Program Structures and Algorithms
Spring 2023(SEC 1)

NAME: Harshil Shah
NUID: 002780887
Assignment: 3

Task: Your task for this assignment is in three parts.

(Part 1) You are to implement three (3) methods (repeat, getClock, and toMillisecs) 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:

1. For an array of random elements, the time complexity of insertion sort will be O(n^2) in the average and worst case scenario, as each element may need to be shifted multiple times before being inserted into the correct position.
2. For an array with elements in reverse order, the time complexity of insertion sort will also be O(n^2) as each element will need to be shifted to the beginning of the array for every iteration, making it the worst case scenario for insertion sort.
3. For an array which is already sorted, the time complexity of insertion sort will be O(n) as each element will only need to be checked and not shifted, making it the best case scenario for insertion sort.
4. For an array which is sorted partially, the time complexity will be somewhere between O(n) and O(n^2), depending on the degree of sortedness of the array. The more sorted the array is, the closer the time complexity will be to O(n).

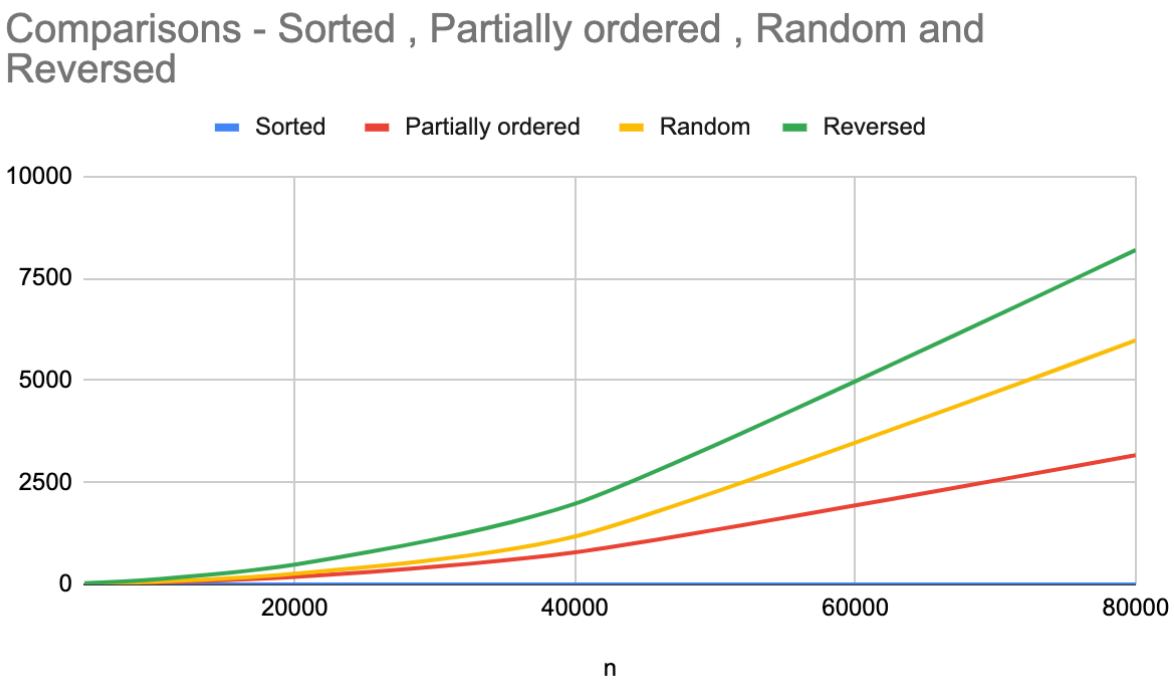
Evidence to support that conclusion:

n	Sorted	Partially ordered	Random	Reversed
5000	0	11.7	15.15	30.2
10000	0	46.3	61.9	122.7
20000	0	181.45	260.6	484.4
40000	0.05	784	1172.9	1974.95
80000	0.1	3165.75	5978	8196.95

Observation: The graph for the time complexity of insertion sort for each of the use cases will grow as follows:

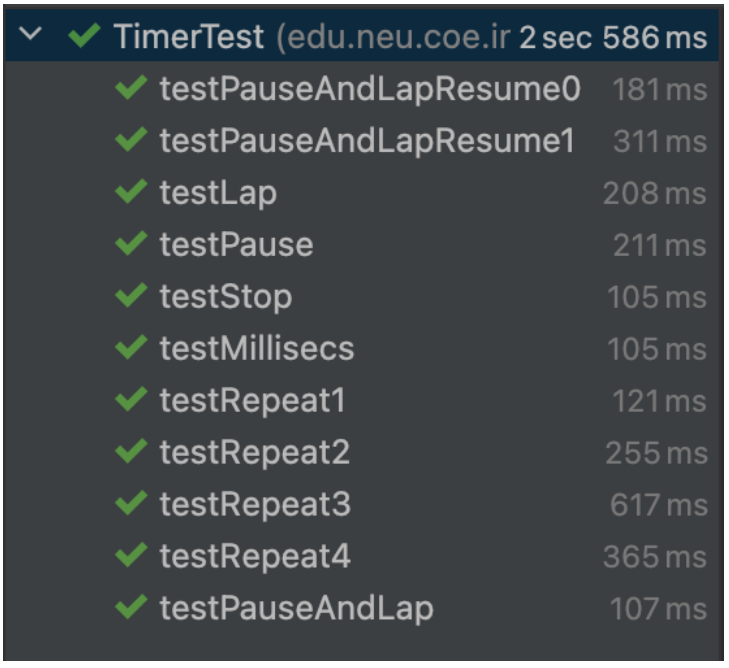
1. For an array of random elements, the graph will grow at a rate of O(n^2) which means that the growth rate of the graph will increase rapidly as the size of the array increases. The graph will form a parabolic shape.
2. For an array with elements in reverse order, the graph will grow at the same rate as for an array of random elements, which is O(n^2). The graph will also form a parabolic shape.
3. For an array which is already sorted, the graph will grow at a rate of O(n), which means that the growth rate of the graph will increase linearly as the size of the array increases. The graph will form a straight line.
4. For an array which is sorted partially, the graph will grow somewhere between the rate of O(n) and O(n^2). If the array is mostly sorted, the graph will grow closer to O(n), and if the array is less sorted, the graph will grow closer to O(n^2). The graph will form a curve somewhere between a straight line and a parabolic shape.

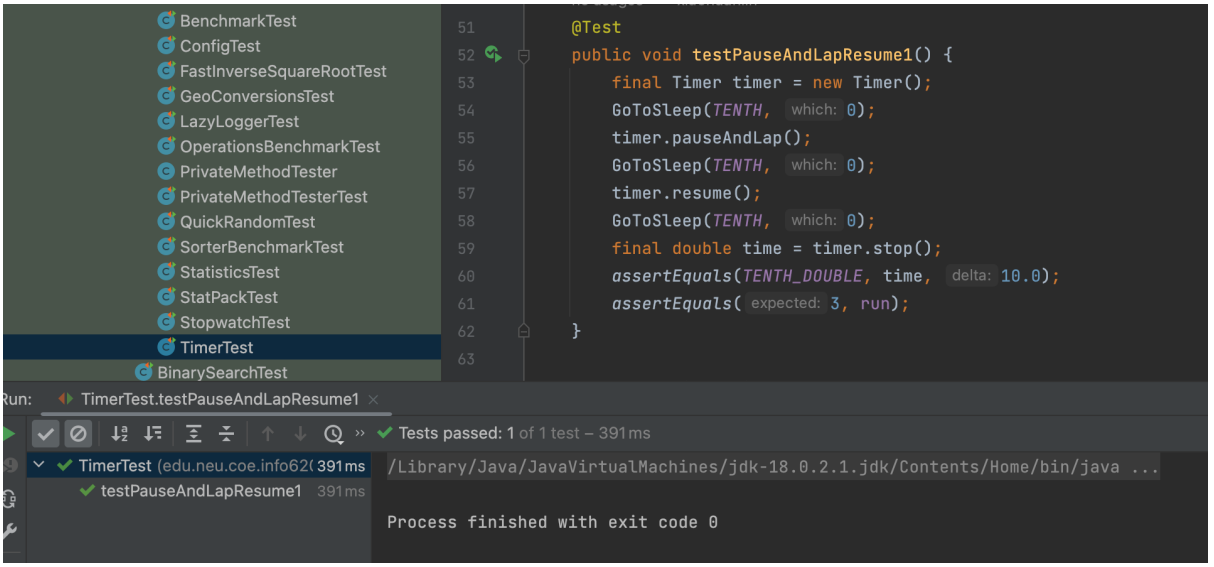
Graphical Representation:

