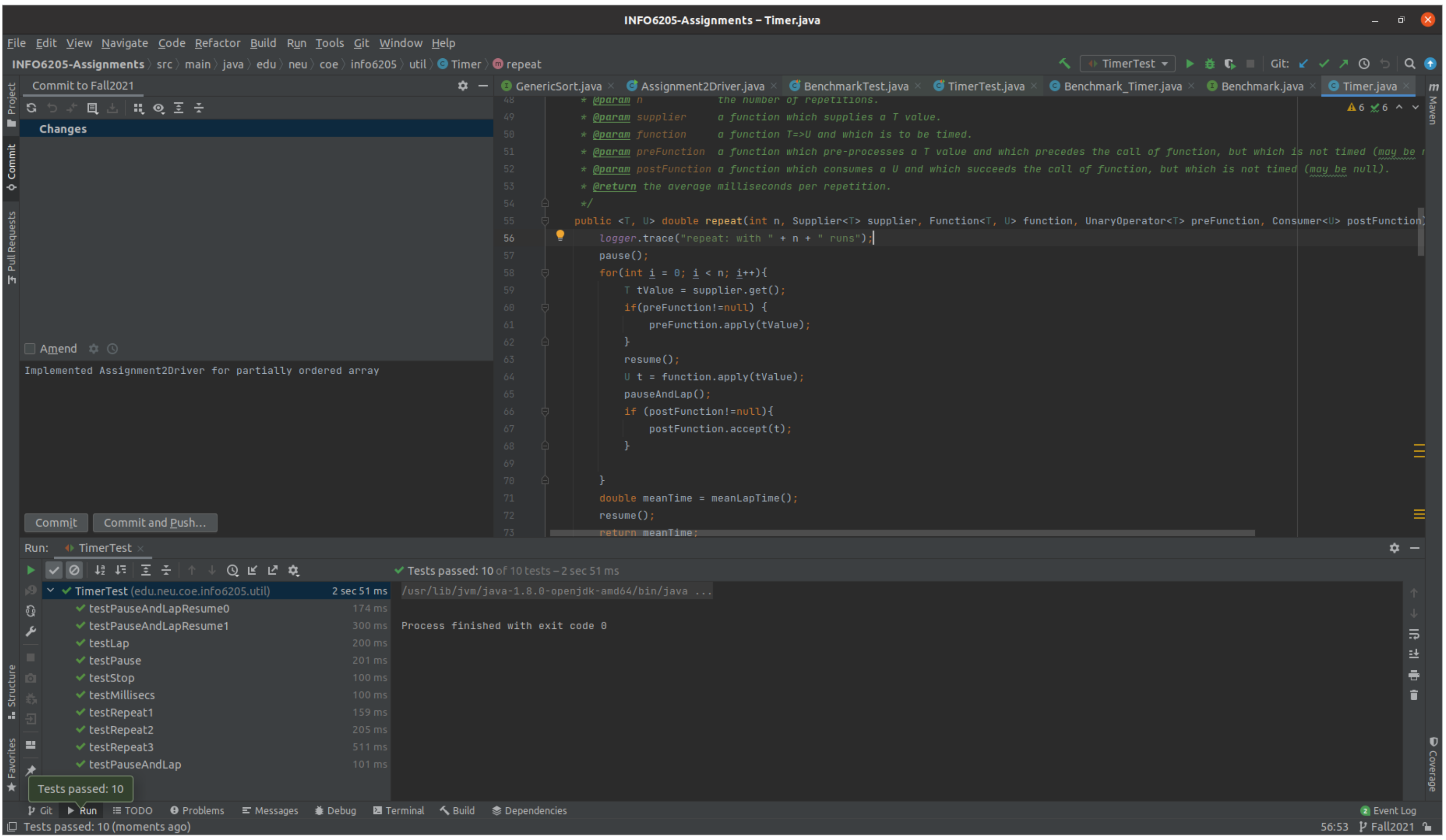
Tasks:

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**



INFO6205-Assignments – Timer.java

```
 * @param n              the number of repetitions.
 * @param supplier     a function which supplies a T value.
 * @param function     a function T=>U and which is to be timed.
 * @param preFunction  a function which pre-processes a T value and which precedes the call of function, but which is not timed (may be r
 * @param postFunction a function which consumes a U and which succeeds the call of function, but which is not timed (may be null).
 * @return the average milliseconds per repetition.
 */
public <T, U> double repeat(int n, Supplier<T> supplier, Function<T, U> function, UnaryOperator<T> preFunction, Consumer<U> postFunction
    logger.trace("repeat: with " + n + " runs");
    pause();
    for(int i = 0; i < n; i++){
        T tValue = supplier.get();
        if(preFunction!=null) {
            preFunction.apply(tValue);
        }
        resume();
        U t = function.apply(tValue);
        pauseAndLap();
        if (postFunction!=null){
            postFunction.accept(t);
        }

    }
    double meanTime = meanLapTime();
    resume();
    return meanTime;
```

Commit message:

```
Implemented Assignment2Driver for partially ordered array
```

