Xinhui Zhao (001560851)

Program Structures & Algorithms

Fall 2021

Assignment No. 2

⊙ Task (List down the tasks performed in the Assignment)

(Part 1) 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.

(Part 2) Implement *InsertionSort* (in the *InsertionSort* class) by simply looking up the insertion code used by *Arrays.sort.* If you have the *instrument = true* setting in *test/resources/config.ini*, then you will need to use the *helper* methods for comparing and swapping (so that they properly count the number of swaps/compares). The easiest is to use the *helper.swapStableConditional* method, continuing if it returns true, otherwise breaking the loop. Alternatively, if you are not using instrumenting, then you can write (or copy) your own compare/swap code. Either way, you must run the unit tests in *InsertionSortTest*.

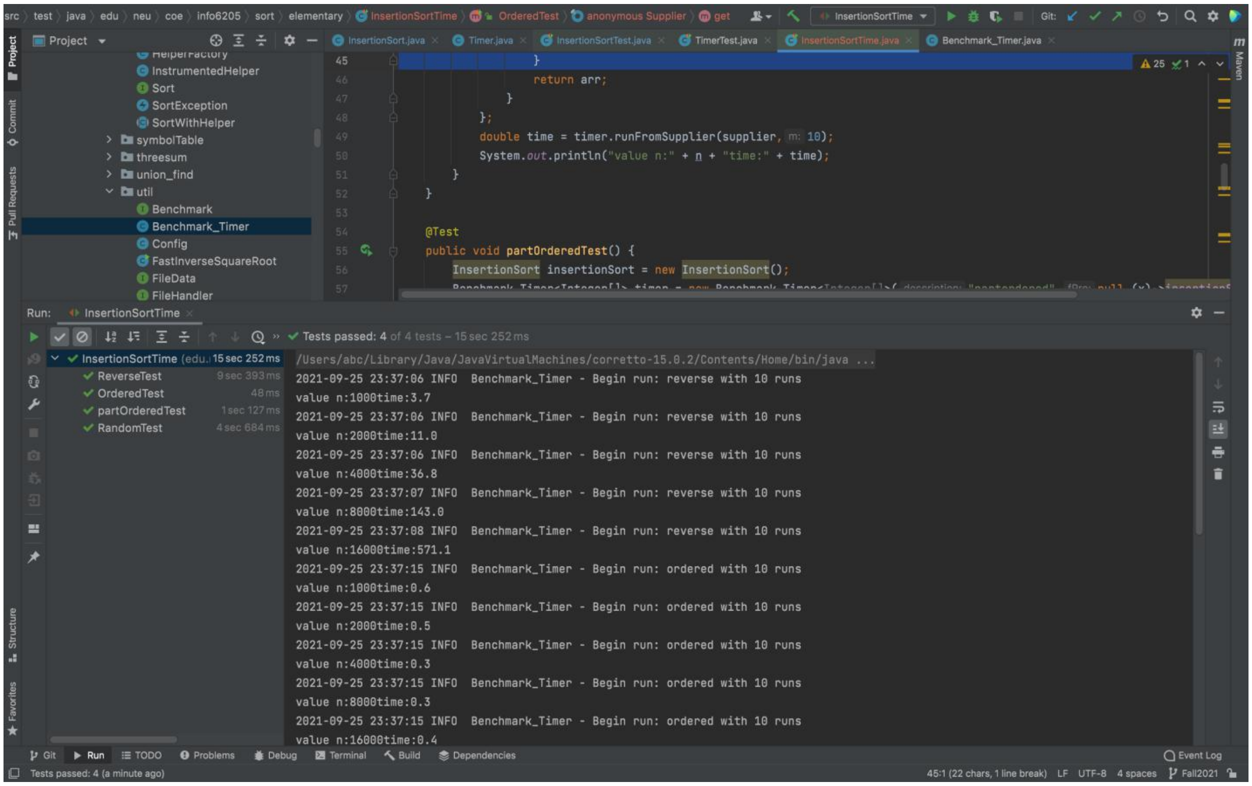
(Part 3) Implement a main program (or you could do it via your own unit tests) to actually run the following benchmarks: measure the running times of this sort, using four different initial array ordering situations: random, ordered, partially-ordered and reverse-ordered. I suggest that your arrays to 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.