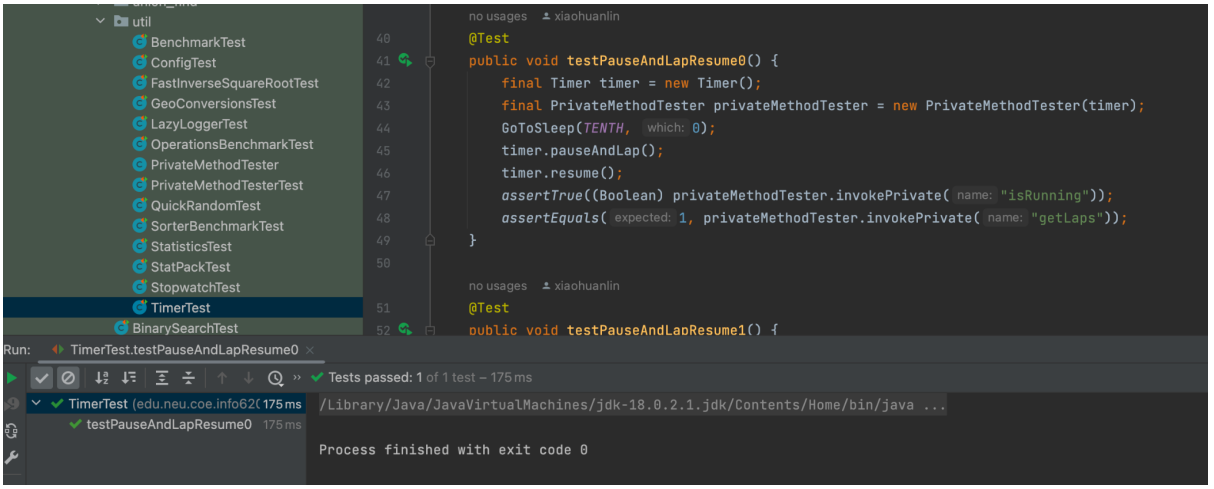


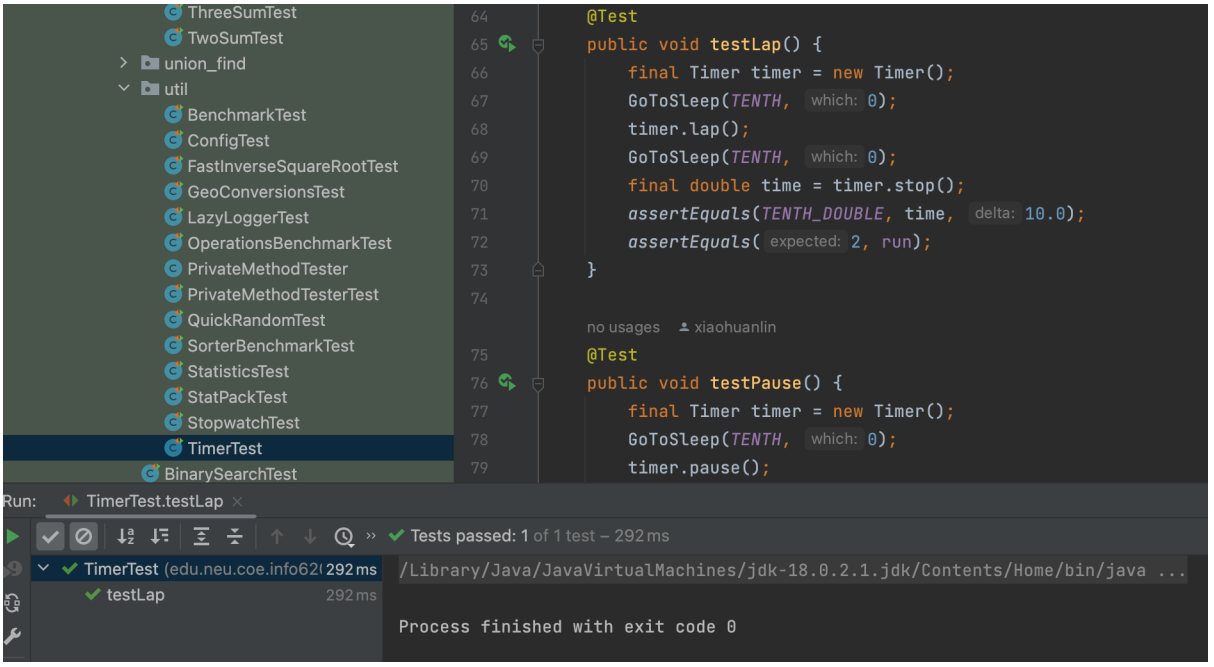
Unit Test Screenshots:

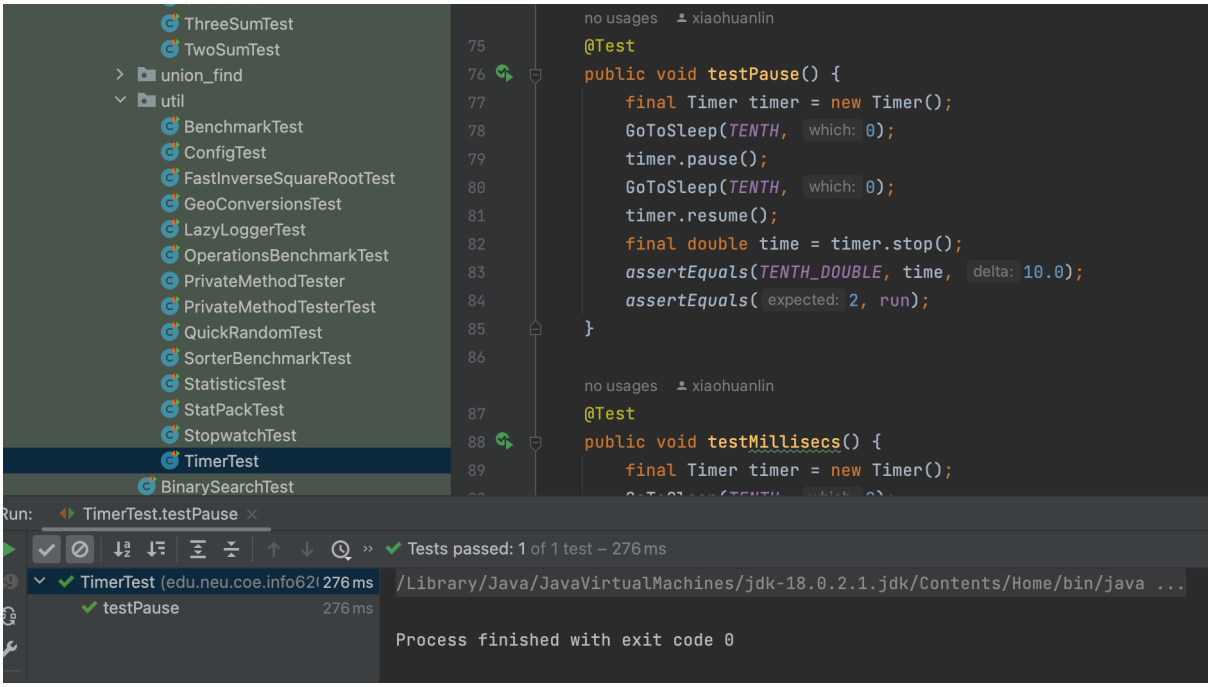
1. **Unit test cases for TimerTest**

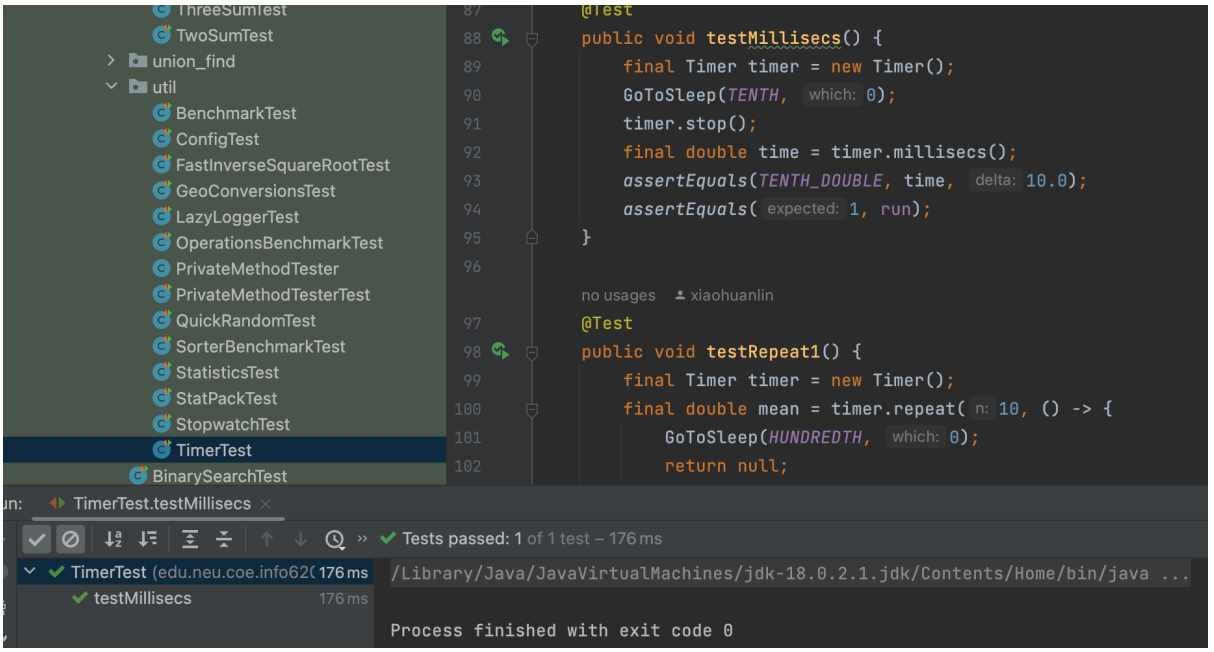


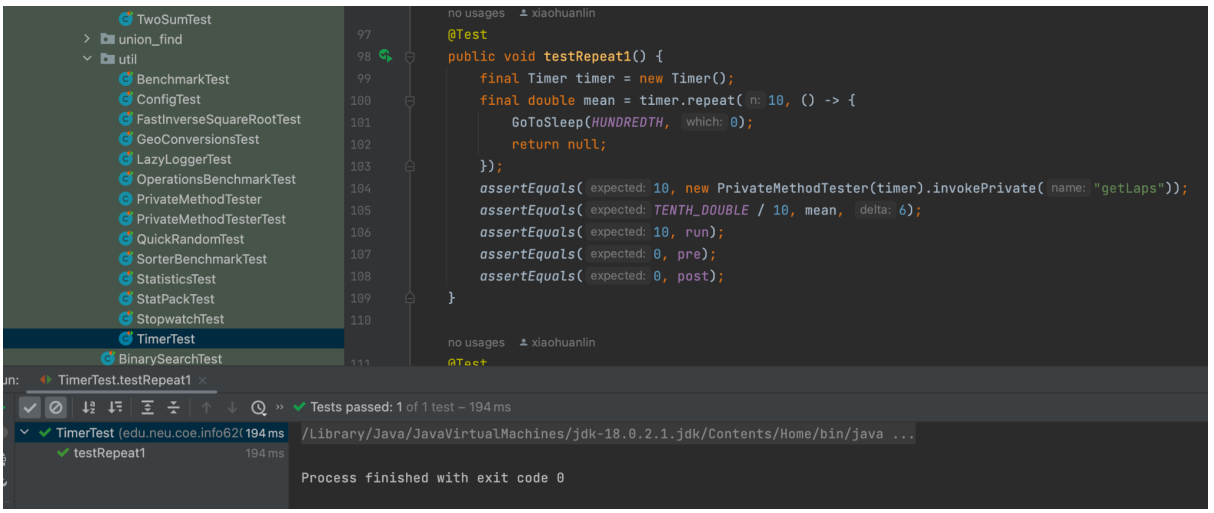

BenchmarkTest
ConfigTest
FastInverseSquareRootTest
GeoConversionsTest
LazyLoggerTest
OperationsBenchmarkTest
PrivateMethodTester
PrivateMethodTesterTest
QuickRandomTest
SorterBenchmarkTest
StatisticsTest
StatPackTest
StopwatchTest
TimerTest
BinarySearchTest
@Test
public void testPauseAndLapResume1() {
final Timer timer = new Timer();
GoToSleep(TENTH, which: 0);
timer.pauseAndLap();
GoToSleep(TENTH, which: 0);
timer.resume();
GoToSleep(TENTH, which: 0);
final double time = timer.stop();
assertEquals(TENTH_DOUBLE, time, delta: 10.0);
assertEquals(expected: 3, run);
}
Run: TimerTest.testPauseAndLapResume1
Tests passed: 1 of 1 test – 391 ms
TimerTest (edu.neu.coe.info62(391 ms
testPauseAndLapResume1 391 ms
/Library/Java/JavaVirtualMachines/jdk-18.0.2.1.jdk/Contents/Home/bin/java ...
Process finished with exit code 0

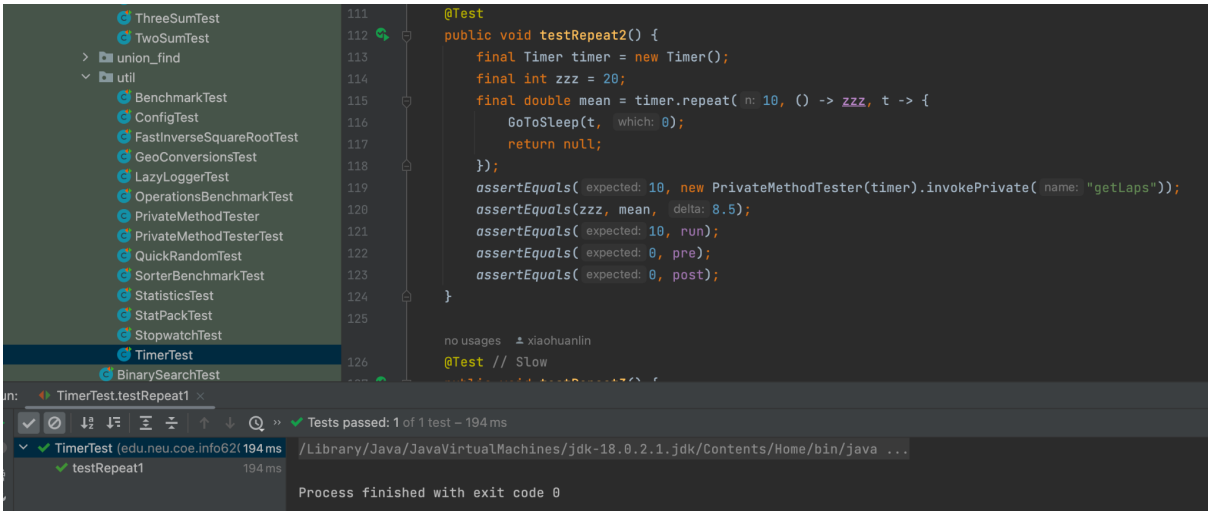

util
BenchmarkTest
ConfigTest
FastInverseSquareRootTest
GeoConversionsTest
LazyLoggerTest
OperationsBenchmarkTest
PrivateMethodTester
PrivateMethodTesterTest
QuickRandomTest
SorterBenchmarkTest
StatisticsTest
StatPackTest
StopwatchTest
TimerTest
BinarySearchTest
no usages xiaohuanlin
@Test
public void testPauseAndLapResume0() {
final Timer timer = new Timer();
final PrivateMethodTester privateMethodTester = new PrivateMethodTester(timer);
GoToSleep(TENTH, which: 0);
timer.pauseAndLap();
timer.resume();
assertTrue((Boolean) privateMethodTester.invokePrivate(name: "isRunning"));
assertEquals(expected: 1, privateMethodTester.invokePrivate(name: "getLaps"));
}
no usages xiaohuanlin
@Test
public void testPauseAndLapResume1() {
Run: TimerTest.testPauseAndLapResume0
Tests passed: 1 of 1 test – 175 ms
TimerTest (edu.neu.coe.info62(175 ms
testPauseAndLapResume0 175 ms
/Library/Java/JavaVirtualMachines/jdk-18.0.2.1.jdk/Contents/Home/bin/java ...
Process finished with exit code 0