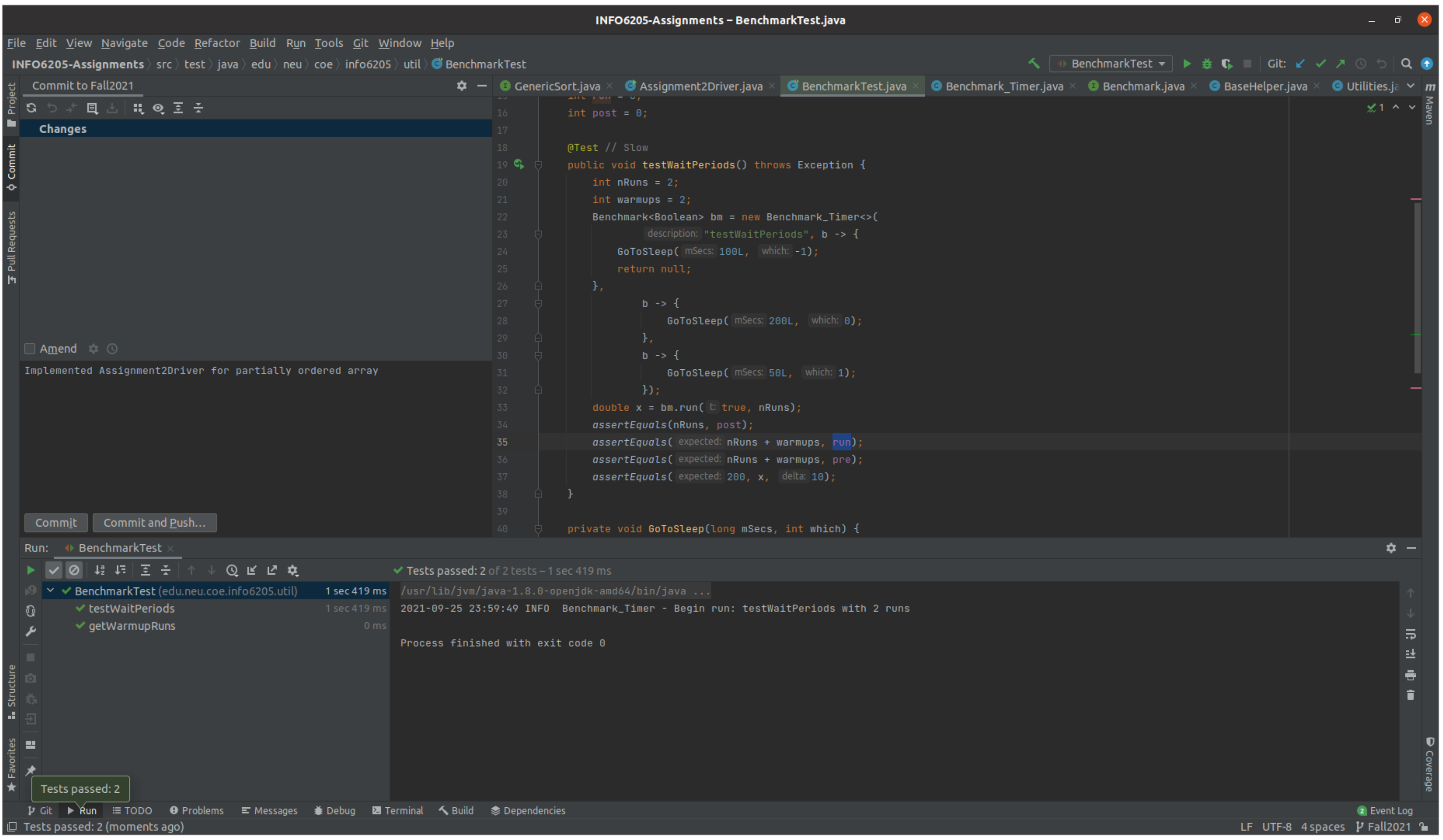
Run: TimerTest

```
Tests passed: 10 of 10 tests – 2 sec 51 ms

TimerTest (edu.neu.coe.info6205.util)  2 sec 51 ms
    testPauseAndLapResume0               174 ms
    testPauseAndLapResume1               300 ms
    testLap                              200 ms
    testPause                            201 ms
    testStop                             100 ms
    testMillisecs                        100 ms
    testRepeat1                          159 ms
    testRepeat2                          205 ms
    testRepeat3                          511 ms
    testPauseAndLap                      101 ms
```

```
/usr/lib/jvm/java-1.8.0-openjdk-amd64/bin/java ...

Process finished with exit code 0
```

Tests passed: 10 (moments ago)

2. BenchmarkTest



INFO6205-Assignments – BenchmarkTest.java

Implemented Assignment2Driver for partially ordered array

```java
    int post = 0;

    @Test // Slow
    public void testWaitPeriods() throws Exception {
        int nRuns = 2;
        int warmups = 2;
        Benchmark<Boolean> bm = new Benchmark_Timer<>(
                description: "testWaitPeriods", b -> {
            GoToSleep( mSecs: 100L,  which: -1);
            return null;
        },
                b -> {
                    GoToSleep( mSecs: 200L,  which: 0);
                },
                b -> {
                    GoToSleep( mSecs: 50L,  which: 1);
                });
        double x = bm.run( t: true, nRuns);
        assertEquals(nRuns, post);
        assertEquals( expected: nRuns + warmups, run);
        assertEquals( expected: nRuns + warmups, pre);
        assertEquals( expected: 200, x,  delta: 10);
    }

    private void GoToSleep(long mSecs, int which) {
```

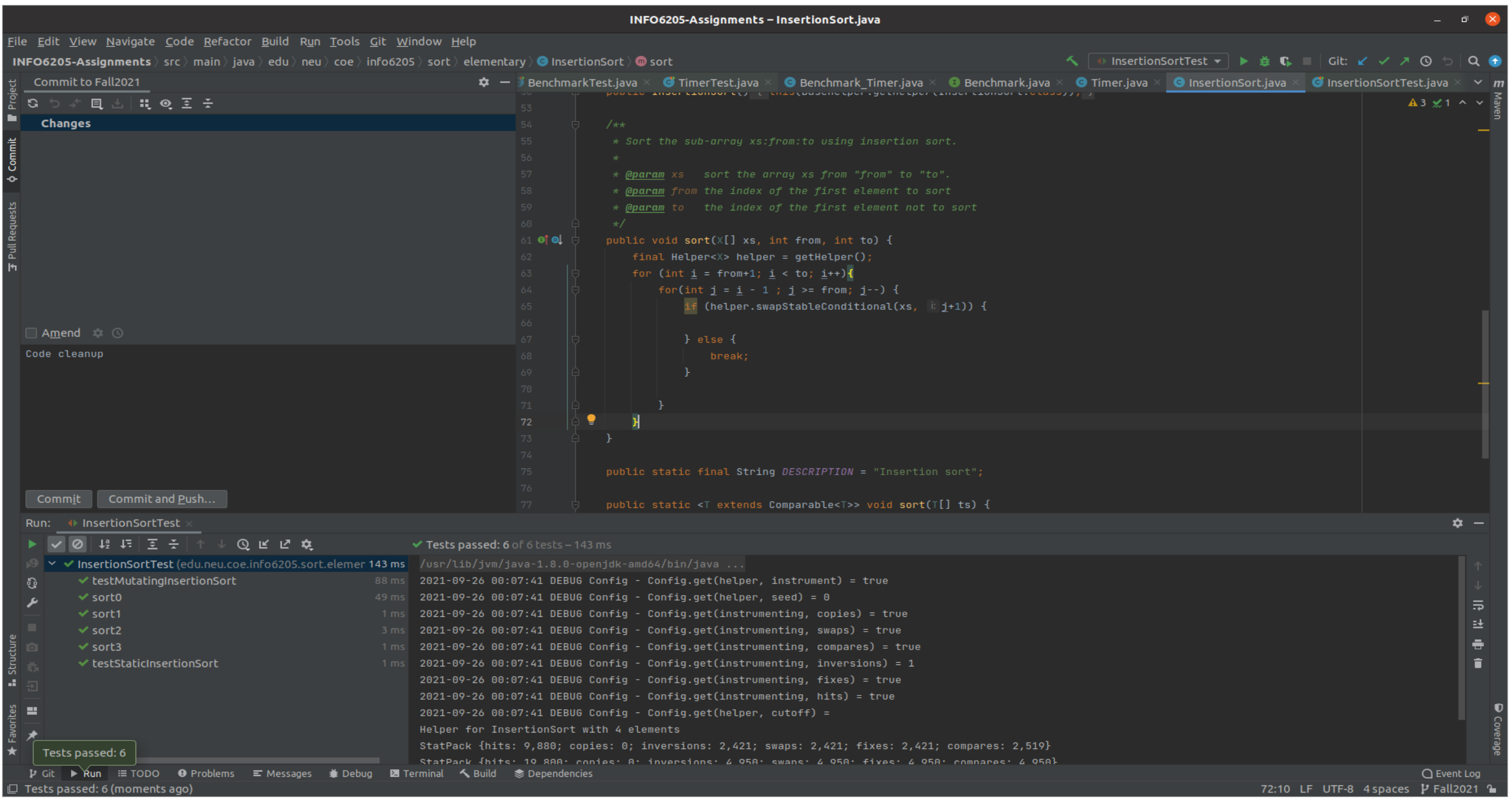
Tests passed: 2 of 2 tests – 1 sec 419 ms

```
BenchmarkTest (edu.neu.coe.info6205.util)   1 sec 419 ms
    testWaitPeriods                         1 sec 419 ms
    getWarmupRuns                           0 ms
```

```
/usr/lib/jvm/java-1.8.0-openjdk-amd64/bin/java ...
2021-09-25 23:59:49 INFO  Benchmark_Timer - Begin run: testWaitPeriods with 2 runs

Process finished with exit code 0
```