⊙ Relationship Conclusion:

Total required time: Ordered Test 48ms < Part Ordered Test 1s 127ms < Random Test 4s 684ms < Reverse Test 9s 393ms

⊙ Evidence to support the conclusion:

1. Output (Snapshot of Code output in the terminal)

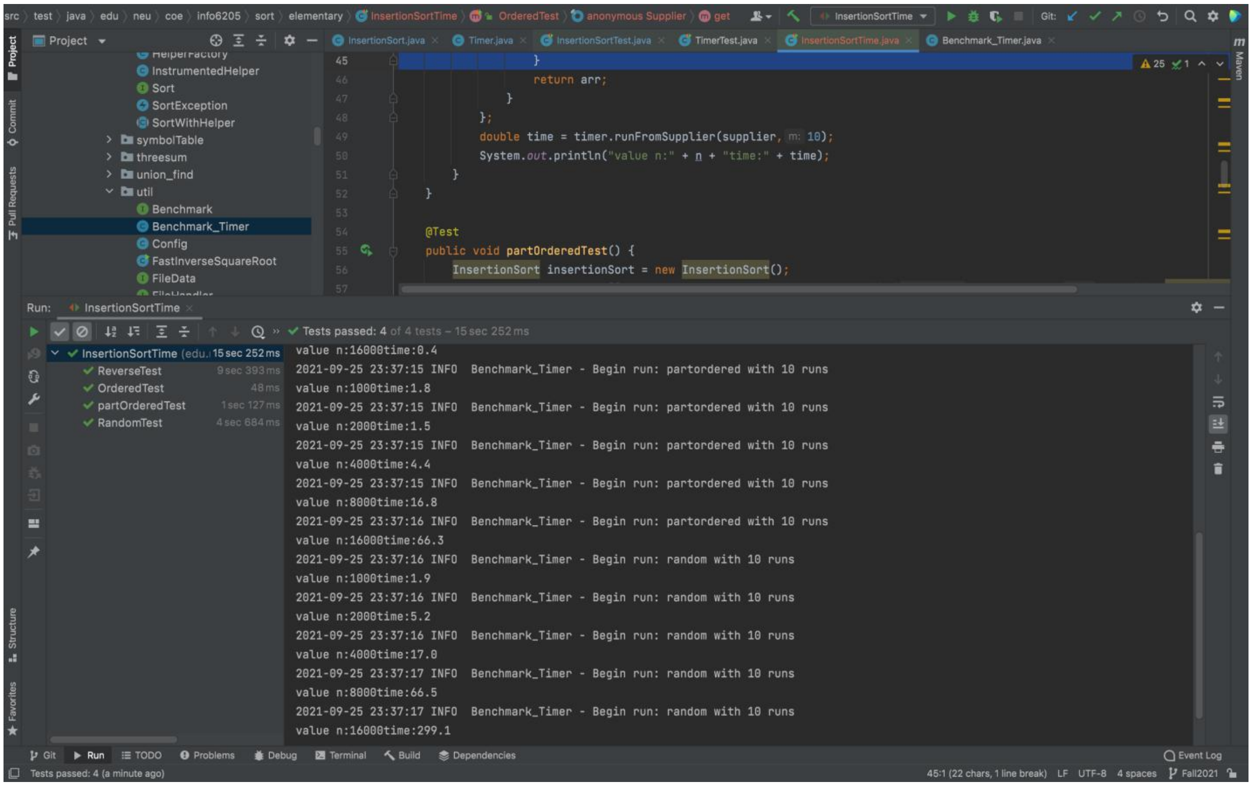


```
                }
                return arr;
            }
        };
        double time = timer.runFromSupplier(supplier, m: 10);
        System.out.println("value n:" + n + "time:" + time);
    }
}

@Test
public void partOrderedTest() {
    InsertionSort insertionSort = new InsertionSort();
```

```
Tests passed: 4 of 4 tests – 15 sec 252 ms

InsertionSortTime (edu.i 15 sec 252 ms
  ReverseTest        9 sec 393 ms
  OrderedTest               48 ms
  partOrderedTest    1 sec 127 ms
  RandomTest         4 sec 684 ms
```

```
/Users/abc/Library/Java/JavaVirtualMachines/corretto-15.0.2/Contents/Home/bin/java ...
2021-09-25 23:37:06 INFO  Benchmark_Timer - Begin run: reverse with 10 runs
value n:1000time:3.7
2021-09-25 23:37:06 INFO  Benchmark_Timer - Begin run: reverse with 10 runs
value n:2000time:11.0
2021-09-25 23:37:06 INFO  Benchmark_Timer - Begin run: reverse with 10 runs
value n:4000time:36.8
2021-09-25 23:37:07 INFO  Benchmark_Timer - Begin run: reverse with 10 runs
value n:8000time:143.0
2021-09-25 23:37:08 INFO  Benchmark_Timer - Begin run: reverse with 10 runs
value n:16000time:571.1
2021-09-25 23:37:15 INFO  Benchmark_Timer - Begin run: ordered with 10 runs
value n:1000time:0.6
2021-09-25 23:37:15 INFO  Benchmark_Timer - Begin run: ordered with 10 runs
value n:2000time:0.5
2021-09-25 23:37:15 INFO  Benchmark_Timer - Begin run: ordered with 10 runs
value n:4000time:0.3
2021-09-25 23:37:15 INFO  Benchmark_Timer - Begin run: ordered with 10 runs
value n:8000time:0.3
2021-09-25 23:37:15 INFO  Benchmark_Timer - Begin run: ordered with 10 runs
value n:16000time:0.4
```

