Program Structures and Algorithms

Spring 2024

NAME: Siddharth Dumbre

NUID: 002247119

GITHUB LINK: https://github.com/dumbresi/Info-6205-spring2024

Task: Benchmark

Observation:

- Random Ordered

Array Length	Runtime in ms
100	0.12176039
200	0.39804616000000004
400	0.49619586
800	1.4768011300000001
1600	5.97964293

- Partially ordered

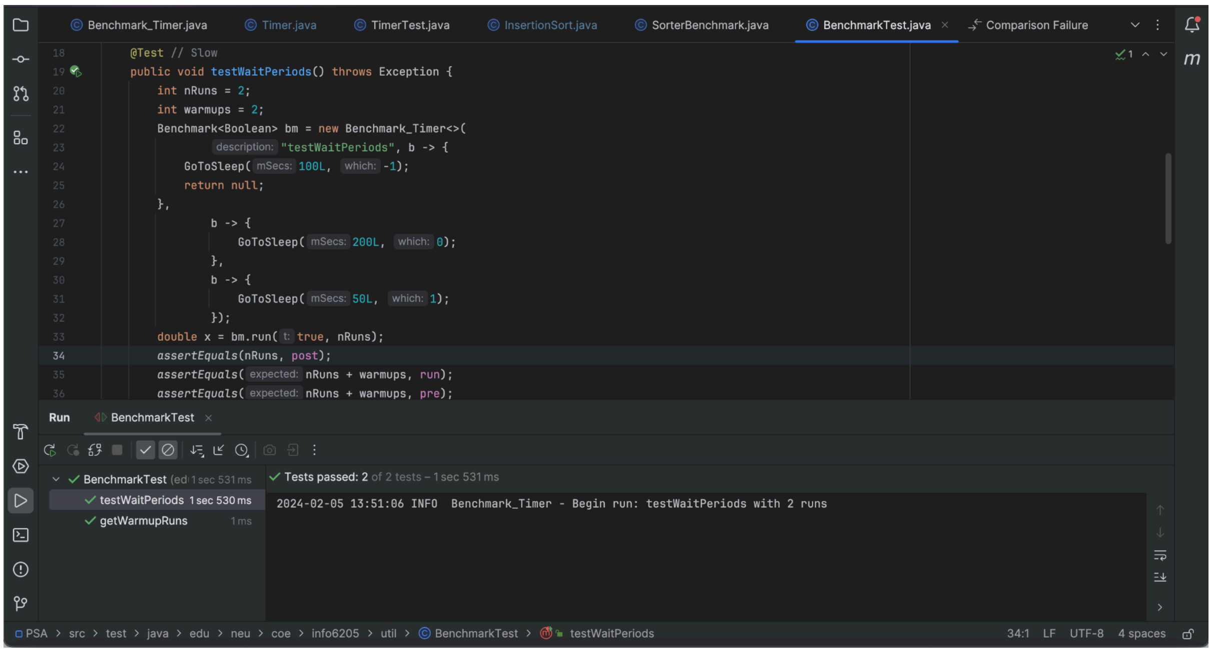
Array Length	Runtime in ms
100	0.25232501
200	0.28879383000000003
400	0.60559833
800	0.69206495
1600	2.32187962

- Reverse ordered

Array Length	Runtime in ms
100	0.22033875
200	0.31638787
400	1.1395463
800	2.87590125
1600	11.38694501

Unit Test Screenshots:

Benchmarktest unit cases



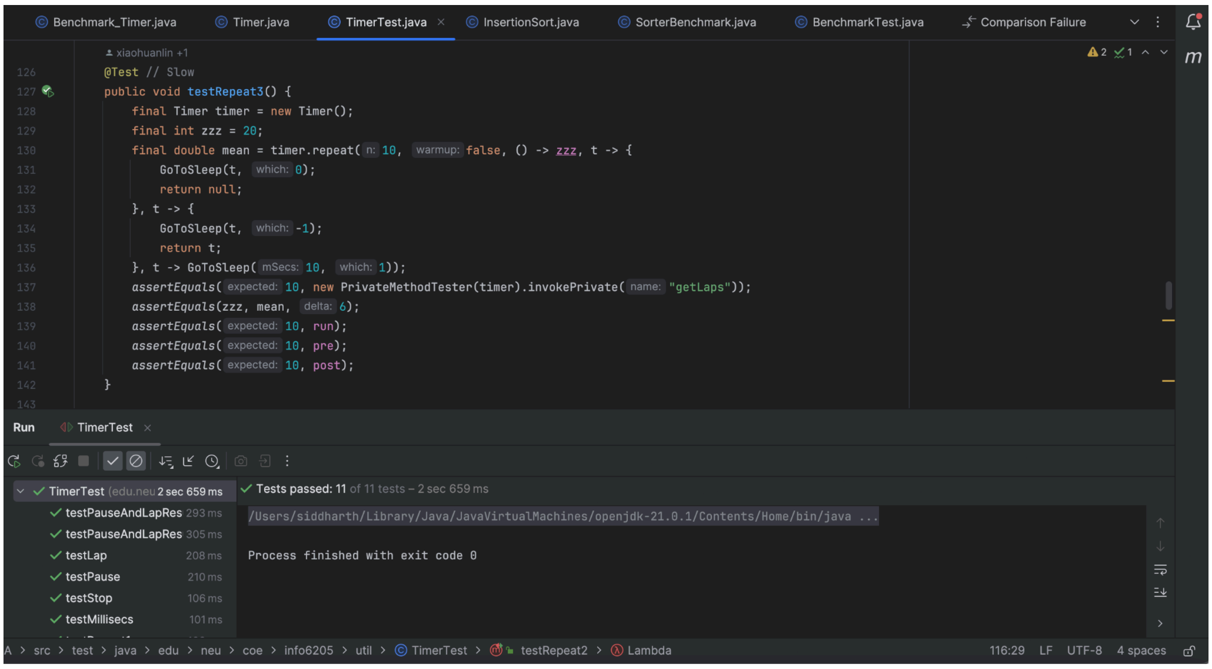
```
    @Test // Slow
    public void testWaitPeriods() throws Exception {
        int nRuns = 2;
        int warmups = 2;
        Benchmark<Boolean> bm = new Benchmark_Timer<>(
                description: "testWaitPeriods", b -> {
            GoToSleep( mSecs: 100L,  which: -1);
            return 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);
```

```
✓ BenchmarkTest (ed 1 sec 531 ms
  ✓ testWaitPeriods 1 sec 530 ms
  ✓ getWarmupRuns 1 ms

✓ Tests passed: 2 of 2 tests – 1 sec 531 ms

2024-02-05 13:51:06 INFO  Benchmark_Timer - Begin run: testWaitPeriods with 2 runs
```