Tests passed: 2 (moments ago)

3. InsertionSortTest



INFO6205-Assignments – InsertionSort.java

Code cleanup

```java
    /**
     * Sort the sub-array xs:from:to using insertion sort.
     *
     * @param xs   sort the array xs from "from" to "to".
     * @param from the index of the first element to sort
     * @param to   the index of the first element not to sort
     */
    public void sort(X[] xs, int from, int to) {
        final Helper<X> helper = getHelper();
        for (int i = from+1; i < to; i++){
            for(int j = i - 1 ; j >= from; j--) {
                if (helper.swapStableConditional(xs,  i: j+1)) {

                } else {
                    break;
                }

            }
        }
    }

    public static final String DESCRIPTION = "Insertion sort";

    public static <T extends Comparable<T>> void sort(T[] ts) {
```

Tests passed: 6 of 6 tests – 143 ms

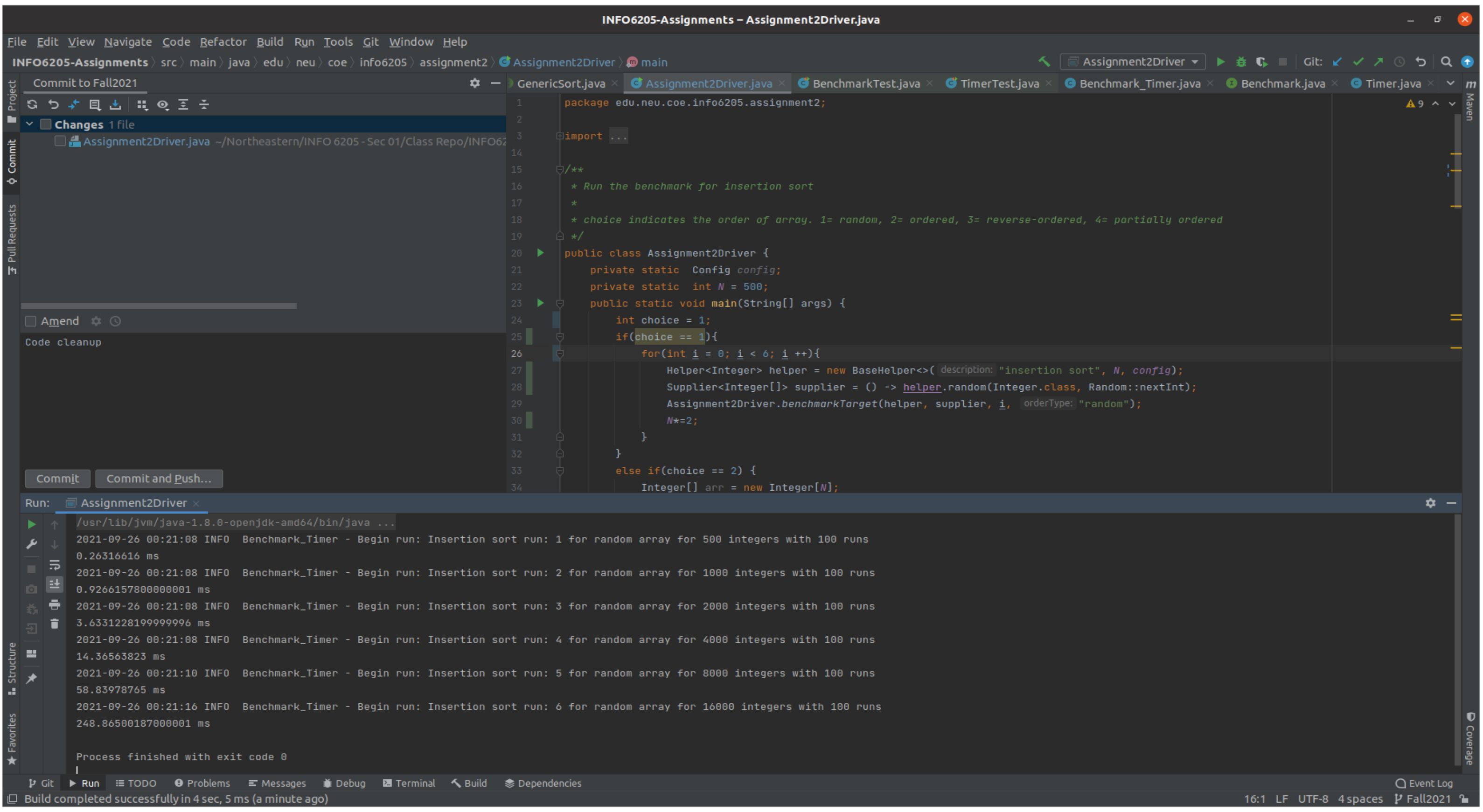
```
InsertionSortTest (edu.neu.coe.info6205.sort.elemer 143 ms
    testMutatingInsertionSort    88 ms
    sort0                        49 ms
    sort1                        1 ms
    sort2                        3 ms
    sort3                        1 ms
    testStaticInsertionSort      1 ms
```

```
/usr/lib/jvm/java-1.8.0-openjdk-amd64/bin/java ...
2021-09-26 00:07:41 DEBUG Config - Config.get(helper, instrument) = true
2021-09-26 00:07:41 DEBUG Config - Config.get(helper, seed) = 0
2021-09-26 00:07:41 DEBUG Config - Config.get(instrumenting, copies) = true
2021-09-26 00:07:41 DEBUG Config - Config.get(instrumenting, swaps) = true
2021-09-26 00:07:41 DEBUG Config - Config.get(instrumenting, compares) = true
2021-09-26 00:07:41 DEBUG Config - Config.get(instrumenting, inversions) = 1
2021-09-26 00:07:41 DEBUG Config - Config.get(instrumenting, fixes) = true
2021-09-26 00:07:41 DEBUG Config - Config.get(instrumenting, hits) = true
2021-09-26 00:07:41 DEBUG Config - Config.get(helper, cutoff) =
Helper for InsertionSort with 4 elements
StatPack {hits: 9,880; copies: 0; inversions: 2,421; swaps: 2,421; fixes: 2,421; compares: 2,519}
StatPack {hits: 19,800; copies: 0; inversions: 4,950; swaps: 4,950; fixes: 4,950; compares: 4,950}
```