Tests passed: 4 (a minute ago)



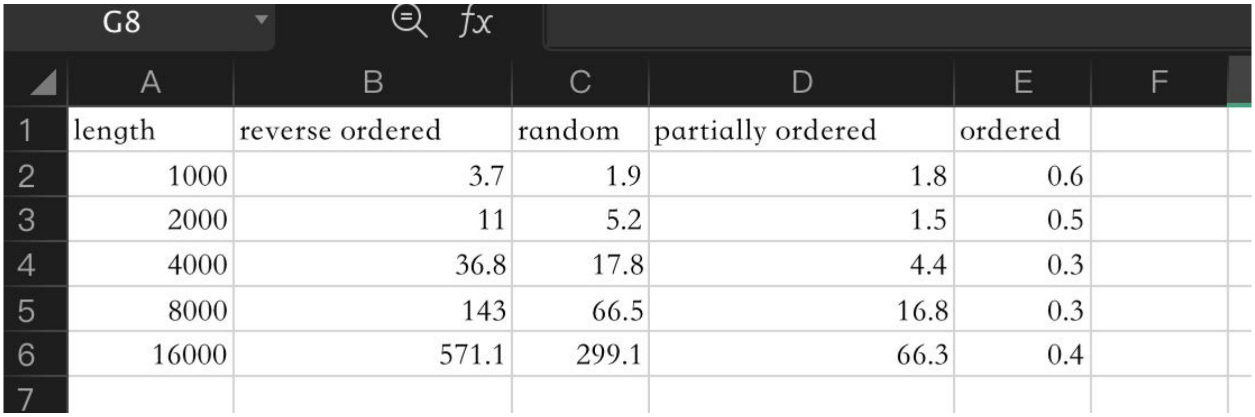
```
                }
                return arr;
            }
        };
        double time = timer.runFromSupplier(supplier, m: 10);
        System.out.println("value n:" + n + "time:" + time);
    }
}

@Test
public void partOrderedTest() {
    InsertionSort insertionSort = new InsertionSort();
```

```
Tests passed: 4 of 4 tests – 15 sec 252 ms

InsertionSortTime (edu.i 15 sec 252 ms
  ReverseTest        9 sec 393 ms
  OrderedTest               48 ms
  partOrderedTest    1 sec 127 ms
  RandomTest         4 sec 684 ms
```

```
value n:16000time:0.4
2021-09-25 23:37:15 INFO  Benchmark_Timer - Begin run: partordered with 10 runs
value n:1000time:1.8
2021-09-25 23:37:15 INFO  Benchmark_Timer - Begin run: partordered with 10 runs
value n:2000time:1.5
2021-09-25 23:37:15 INFO  Benchmark_Timer - Begin run: partordered with 10 runs
value n:4000time:4.4
2021-09-25 23:37:15 INFO  Benchmark_Timer - Begin run: partordered with 10 runs
value n:8000time:16.8
2021-09-25 23:37:16 INFO  Benchmark_Timer - Begin run: partordered with 10 runs
value n:16000time:66.3
2021-09-25 23:37:16 INFO  Benchmark_Timer - Begin run: random with 10 runs
value n:1000time:1.9
2021-09-25 23:37:16 INFO  Benchmark_Timer - Begin run: random with 10 runs
value n:2000time:5.2
2021-09-25 23:37:16 INFO  Benchmark_Timer - Begin run: random with 10 runs
value n:4000time:17.0
2021-09-25 23:37:17 INFO  Benchmark_Timer - Begin run: random with 10 runs
value n:8000time:66.5
2021-09-25 23:37:17 INFO  Benchmark_Timer - Begin run: random with 10 runs
value n:16000time:299.1
```