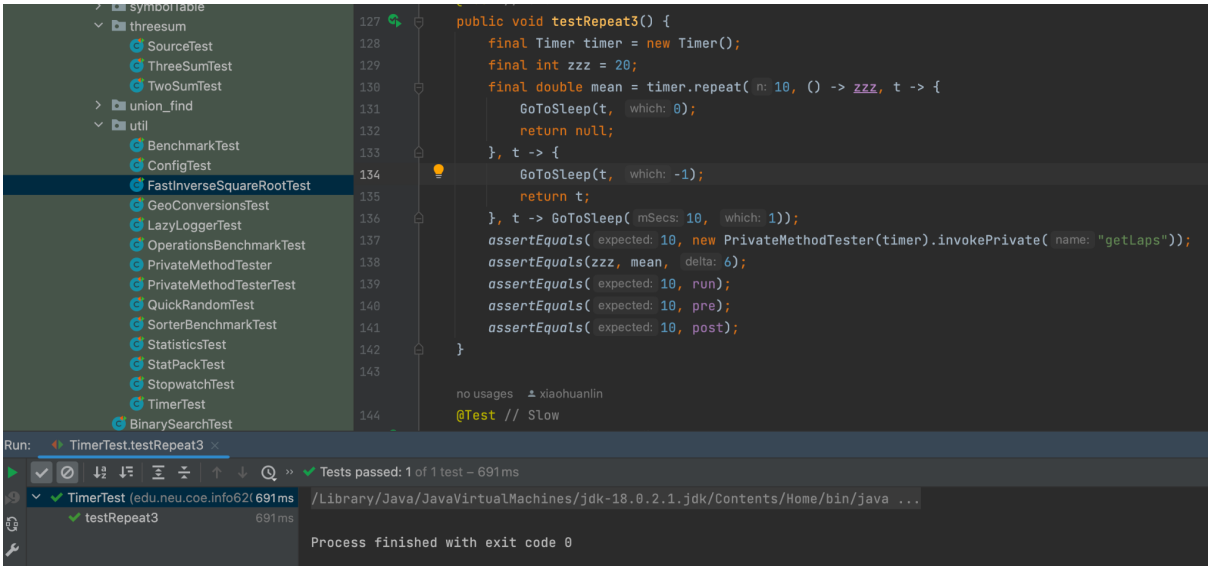

ThreeSumTest
TwoSumTest
union_find
util
BenchmarkTest
ConfigTest
FastInverseSquareRootTest
GeoConversionsTest
LazyLoggerTest
OperationsBenchmarkTest
PrivateMethodTester
PrivateMethodTesterTest
QuickRandomTest
SorterBenchmarkTest
StatisticsTest
StatPackTest
StopwatchTest
TimerTest
BinarySearchTest
@Test
public void testLap() {
final Timer timer = new Timer();
GoToSleep(TENTH, which: 0);
timer.lap();
GoToSleep(TENTH, which: 0);
final double time = timer.stop();
assertEquals(TENTH_DOUBLE, time, delta: 10.0);
assertEquals(expected: 2, run);
}
no usages xiaohuanlin
@Test
public void testPause() {
final Timer timer = new Timer();
GoToSleep(TENTH, which: 0);
timer.pause();
Run: TimerTest.testLap
Tests passed: 1 of 1 test – 292 ms
TimerTest (edu.neu.coe.info62(292 ms
testLap 292 ms
/Library/Java/JavaVirtualMachines/jdk-18.0.2.1.jdk/Contents/Home/bin/java ...
Process finished with exit code 0

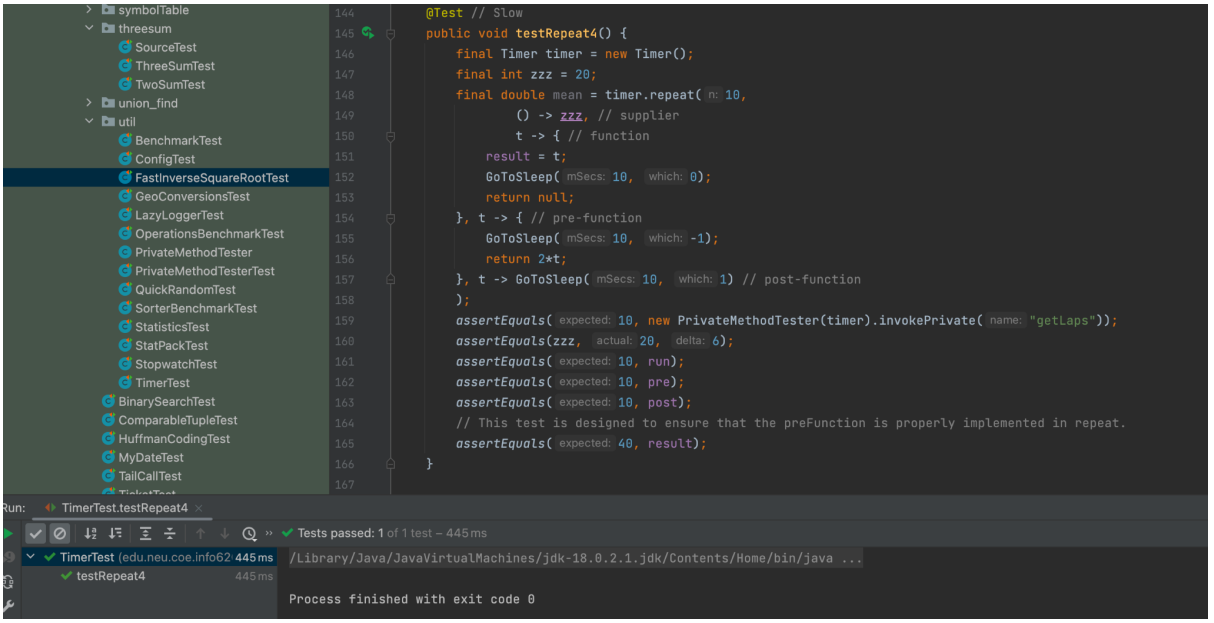

ThreeSumTest
TwoSumTest
union_find
util
BenchmarkTest
ConfigTest
FastInverseSquareRootTest
GeoConversionsTest
LazyLoggerTest
OperationsBenchmarkTest
PrivateMethodTester
PrivateMethodTesterTest
QuickRandomTest
SorterBenchmarkTest
StatisticsTest
StatPackTest
StopwatchTest
TimerTest
BinarySearchTest
no usages xiaohuanlin
@Test
public void testPause() {
final Timer timer = new Timer();
GoToSleep(TENTH, which: 0);
timer.pause();
GoToSleep(TENTH, which: 0);
timer.resume();
final double time = timer.stop();
assertEquals(TENTH_DOUBLE, time, delta: 10.0);
assertEquals(expected: 2, run);
}
no usages xiaohuanlin
@Test
public void testMillisecs() {
final Timer timer = new Timer();
Run: TimerTest.testPause
Tests passed: 1 of 1 test – 276 ms
TimerTest (edu.neu.coe.info62(276 ms
testPause 276 ms
/Library/Java/JavaVirtualMachines/jdk-18.0.2.1.jdk/Contents/Home/bin/java ...
Process finished with exit code 0

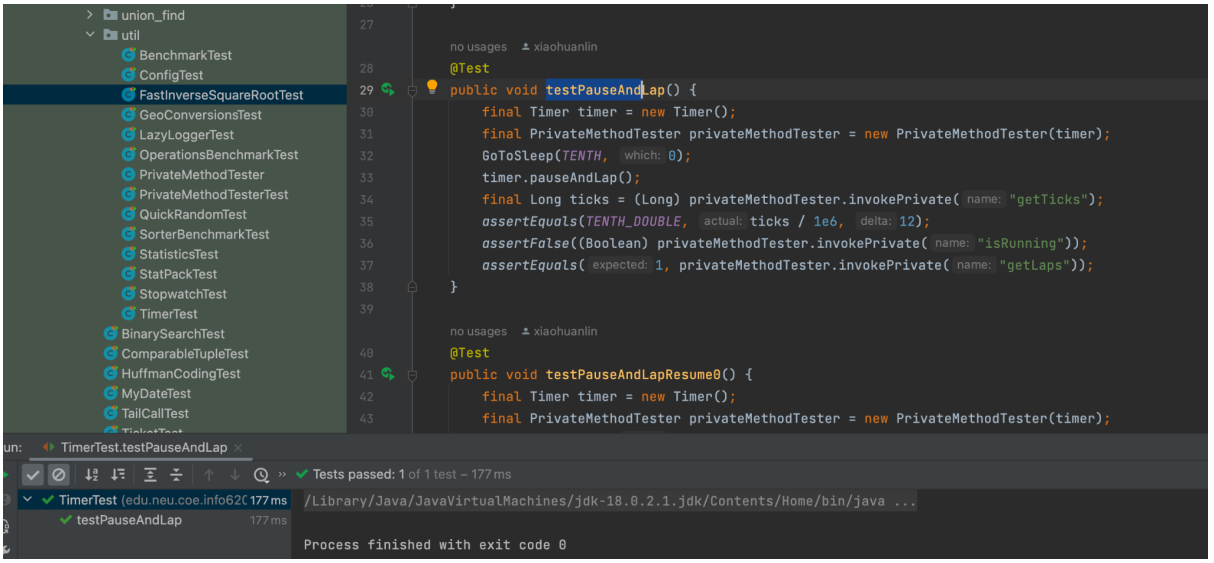

TwoSumTest
union_find
util
BenchmarkTest
ConfigTest
FastInverseSquareRootTest
GeoConversionsTest
LazyLoggerTest
OperationsBenchmarkTest
PrivateMethodTester
PrivateMethodTesterTest
QuickRandomTest
SorterBenchmarkTest
StatisticsTest
StatPackTest
StopwatchTest
TimerTest
BinarySearchTest
@Test
public void testMillisecs() {
final Timer timer = new Timer();
GoToSleep(TENTH, which: 0);
timer.stop();
final double time = timer.millisecs();
assertEquals(TENTH_DOUBLE, time, delta: 10.0);
assertEquals(expected: 1, run);
}
no usages xiaohuanlin
@Test
public void testRepeat1() {
final Timer timer = new Timer();
final double mean = timer.repeat(n: 10, () -> {
GoToSleep(HUNDREDTH, which: 0);
return null;
TimerTest.testMillisecs
Tests passed: 1 of 1 test – 176 ms
TimerTest (edu.neu.coe.info62(176 ms
testMillisecs 176 ms
/Library/Java/JavaVirtualMachines/jdk-18.0.2.1.jdk/Contents/Home/bin/java ...
Process finished with exit code 0