Tests passed: 6 (moments ago)

Outputs

1. Random Array


INFO6205-Assignments – Assignment2Driver.java
File Edit View Navigate Code Refactor Build Run Tools Git Window Help
INFO6205-Assignments 〉 src 〉 main 〉 java 〉 edu 〉 neu 〉 coe 〉 info6205 〉 assignment2 〉 Assignment2Driver 〉 main
Commit to Fall2021
Changes 1 file
Assignment2Driver.java ~/Northeastern/INFO 6205 - Sec 01/Class Repo/INFO62
Amend
Commit
Commit and Push...
GenericSort.java
Assignment2Driver.java
BenchmarkTest.java
TimerTest.java
Benchmark_Timer.java
Benchmark.java
Timer.java
Run: Assignment2Driver
Git Run TODO Problems Messages Debug Terminal Build Dependencies
Event Log
16:1 LF UTF-8 4 spaces Fall2021

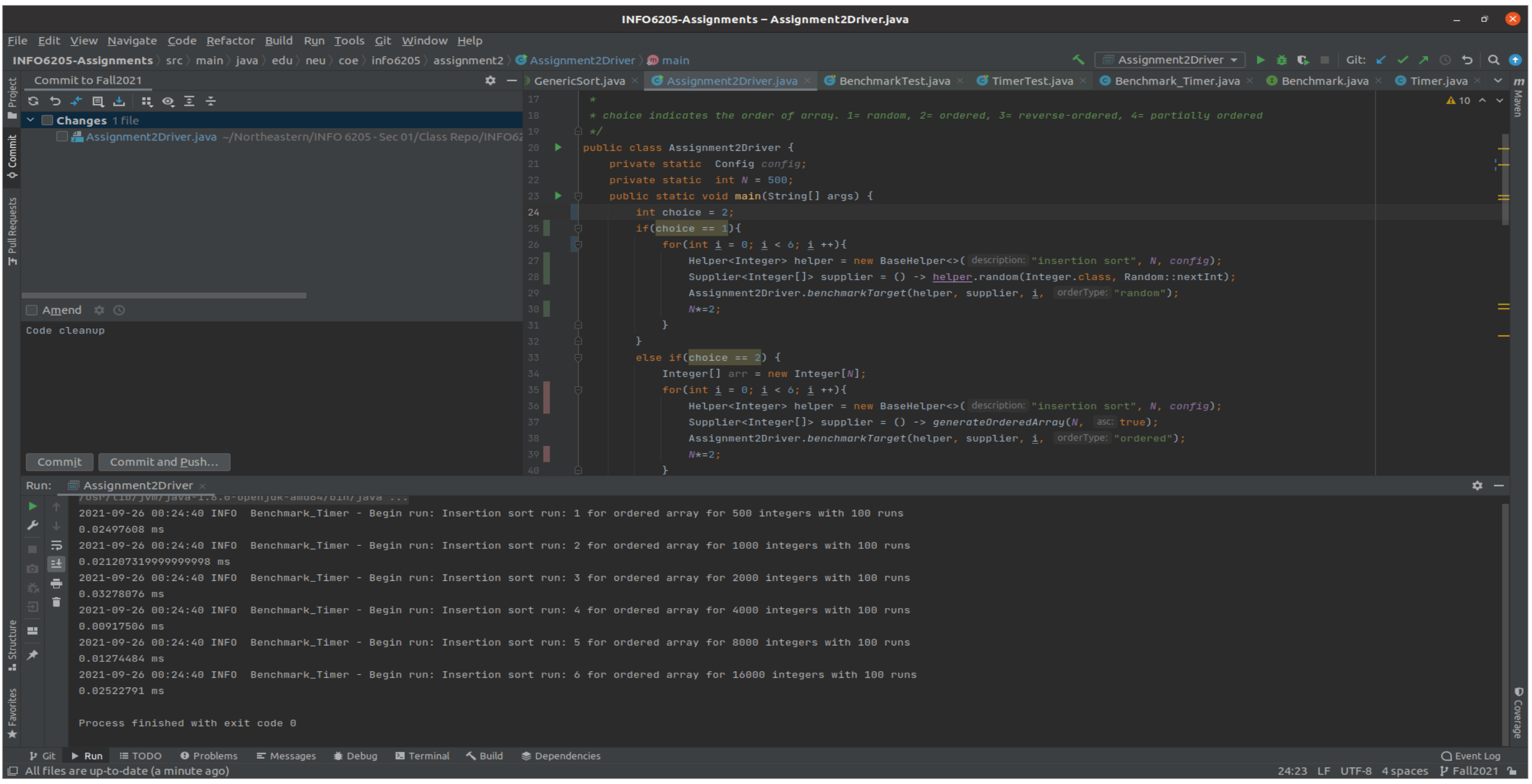
```
Code cleanup
```

```java
package edu.neu.coe.info6205.assignment2;

import ...

/**
 * Run the benchmark for insertion sort
 *
 * choice indicates the order of array. 1= random, 2= ordered, 3= reverse-ordered, 4= partially ordered
 */
public class Assignment2Driver {
    private static  Config config;
    private static  int N = 500;
    public static void main(String[] args) {
        int choice = 1;
        if(choice == 1){
            for(int i = 0; i < 6; i ++){
                Helper<Integer> helper = new BaseHelper<>("insertion sort", N, config);
                Supplier<Integer[]> supplier = () -> helper.random(Integer.class, Random::nextInt);
                Assignment2Driver.benchmarkTarget(helper, supplier, i, "random");
                N*=2;
            }
        }
        else if(choice == 2) {
            Integer[] arr = new Integer[N];
```

```
/usr/lib/jvm/java-1.8.0-openjdk-amd64/bin/java ...
2021-09-26 00:21:08 INFO  Benchmark_Timer - Begin run: Insertion sort run: 1 for random array for 500 integers with 100 runs
0.26316616 ms
2021-09-26 00:21:08 INFO  Benchmark_Timer - Begin run: Insertion sort run: 2 for random array for 1000 integers with 100 runs
0.9266157800000001 ms
2021-09-26 00:21:08 INFO  Benchmark_Timer - Begin run: Insertion sort run: 3 for random array for 2000 integers with 100 runs
3.6331228199999996 ms
2021-09-26 00:21:08 INFO  Benchmark_Timer - Begin run: Insertion sort run: 4 for random array for 4000 integers with 100 runs
14.36563823 ms
2021-09-26 00:21:10 INFO  Benchmark_Timer - Begin run: Insertion sort run: 5 for random array for 8000 integers with 100 runs
58.83978765 ms
2021-09-26 00:21:16 INFO  Benchmark_Timer - Begin run: Insertion sort run: 6 for random array for 16000 integers with 100 runs
248.86500187000001 ms

Process finished with exit code 0
```