Tests passed: 4 (a minute ago)

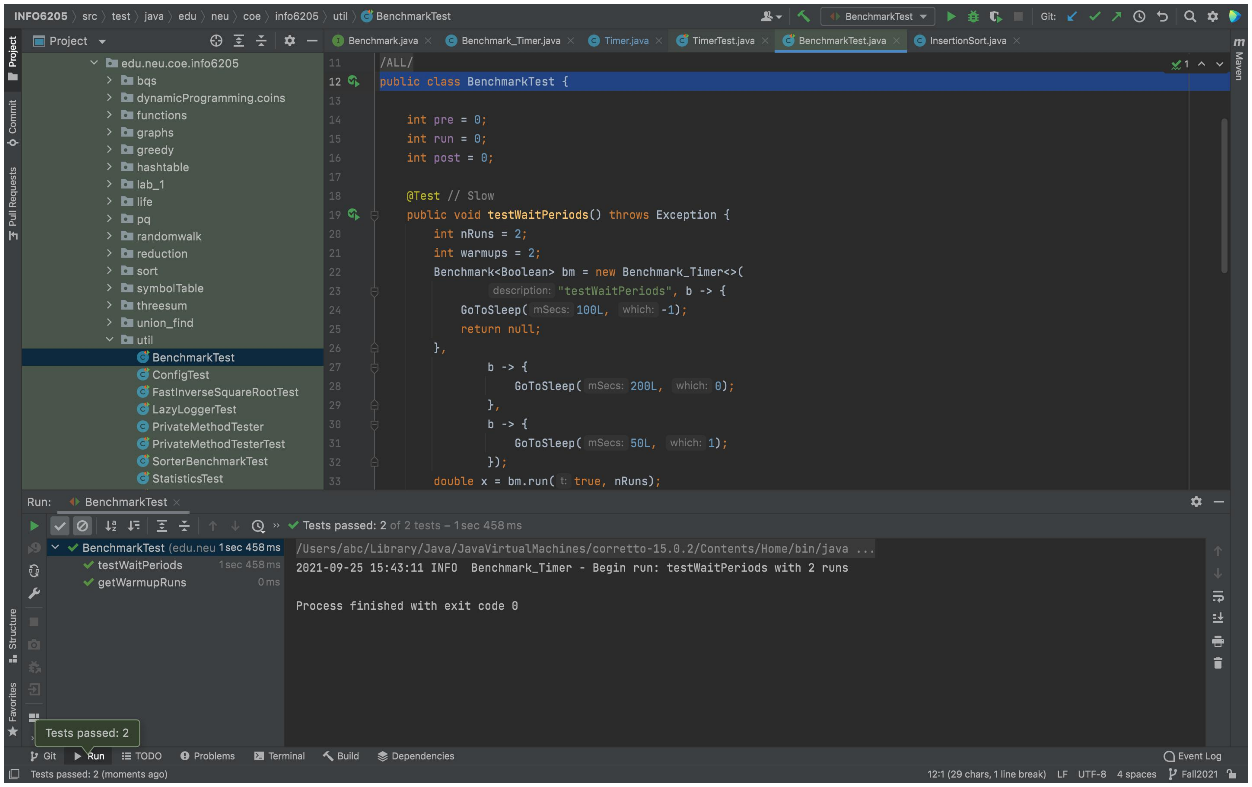
2. Graphical Representation(Observations from experiments should be tabulated and analyzed by plotting graphs(usually in excel) to arrive on the relationship conclusion)