TwoSumTest
union_find
util
BenchmarkTest
ConfigTest
FastInverseSquareRootTest
GeoConversionsTest
LazyLoggerTest
OperationsBenchmarkTest
PrivateMethodTester
PrivateMethodTesterTest
QuickRandomTest
SorterBenchmarkTest
StatisticsTest
StatPackTest
StopwatchTest
TimerTest
BinarySearchTest
no usages xiaohuanlin
@Test
public void testRepeat1() {
final Timer timer = new Timer();
final double mean = timer.repeat(n: 10, () -> {
GoToSleep(HUNDREDTH, which: 0);
return null;
});
assertEquals(expected: 10, new PrivateMethodTester(timer).invokePrivate(name: "getLaps"));
assertEquals(expected: TENTH_DOUBLE / 10, mean, delta: 6);
assertEquals(expected: 10, run);
assertEquals(expected: 0, pre);
assertEquals(expected: 0, post);
}
no usages xiaohuanlin
@Test
Run: TimerTest.testRepeat1
Tests passed: 1 of 1 test – 194 ms
TimerTest (edu.neu.coe.info62(194 ms
testRepeat1 194 ms
/Library/Java/JavaVirtualMachines/jdk-18.0.2.1.jdk/Contents/Home/bin/java ...
Process finished with exit code 0

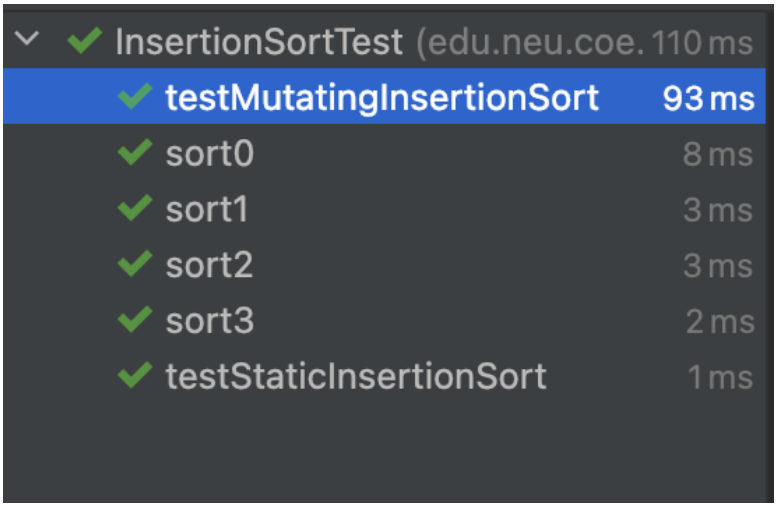

ThreeSumTest
TwoSumTest
union_find
util
BenchmarkTest
ConfigTest
FastInverseSquareRootTest
GeoConversionsTest
LazyLoggerTest
OperationsBenchmarkTest
PrivateMethodTester
PrivateMethodTesterTest
QuickRandomTest
SorterBenchmarkTest
StatisticsTest
StatPackTest
StopwatchTest
TimerTest
BinarySearchTest
@Test
public void testRepeat2() {
final Timer timer = new Timer();
final int zzz = 20;
final double mean = timer.repeat(n: 10, () -> zzz, t -> {
GoToSleep(t, which: 0);
return null;
});
assertEquals(expected: 10, new PrivateMethodTester(timer).invokePrivate(name: "getLaps"));
assertEquals(zzz, mean, delta: 8.5);
assertEquals(expected: 10, run);
assertEquals(expected: 0, pre);
assertEquals(expected: 0, post);
}
no usages xiaohuanlin
@Test // Slow
Run: TimerTest.testRepeat1
Tests passed: 1 of 1 test – 194 ms
TimerTest (edu.neu.coe.info62(194 ms
testRepeat1 194 ms
/Library/Java/JavaVirtualMachines/jdk-18.0.2.1.jdk/Contents/Home/bin/java ...
Process finished with exit code 0

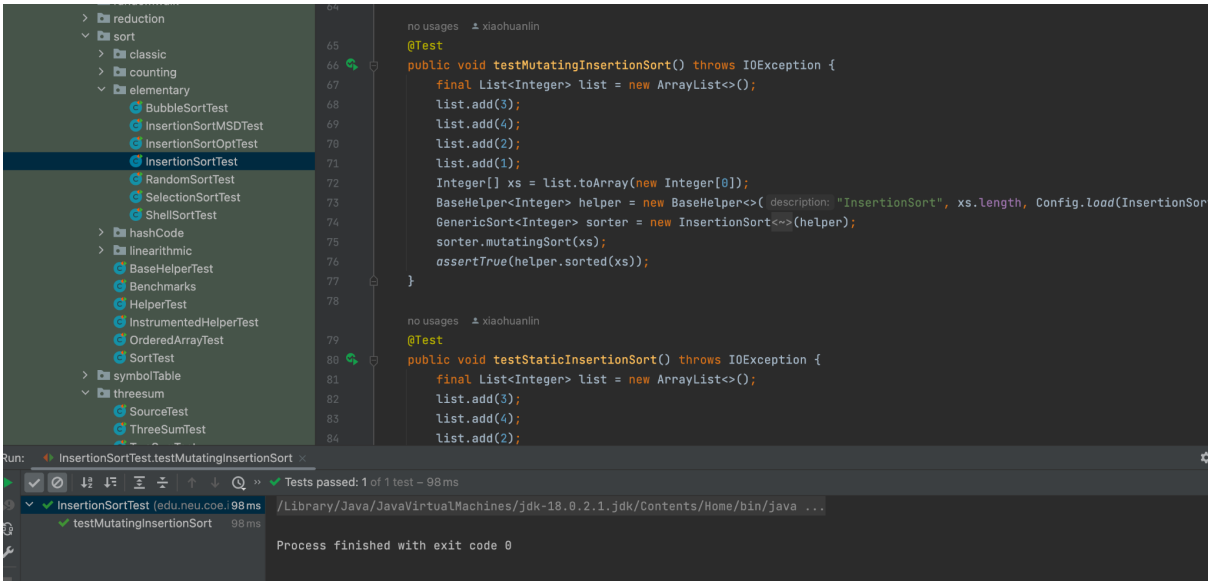

threesum
SourceTest
ThreeSumTest
TwoSumTest
union_find
util
BenchmarkTest
ConfigTest
FastInverseSquareRootTest
GeoConversionsTest
LazyLoggerTest
OperationsBenchmarkTest
PrivateMethodTester
PrivateMethodTesterTest
QuickRandomTest
SorterBenchmarkTest
StatisticsTest
StatPackTest
StopwatchTest
TimerTest
BinarySearchTest
public void testRepeat3() {
final Timer timer = new Timer();
final int zzz = 20;
final double mean = timer.repeat(n: 10, () -> 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);
}
no usages xiaohuanlin
@Test // Slow
Run: TimerTest.testRepeat3
Tests passed: 1 of 1 test – 691 ms
TimerTest (edu.neu.coe.info62(691 ms
testRepeat3 691 ms
/Library/Java/JavaVirtualMachines/jdk-18.0.2.1.jdk/Contents/Home/bin/java ...
Process finished with exit code 0