Build completed successfully in 4 sec, 5 ms (a minute ago)

2. Ordered Array


INFO6205-Assignments – Assignment2Driver.java
File Edit View Navigate Code Refactor Build Run Tools Git Window Help
INFO6205-Assignments 〉 src 〉 main 〉 java 〉 edu 〉 neu 〉 coe 〉 info6205 〉 assignment2 〉 Assignment2Driver 〉 main
Commit to Fall2021
Changes 1 file
Assignment2Driver.java ~/Northeastern/INFO 6205 - Sec 01/Class Repo/INFO62
Amend
Commit
Commit and Push...
GenericSort.java
Assignment2Driver.java
BenchmarkTest.java
TimerTest.java
Benchmark_Timer.java
Benchmark.java
Timer.java
Run: Assignment2Driver
Git Run TODO Problems Messages Debug Terminal Build Dependencies
Event Log
24:23 LF UTF-8 4 spaces Fall2021

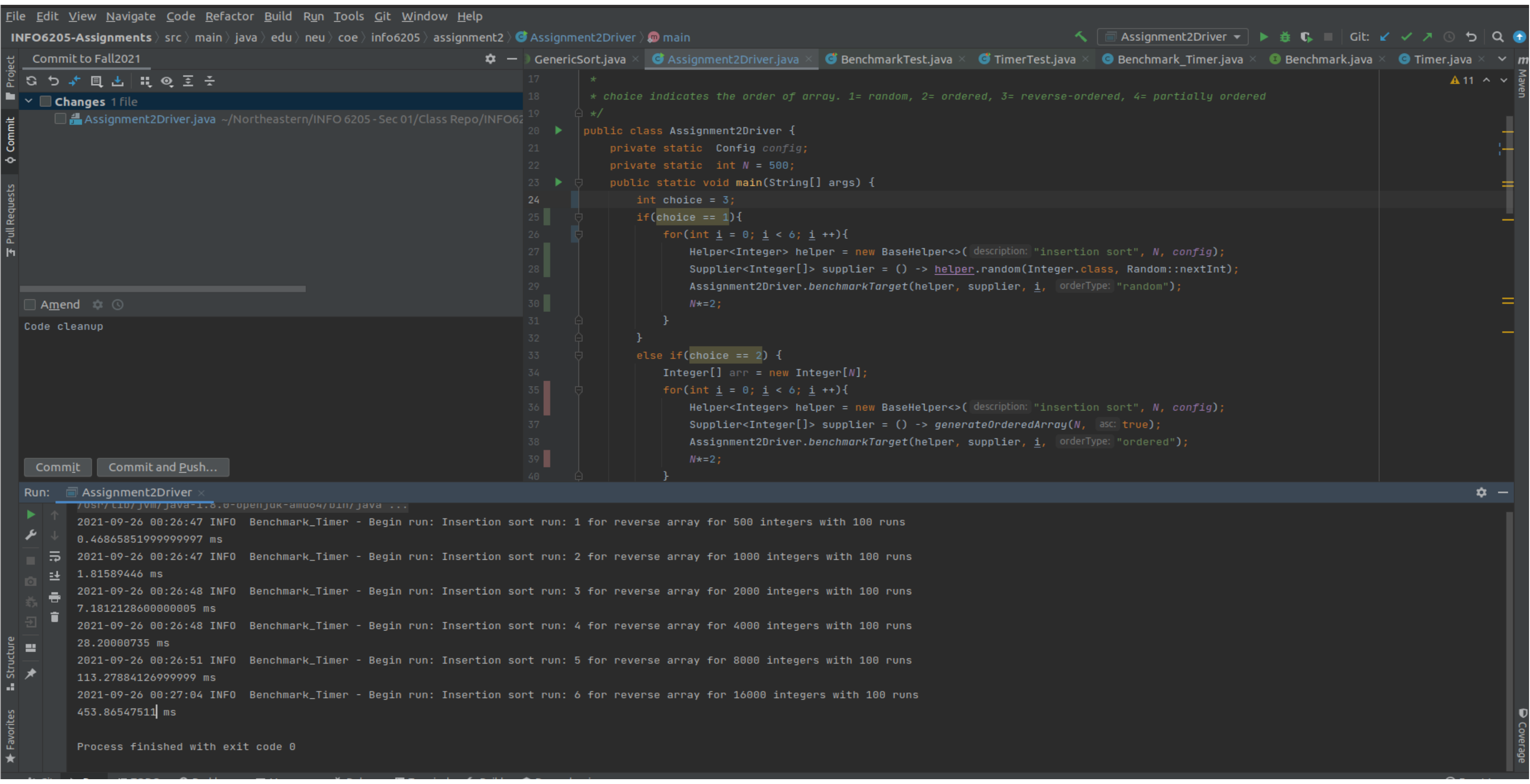
```
Code cleanup
```

```java
 *
 * choice indicates the order of array. 1= random, 2= ordered, 3= reverse-ordered, 4= partially ordered
 */
public class Assignment2Driver {
    private static  Config config;
    private static  int N = 500;
    public static void main(String[] args) {
        int choice = 2;
        if(choice == 1){
            for(int i = 0; i < 6; i ++){
                Helper<Integer> helper = new BaseHelper<>("insertion sort", N, config);
                Supplier<Integer[]> supplier = () -> helper.random(Integer.class, Random::nextInt);
                Assignment2Driver.benchmarkTarget(helper, supplier, i, "random");
                N*=2;
            }
        }
        else if(choice == 2) {
            Integer[] arr = new Integer[N];
            for(int i = 0; i < 6; i ++){
                Helper<Integer> helper = new BaseHelper<>("insertion sort", N, config);
                Supplier<Integer[]> supplier = () -> generateOrderedArray(N, true);
                Assignment2Driver.benchmarkTarget(helper, supplier, i, "ordered");
                N*=2;
            }
```

```
/usr/lib/jvm/java-1.8.0-openjdk-amd64/bin/java ...
2021-09-26 00:24:40 INFO  Benchmark_Timer - Begin run: Insertion sort run: 1 for ordered array for 500 integers with 100 runs
0.02497608 ms
2021-09-26 00:24:40 INFO  Benchmark_Timer - Begin run: Insertion sort run: 2 for ordered array for 1000 integers with 100 runs
0.021207319999999998 ms
2021-09-26 00:24:40 INFO  Benchmark_Timer - Begin run: Insertion sort run: 3 for ordered array for 2000 integers with 100 runs
0.03278076 ms
2021-09-26 00:24:40 INFO  Benchmark_Timer - Begin run: Insertion sort run: 4 for ordered array for 4000 integers with 100 runs
0.00917506 ms
2021-09-26 00:24:40 INFO  Benchmark_Timer - Begin run: Insertion sort run: 5 for ordered array for 8000 integers with 100 runs
0.01274484 ms
2021-09-26 00:24:40 INFO  Benchmark_Timer - Begin run: Insertion sort run: 6 for ordered array for 16000 integers with 100 runs
0.02522791 ms

Process finished with exit code 0
```