length	reverse ordered	random	partially ordered	ordered
1000	3.7	1.9	1.8	0.6
2000	11	5.2	1.5	0.5
4000	36.8	17.8	4.4	0.3
8000	143	66.5	16.8	0.3
16000	571.1	299.1	66.3	0.4

- ⊙ Unit tests result:(Snapshot of successful unit test run)

Benchmark Test Result



```java
/ALL/
public class BenchmarkTest {

    int pre = 0;
    int run = 0;
    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);
```

Tests passed: 2 of 2 tests – 1 sec 458 ms

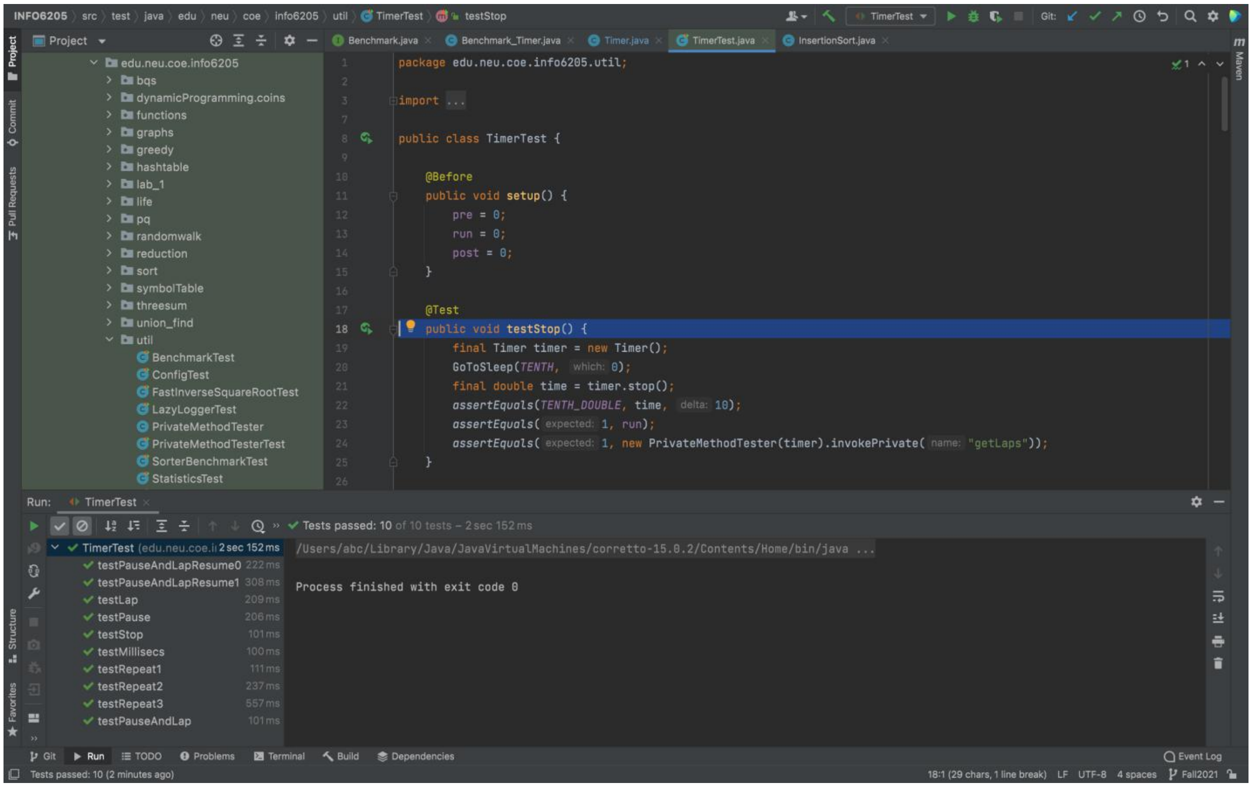
```
BenchmarkTest (edu.neu 1 sec 458 ms
  testWaitPeriods        1 sec 458 ms
  getWarmupRuns                  0 ms
```

```
/Users/abc/Library/Java/JavaVirtualMachines/corretto-15.0.2/Contents/Home/bin/java ...
2021-09-25 15:43:11 INFO  Benchmark_Timer - Begin run: testWaitPeriods with 2 runs

Process finished with exit code 0
```