TimerTest unit cases



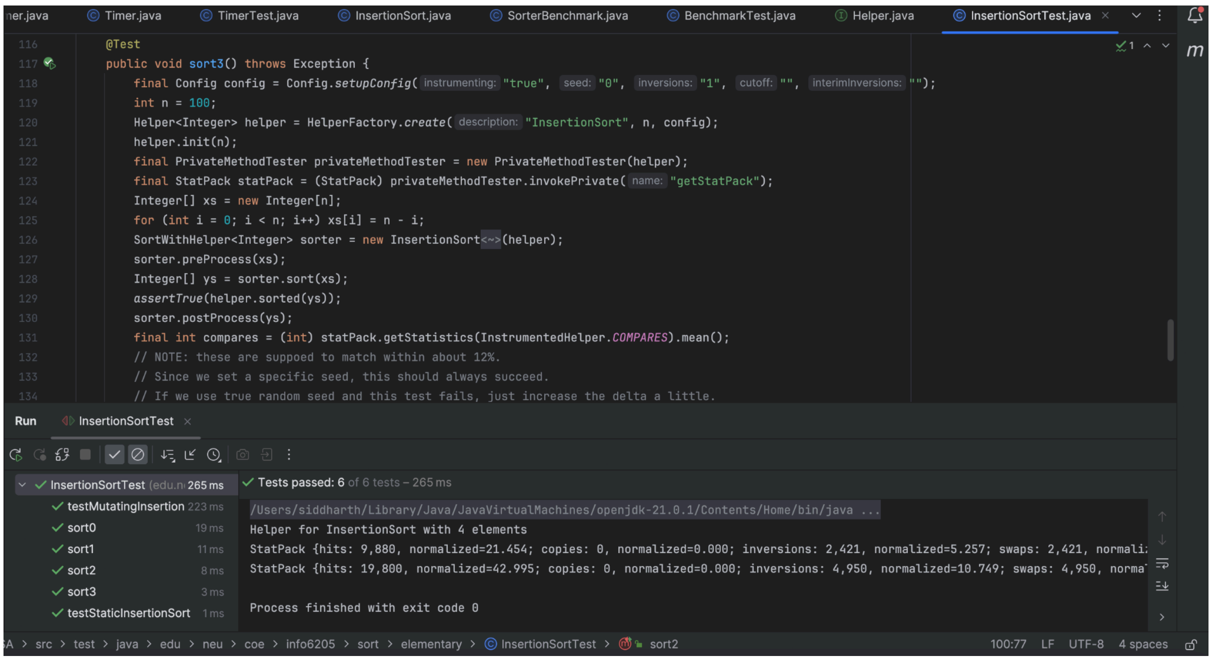
```java
@Test // Slow
public void testRepeat3() {
    final Timer timer = new Timer();
    final int zzz = 20;
    final double mean = timer.repeat( n: 10,  warmup: false, () -> zzz, t -> {
        GoToSleep(t,  which: 0);
        return null;
    }, t -> {
        GoToSleep(t,  which: -1);
        return t;
    }, t -> GoToSleep( mSecs: 10,  which: 1));
    assertEquals( expected: 10, new PrivateMethodTester(timer).invokePrivate( name: "getLaps"));
    assertEquals(zzz, mean,  delta: 6);
    assertEquals( expected: 10, run);
    assertEquals( expected: 10, pre);
    assertEquals( expected: 10, post);
}
```

```
TimerTest (edu.neu 2 sec 659 ms
  testPauseAndLapRes 293 ms
  testPauseAndLapRes 305 ms
  testLap 208 ms
  testPause 210 ms
  testStop 106 ms
  testMillisecs 101 ms
```

```
Tests passed: 11 of 11 tests – 2 sec 659 ms
/Users/siddharth/Library/Java/JavaVirtualMachines/openjdk-21.0.1/Contents/Home/bin/java ...

Process finished with exit code 0
```

InsertionSortTest unit cases



```java
@Test
public void sort3() throws Exception {
    final Config config = Config.setupConfig( instrumenting: "true",  seed: "0",  inversions: "1",  cutoff: "",  interimInversions: "");
    int n = 100;
    Helper<Integer> helper = HelperFactory.create( description: "InsertionSort", n, config);
    helper.init(n);
    final PrivateMethodTester privateMethodTester = new PrivateMethodTester(helper);
    final StatPack statPack = (StatPack) privateMethodTester.invokePrivate( name: "getStatPack");
    Integer[] xs = new Integer[n];
    for (int i = 0; i < n; i++) xs[i] = n - i;
    SortWithHelper<Integer> sorter = new InsertionSort<~>(helper);
    sorter.preProcess(xs);
    Integer[] ys = sorter.sort(xs);
    assertTrue(helper.sorted(ys));
    sorter.postProcess(ys);
    final int compares = (int) statPack.getStatistics(InstrumentedHelper.COMPARES).mean();
    // NOTE: these are suppoed to match within about 12%.
    // Since we set a specific seed, this should always succeed.
    // If we use true random seed and this test fails, just increase the delta a little.
```

```
InsertionSortTest (edu.n 265 ms
  testMutatingInsertion 223 ms
  sort0 19 ms
  sort1 11 ms
  sort2 8 ms
  sort3 3 ms
  testStaticInsertionSort 1 ms
```

```
Tests passed: 6 of 6 tests – 265 ms
/Users/siddharth/Library/Java/JavaVirtualMachines/openjdk-21.0.1/Contents/Home/bin/java ...
Helper for InsertionSort with 4 elements
StatPack {hits: 9,880, normalized=21.454; copies: 0, normalized=0.000; inversions: 2,421, normalized=5.257; swaps: 2,421, normali
StatPack {hits: 19,800, normalized=42.995; copies: 0, normalized=0.000; inversions: 4,950, normalized=10.749; swaps: 4,950, norma

Process finished with exit code 0
```