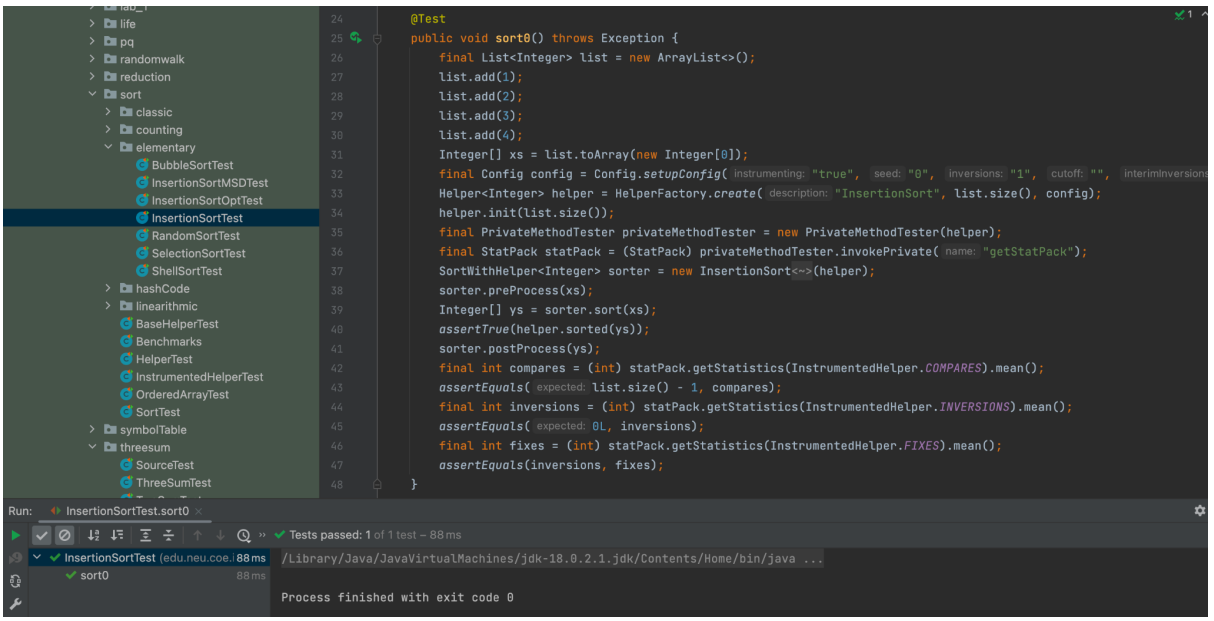


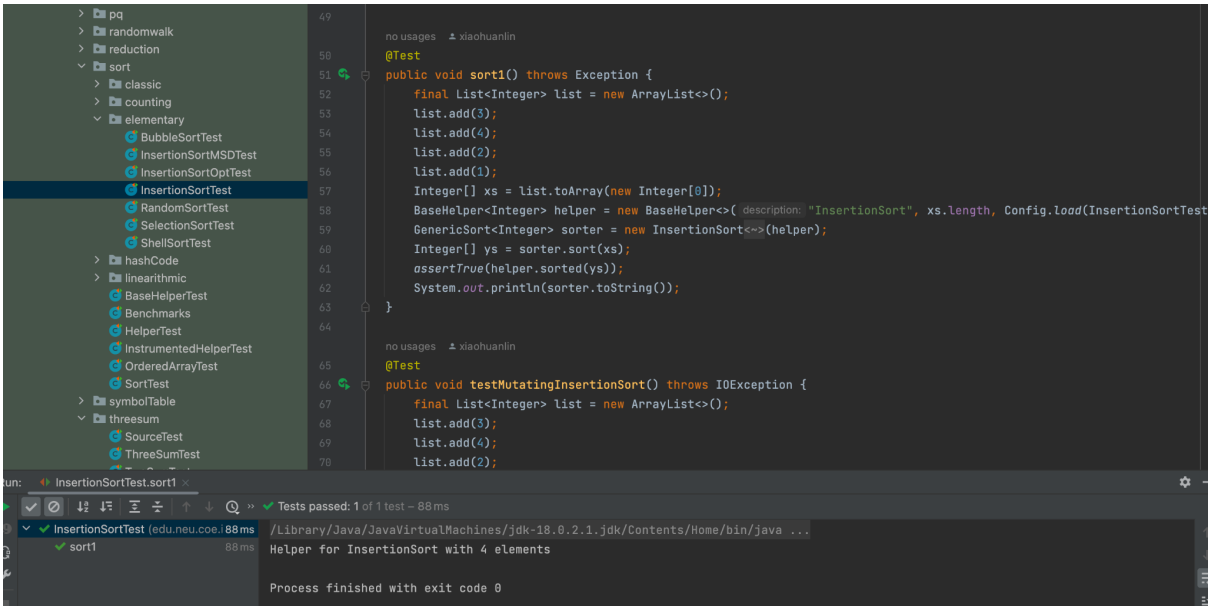


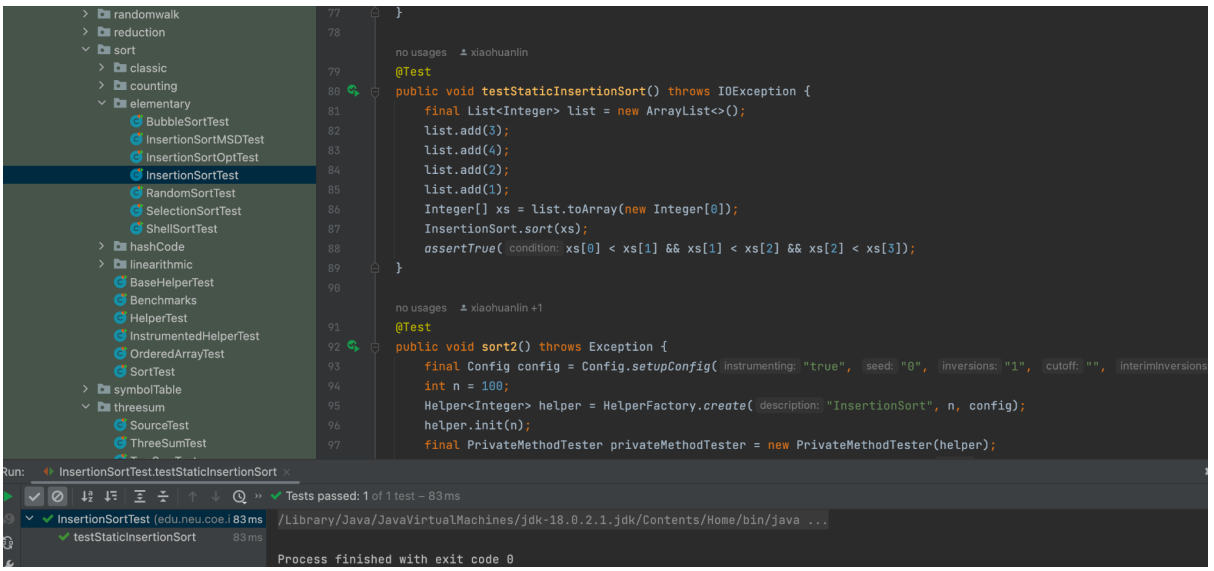
2. Unit tests for InsertionSort

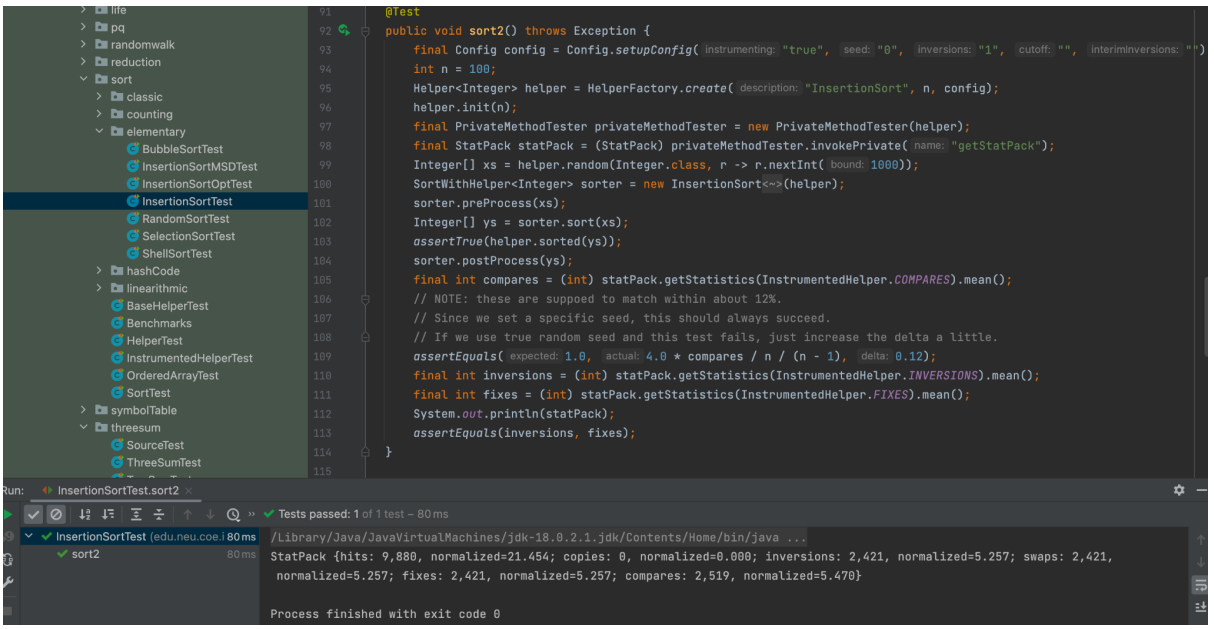


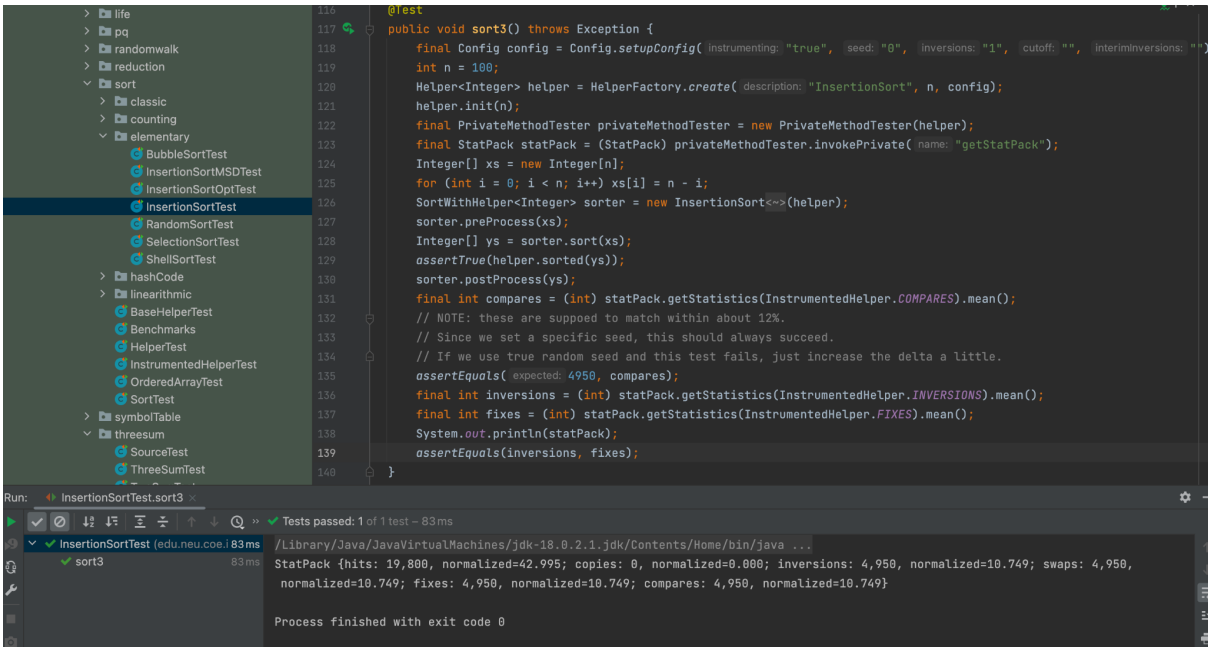

reduction
sort
classic
counting
elementary
BubbleSortTest
InsertionSortMSDTest
InsertionSortOptTest
InsertionSortTest
RandomSortTest
SelectionSortTest
ShellSortTest
hashCode
linearithmic
BaseHelperTest
Benchmarks
HelperTest
InstrumentedHelperTest
OrderedArrayTest
SortTest
symbolTable
threesum
SourceTest
ThreeSumTest
no usages xiaohuanlin
@Test
public void testMutatingInsertionSort() throws IOException {
final List<Integer> list = new ArrayList<>();
list.add(3);
list.add(4);
list.add(2);
list.add(1);
Integer[] xs = list.toArray(new Integer[0]);
BaseHelper<Integer> helper = new BaseHelper<>(description: "InsertionSort", xs.length, Config.load(InsertionSor
GenericSort<Integer> sorter = new InsertionSort<~>(helper);
sorter.mutatingSort(xs);
assertTrue(helper.sorted(xs));
}
no usages xiaohuanlin
@Test
public void testStaticInsertionSort() throws IOException {
final List<Integer> list = new ArrayList<>();
list.add(3);
list.add(4);
list.add(2);
Run: InsertionSortTest.testMutatingInsertionSort
Tests passed: 1 of 1 test – 98 ms
InsertionSortTest (edu.neu.coe.i 98 ms
testMutatingInsertionSort 98 ms
/Library/Java/JavaVirtualMachines/jdk-18.0.2.1.jdk/Contents/Home/bin/java ...
Process finished with exit code 0


life
pq
randomwalk
reduction
sort
classic
counting
elementary
BubbleSortTest
InsertionSortMSDTest
InsertionSortOptTest
InsertionSortTest
RandomSortTest
SelectionSortTest
ShellSortTest
hashCode
linearithmic
BaseHelperTest
Benchmarks
HelperTest
InstrumentedHelperTest
OrderedArrayTest
SortTest
symbolTable
threesum
SourceTest
ThreeSumTest
@Test
public void sort0() throws Exception {
final List<Integer> list = new ArrayList<>();
list.add(1);
list.add(2);
list.add(3);
list.add(4);
Integer[] xs = list.toArray(new Integer[0]);
final Config config = Config.setupConfig(instrumenting: "true", seed: "0", inversions: "1", cutoff: "", interiminversions