Tests passed: 2

Tests passed: 2 (moments ago)

Timer Test Result



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

import ...

public class TimerTest {

    @Before
    public void setup() {
        pre = 0;
        run = 0;
        post = 0;
    }

    @Test
    public void testStop() {
        final Timer timer = new Timer();
        GoToSleep(TENTH,  which: 0);
        final double time = timer.stop();
        assertEquals(TENTH_DOUBLE, time,  delta: 10);
        assertEquals( expected: 1, run);
        assertEquals( expected: 1, new PrivateMethodTester(timer).invokePrivate( name: "getLaps"));
    }
```

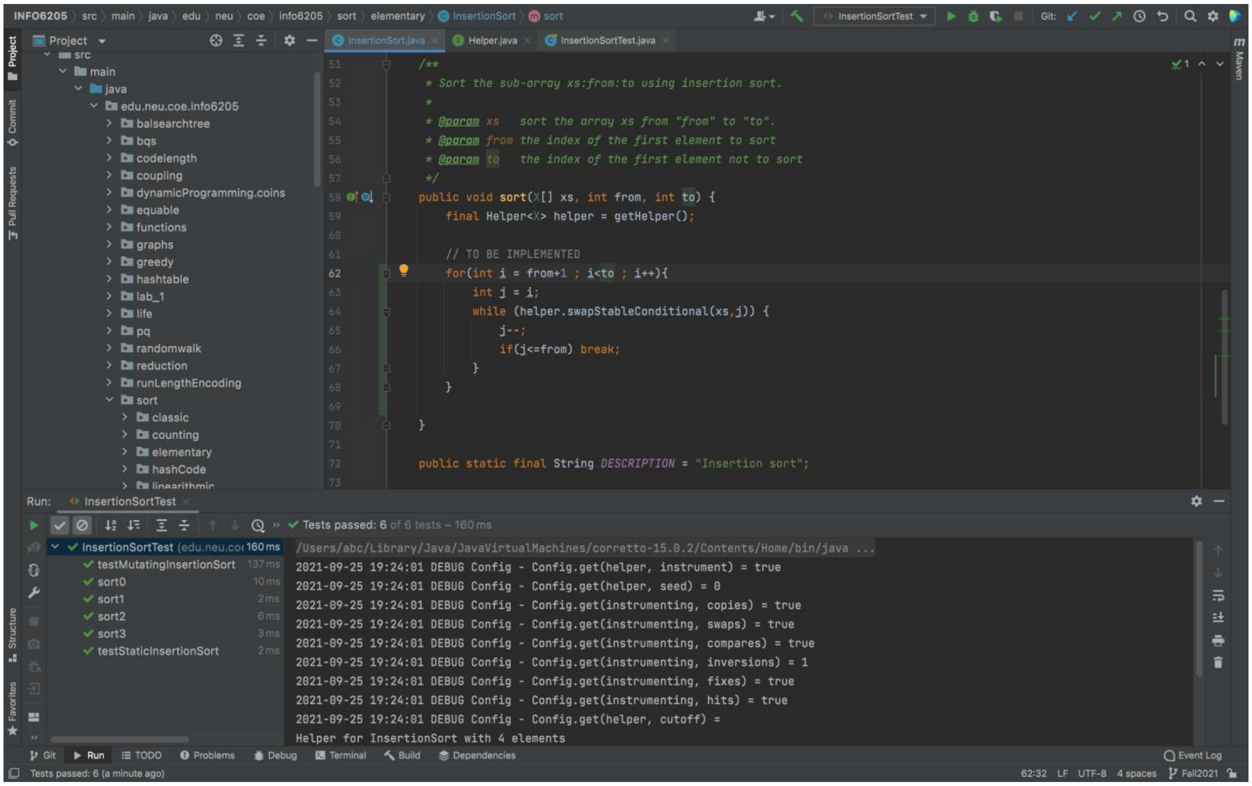
✓ Tests passed: 10 of 10 tests – 2 sec 152 ms