All files are up-to-date (a minute ago)

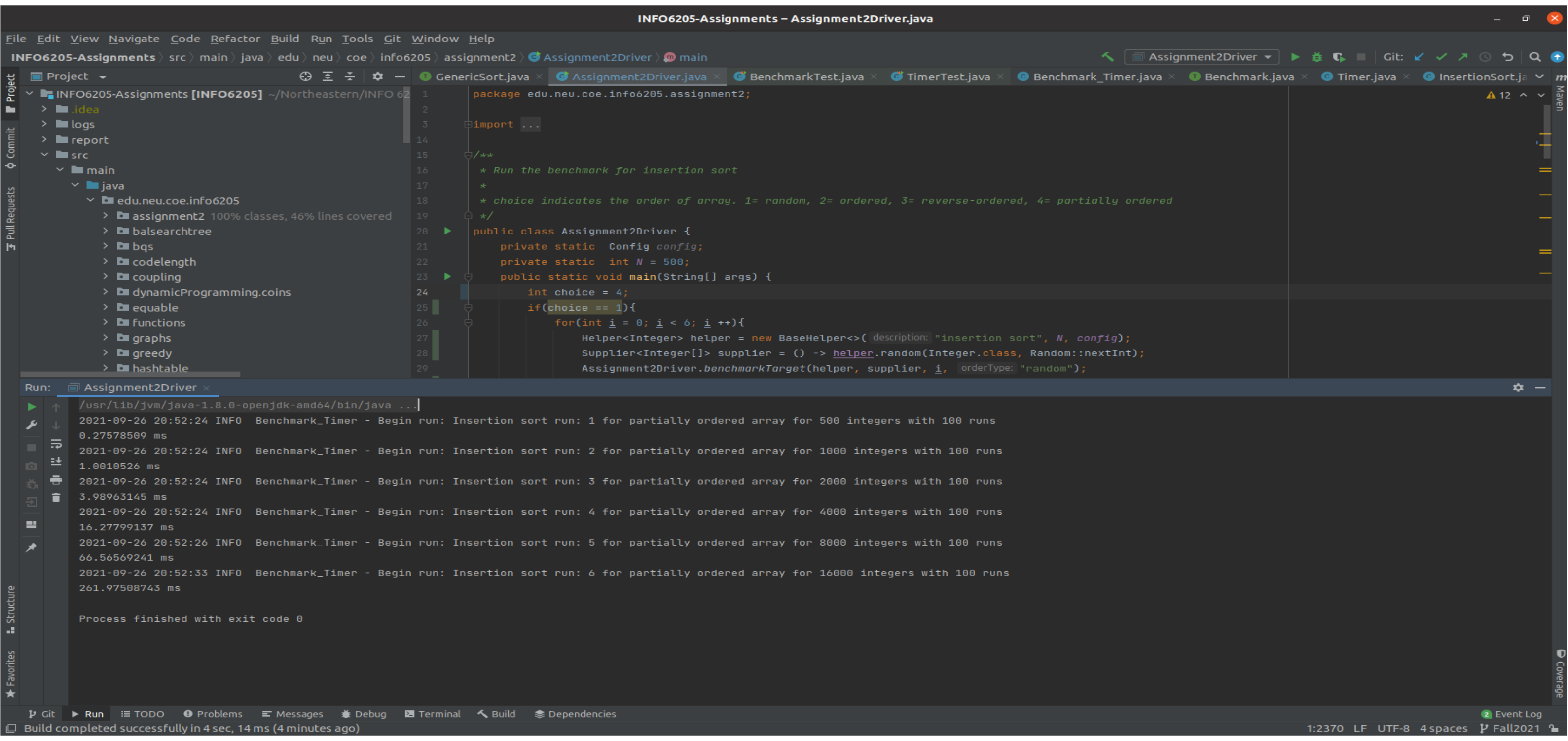
3. Reverse Array



```
 *
 * choice indicates the order of array. 1= random, 2= ordered, 3= reverse-ordered, 4= partially ordered
 */
public class Assignment2Driver {
    private static  Config config;
    private static  int N = 500;
    public static void main(String[] args) {
        int choice = 3;
        if(choice == 1){
            for(int i = 0; i < 6; i ++){
                Helper<Integer> helper = new BaseHelper<>( description: "insertion sort", N, config);
                Supplier<Integer[]> supplier = () -> helper.random(Integer.class, Random::nextInt);
                Assignment2Driver.benchmarkTarget(helper, supplier, i,  orderType: "random");
                N*=2;
            }
        }
        else if(choice == 2) {
            Integer[] arr = new Integer[N];
            for(int i = 0; i < 6; i ++){
                Helper<Integer> helper = new BaseHelper<>( description: "insertion sort", N, config);
                Supplier<Integer[]> supplier = () -> generateOrderedArray(N,  asc: true);
                Assignment2Driver.benchmarkTarget(helper, supplier, i,  orderType: "ordered");
                N*=2;
            }
```

```
/usr/lib/jvm/java-1.8.0-openjdk-amd64/bin/java ...
2021-09-26 00:26:47 INFO  Benchmark_Timer - Begin run: Insertion sort run: 1 for reverse array for 500 integers with 100 runs
0.46865851999999997 ms
2021-09-26 00:26:47 INFO  Benchmark_Timer - Begin run: Insertion sort run: 2 for reverse array for 1000 integers with 100 runs
1.81589446 ms
2021-09-26 00:26:48 INFO  Benchmark_Timer - Begin run: Insertion sort run: 3 for reverse array for 2000 integers with 100 runs
7.1812128600000005 ms
2021-09-26 00:26:48 INFO  Benchmark_Timer - Begin run: Insertion sort run: 4 for reverse array for 4000 integers with 100 runs
28.20000735 ms
2021-09-26 00:26:51 INFO  Benchmark_Timer - Begin run: Insertion sort run: 5 for reverse array for 8000 integers with 100 runs
113.27884126999999 ms
2021-09-26 00:27:04 INFO  Benchmark_Timer - Begin run: Insertion sort run: 6 for reverse array for 16000 integers with 100 runs
453.86547511 ms

Process finished with exit code 0
```