Helper<Integer> helper = HelperFactory.create(description: "InsertionSort", list.size(), config);
helper.init(list.size());
final PrivateMethodTester privateMethodTester = new PrivateMethodTester(helper);
final StatPack statPack = (StatPack) privateMethodTester.invokePrivate(name: "getStatPack");
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();
assertEquals(expected: list.size() - 1, compares);
final int inversions = (int) statPack.getStatistics(InstrumentedHelper.INVERSIONS).mean();
assertEquals(expected: 0L, inversions);
final int fixes = (int) statPack.getStatistics(InstrumentedHelper.FIXES).mean();
assertEquals(inversions, fixes);
}
Run: InsertionSortTest.sort0
Tests passed: 1 of 1 test – 88 ms
InsertionSortTest (edu.neu.coe.i 88 ms
sort0 88 ms
/Library/Java/JavaVirtualMachines/jdk-18.0.2.1.jdk/Contents/Home/bin/java ...
Process finished with exit code 0

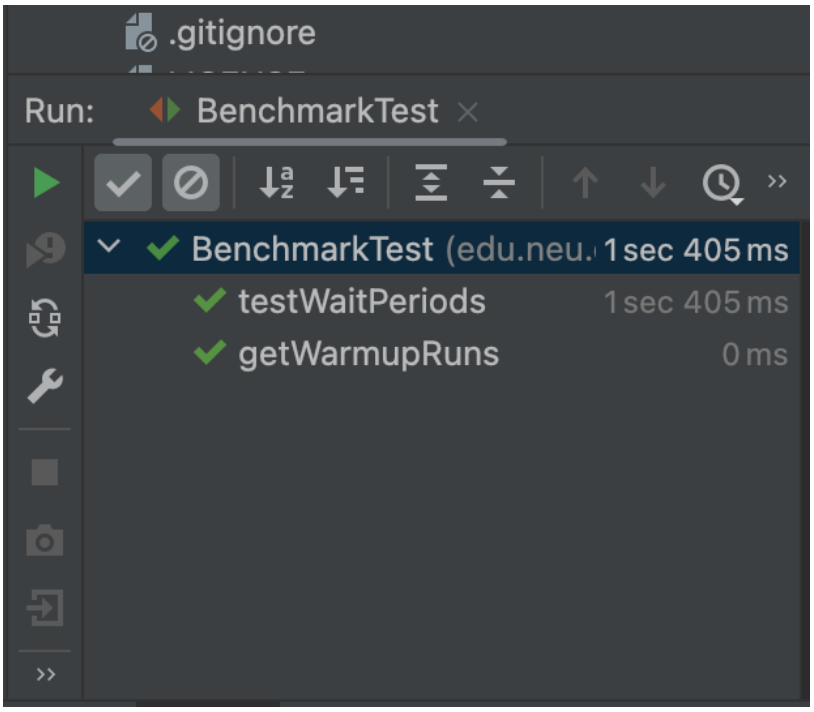

pq
randomwalk
reduction
sort
classic
counting
elementary
BubbleSortTest
InsertionSortMSDTest
InsertionSortOptTest
InsertionSortTest
RandomSortTest
SelectionSortTest
ShellSortTest
hashCode
linearithmic
BaseHelperTest
Benchmarks
HelperTest
InstrumentedHelperTest
OrderedArrayTest
SortTest
symbolTable
threesum
SourceTest
ThreeSumTest
no usages xiaohuanlin
@Test
public void sort1() throws Exception {
final List<Integer> list = new ArrayList<>();
list.add(3);
list.add(4);
list.add(2);
list.add(1);
Integer[] xs = list.toArray(new Integer[0]);
BaseHelper<Integer> helper = new BaseHelper<>(description: "InsertionSort", xs.length, Config.load(InsertionSortTest
GenericSort<Integer> sorter = new InsertionSort<~>(helper);
Integer[] ys = sorter.sort(xs);
assertTrue(helper.sorted(ys));
System.out.println(sorter.toString());
}
no usages xiaohuanlin
@Test
public void testMutatingInsertionSort() throws IOException {
final List<Integer> list = new ArrayList<>();
list.add(3);
list.add(4);
list.add(2);
Run: InsertionSortTest.sort1
Tests passed: 1 of 1 test – 88 ms
InsertionSortTest (edu.neu.coe.i 88 ms
sort1 88 ms
/Library/Java/JavaVirtualMachines/jdk-18.0.2.1.jdk/Contents/Home/bin/java ...
Helper for InsertionSort with 4 elements
Process finished with exit code 0