```
✓ TimerTest (edu.neu.coe.i 2 sec 152 ms
  ✓ testPauseAndLapResume0    222 ms
  ✓ testPauseAndLapResume1    308 ms
  ✓ testLap                   209 ms
  ✓ testPause                 206 ms
  ✓ testStop                  101 ms
  ✓ testMillisecs             100 ms
  ✓ testRepeat1               111 ms
  ✓ testRepeat2               237 ms
  ✓ testRepeat3               557 ms
  ✓ testPauseAndLap           101 ms
```

```
/Users/abc/Library/Java/JavaVirtualMachines/corretto-15.0.2/Contents/Home/bin/java ...

Process finished with exit code 0
```

Tests passed: 10 (2 minutes ago)

InsertionSort Test Result



```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();

        // TO BE IMPLEMENTED
        for(int i = from+1 ; i<to ; i++){
            int j = i;
            while (helper.swapStableConditional(xs,j)) {
                j--;
                if(j<=from) break;
            }
        }

    }

    public static final String DESCRIPTION = "Insertion sort";
```

✓ Tests passed: 6 of 6 tests – 160 ms

```
✓ InsertionSortTest (edu.neu.coe 160 ms
  ✓ testMutatingInsertionSort  137 ms
  ✓ sort0                       10 ms
  ✓ sort1                        2 ms
  ✓ sort2                        6 ms
  ✓ sort3                        3 ms
  ✓ testStaticInsertionSort      2 ms
```

```
/Users/abc/Library/Java/JavaVirtualMachines/corretto-15.0.2/Contents/Home/bin/java ...
2021-09-25 19:24:01 DEBUG Config - Config.get(helper, instrument) = true
2021-09-25 19:24:01 DEBUG Config - Config.get(helper, seed) = 0
2021-09-25 19:24:01 DEBUG Config - Config.get(instrumenting, copies) = true
2021-09-25 19:24:01 DEBUG Config - Config.get(instrumenting, swaps) = true
2021-09-25 19:24:01 DEBUG Config - Config.get(instrumenting, compares) = true
2021-09-25 19:24:01 DEBUG Config - Config.get(instrumenting, inversions) = 1
2021-09-25 19:24:01 DEBUG Config - Config.get(instrumenting, fixes) = true
2021-09-25 19:24:01 DEBUG Config - Config.get(instrumenting, hits) = true
2021-09-25 19:24:01 DEBUG Config - Config.get(helper, cutoff) =
Helper for InsertionSort with 4 elements
```

Tests passed: 6 (a minute ago)