4. Partially Ordered Array



INFO6205-Assignments – Assignment2Driver.java

```
package edu.neu.coe.info6205.assignment2;

import ...

/**
 * Run the benchmark for insertion sort
 *
 * choice indicates the order of array. 1= random, 2= ordered, 3= reverse-ordered, 4= partially ordered
 */
public class Assignment2Driver {
    private static  Config config;
    private static  int N = 500;
    public static void main(String[] args) {
        int choice = 4;
        if(choice == 1){
            for(int i = 0; i < 6; i ++){
                Helper<Integer> helper = new BaseHelper<>( description: "insertion sort", N, config);
                Supplier<Integer[]> supplier = () -> helper.random(Integer.class, Random::nextInt);
                Assignment2Driver.benchmarkTarget(helper, supplier, i,  orderType: "random");
```

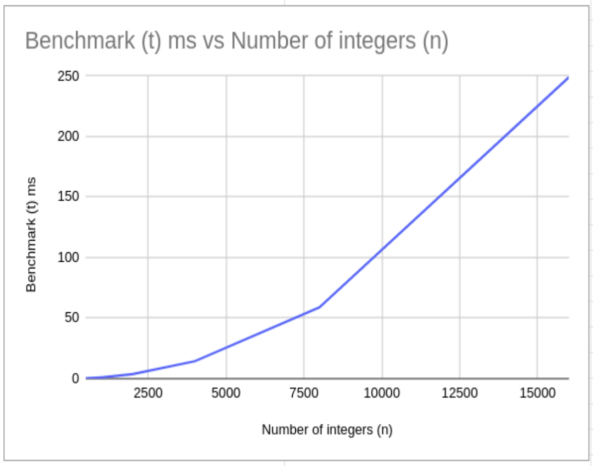
```
/usr/lib/jvm/java-1.8.0-openjdk-amd64/bin/java ...
2021-09-26 20:52:24 INFO  Benchmark_Timer - Begin run: Insertion sort run: 1 for partially ordered array for 500 integers with 100 runs
0.27578509 ms
2021-09-26 20:52:24 INFO  Benchmark_Timer - Begin run: Insertion sort run: 2 for partially ordered array for 1000 integers with 100 runs
1.0010526 ms
2021-09-26 20:52:24 INFO  Benchmark_Timer - Begin run: Insertion sort run: 3 for partially ordered array for 2000 integers with 100 runs
3.98963145 ms
2021-09-26 20:52:24 INFO  Benchmark_Timer - Begin run: Insertion sort run: 4 for partially ordered array for 4000 integers with 100 runs
16.27799137 ms
2021-09-26 20:52:26 INFO  Benchmark_Timer - Begin run: Insertion sort run: 5 for partially ordered array for 8000 integers with 100 runs
66.56569241 ms
2021-09-26 20:52:33 INFO  Benchmark_Timer - Begin run: Insertion sort run: 6 for partially ordered array for 16000 integers with 100 runs
261.97508743 ms

Process finished with exit code 0
```

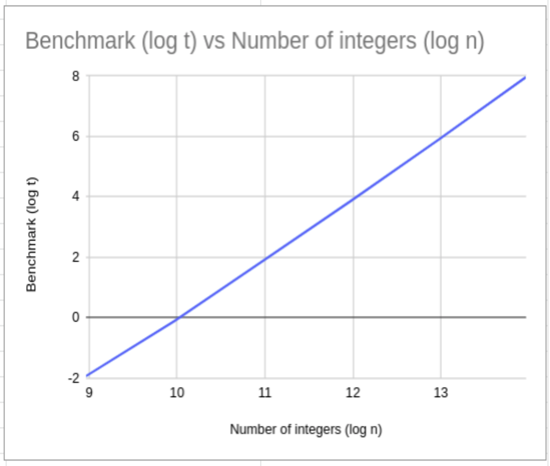
Build completed successfully in 4 sec, 14 ms (4 minutes ago)

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

1. RandomArray

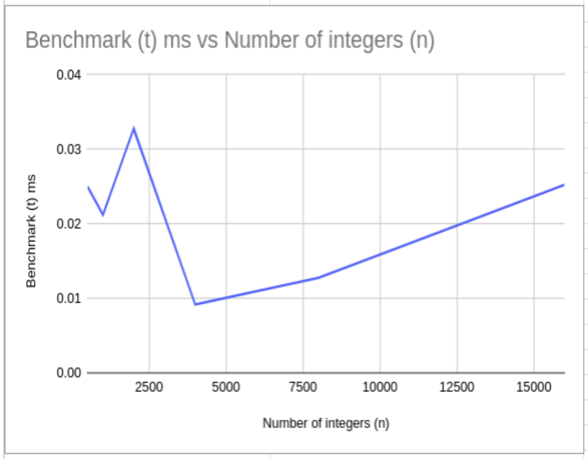
Number of integers (n)	Benchmark (t) ms	Number of integers (log n)	Benchmark (log t)
500	0.26316616	8.965784285	-1.925954107
1000	0.92661578	9.965784285	-0.1099568436
2000	3.63312282	10.96578428	1.861210138
4000	14.36563823	11.96578428	3.844550185
8000	58.83978765	12.96578428	5.878720135
16000	248.8650019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

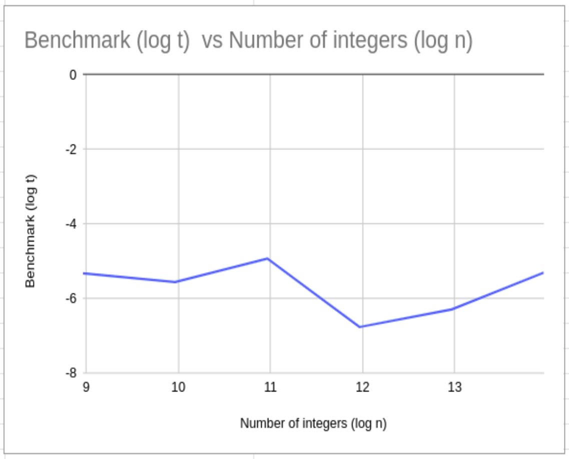




2. Ordered Array

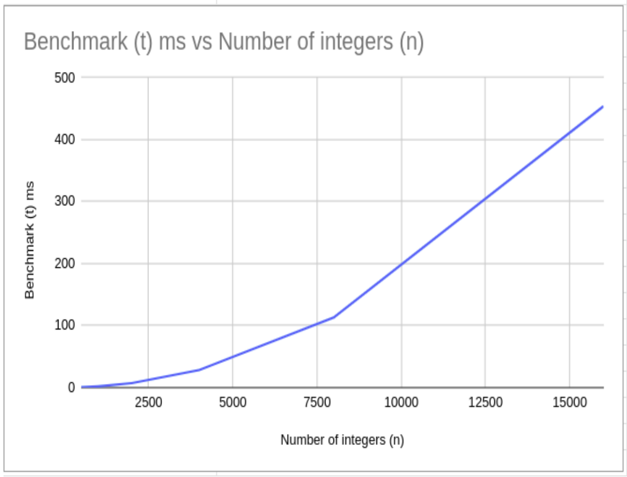
A	B	C	D	E
Number of integers (n)	Benchmark (t) ms		Number of integers (log n)	Benchmark (log t)
500	0.02497608		8.965784285	-5.323309126
1000	0.02120732		9.965784285	-5.559293873
2000	0.03278076		10.96578428	-4.931006887
4000	0.00917506		11.96578428	-6.768066692
8000	0.01274484		12.96578428	-6.293942928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

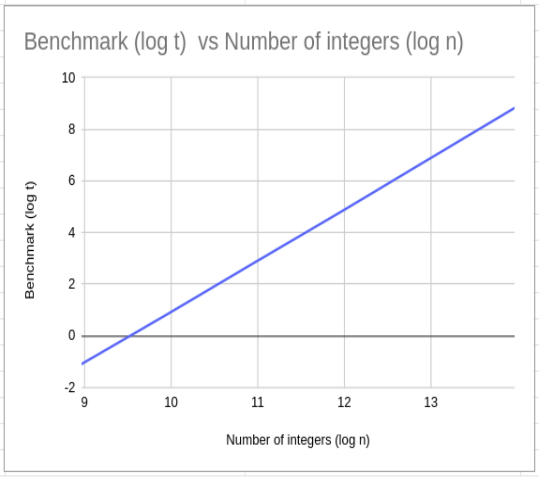




3. Reverse Array

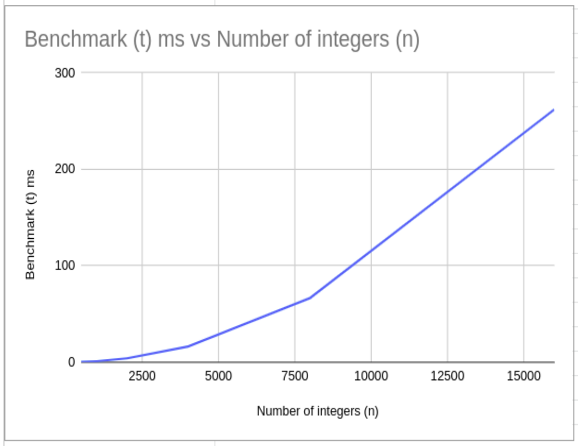
Number of integers (n)	Benchmark (t) ms		Number of integers (log n)	Benchmark (log t)
500	0.46865852		8.965784285	-1.093390984
1000	1.81589446		9.965784285	0.8606803555
2000	7.18121286		10.96578428	2.844227526
4000	28.20000735		11.96578428	4.817623634
8000	113.2788413		12.96578428	6.823734603
16000	453.8654751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

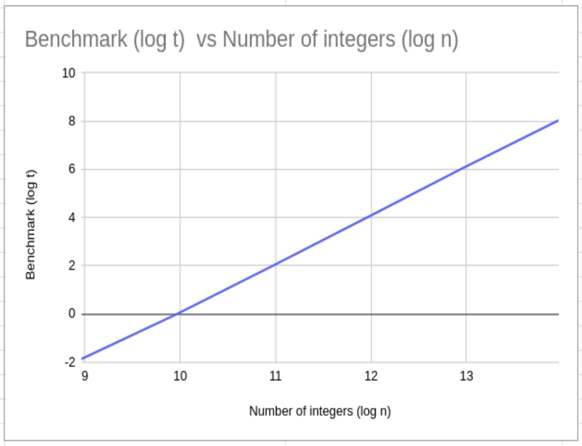




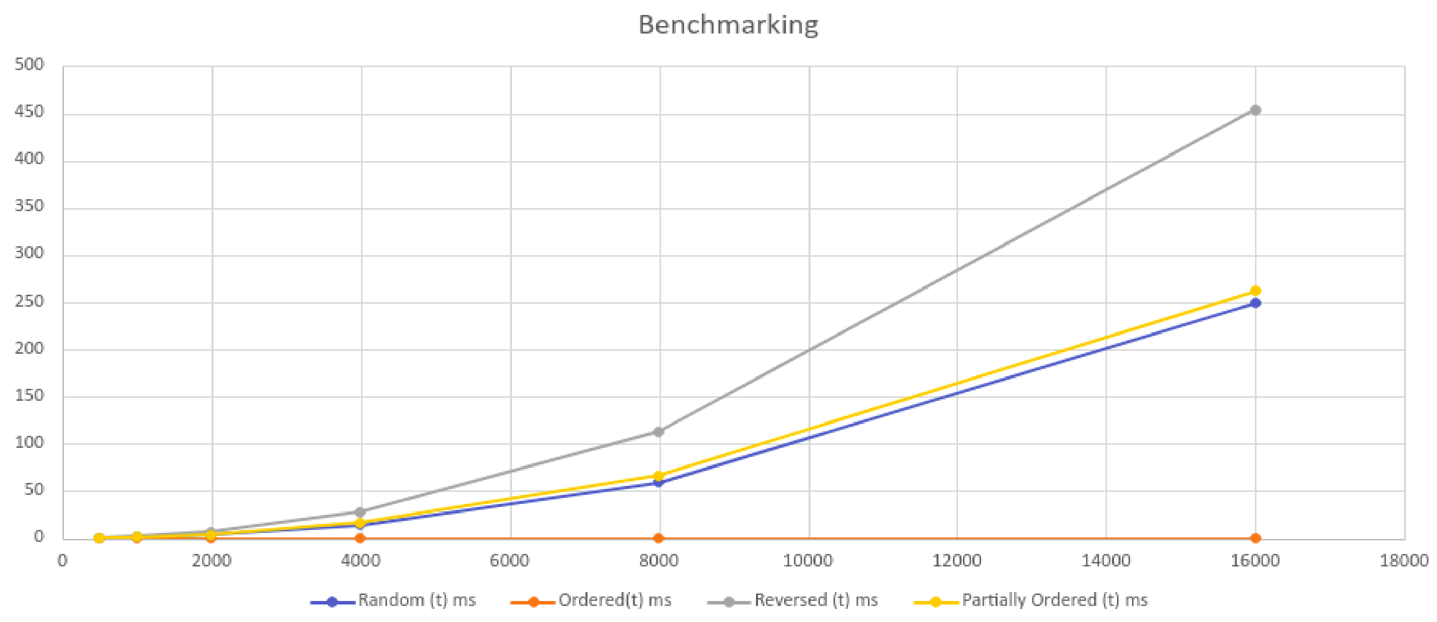
4. Partially Ordered Array

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: ["t" on the Y axis and "n" on the X axis]



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