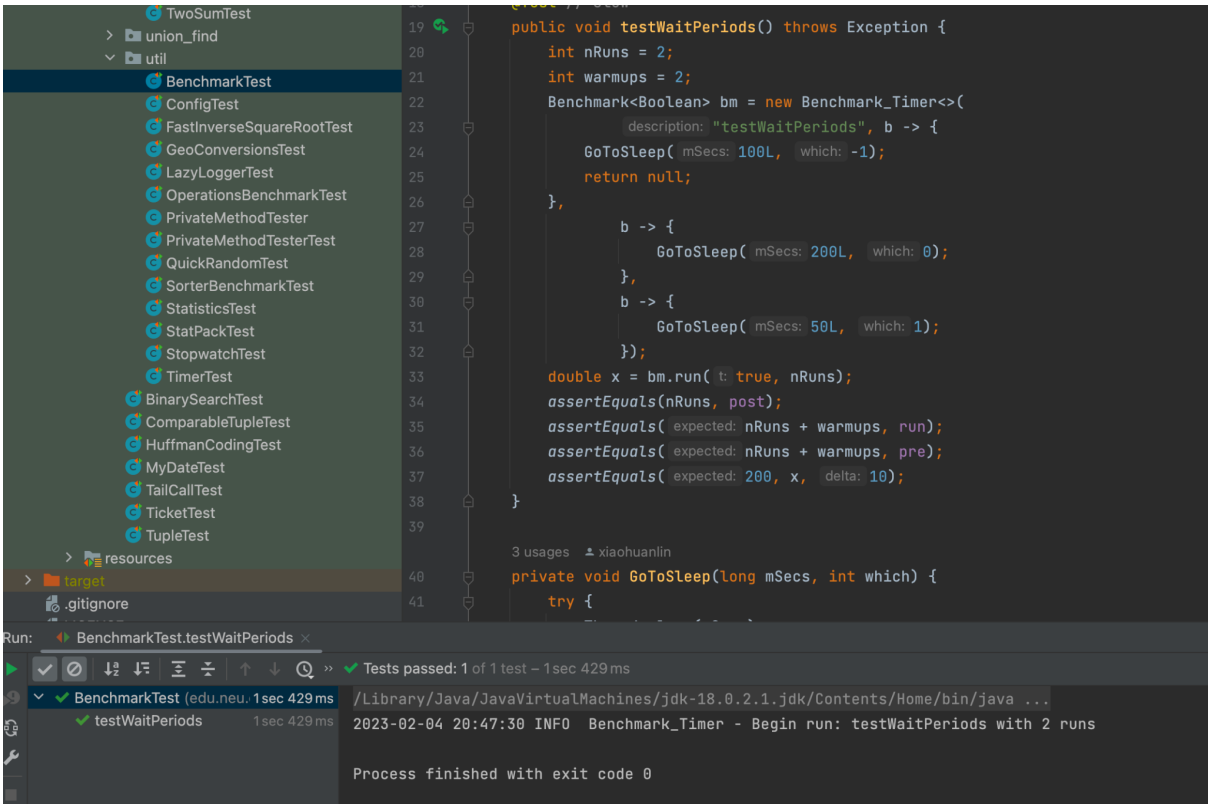

randomwalk
reduction
sort
classic
counting
elementary
BubbleSortTest
InsertionSortMSDTest
InsertionSortOptTest
InsertionSortTest
RandomSortTest
SelectionSortTest
ShellSortTest
hashCode
linearithmic
BaseHelperTest
Benchmarks
HelperTest
InstrumentedHelperTest
OrderedArrayTest
SortTest
symbolTable
threesum
SourceTest
ThreeSumTest
}
no usages xiaohuanlin
@Test
public void testStaticInsertionSort() throws IOException {
final List<Integer> list = new ArrayList<>();
list.add(3);
list.add(4);
list.add(2);
list.add(1);
Integer[] xs = list.toArray(new Integer[0]);
InsertionSort.sort(xs);
assertTrue(condition: xs[0] < xs[1] && xs[1] < xs[2] && xs[2] < xs[3]);
}
no usages xiaohuanlin +1
@Test
public void sort2() 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);
Run: InsertionSortTest.testStaticInsertionSort
Tests passed: 1 of 1 test – 83 ms
InsertionSortTest (edu.neu.coe.i 83 ms
testStaticInsertionSort 83 ms
/Library/Java/JavaVirtualMachines/jdk-18.0.2.1.jdk/Contents/Home/bin/java ...
Process finished with exit code 0

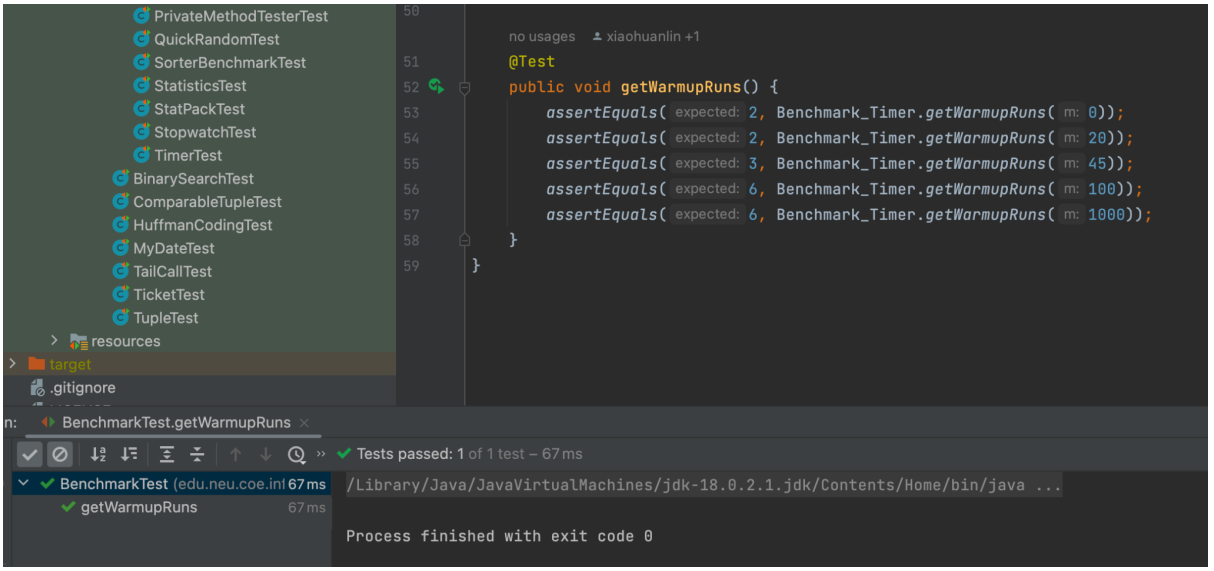

life
pq
randomwalk
reduction
sort
classic
counting
elementary
BubbleSortTest
InsertionSortMSDTest
InsertionSortOptTest
InsertionSortTest
RandomSortTest
SelectionSortTest
ShellSortTest
hashCode
linearithmic
BaseHelperTest
Benchmarks
HelperTest
InstrumentedHelperTest
OrderedArrayTest
SortTest
symbolTable
threesum
SourceTest
ThreeSumTest
@Test
public void sort2() 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 = helper.random(Integer.class, r -> r.nextInt(bound: 1000));
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 supposed 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.
assertEquals(expected: 1.0, actual: 4.0 * compares / n / (n - 1), delta: 0.12);
final int inversions = (int) statPack.getStatistics(InstrumentedHelper.INVERSIONS).mean();
final int fixes = (int) statPack.getStatistics(InstrumentedHelper.FIXES).mean();
System.out.println(statPack);
assertEquals(inversions, fixes);
}
Run: InsertionSortTest.sort2
Tests passed: 1 of 1 test – 80 ms
InsertionSortTest (edu.neu.coe.i 80 ms
sort2 80 ms
/Library/Java/JavaVirtualMachines/jdk-18.0.2.1.jdk/Contents/Home/bin/java ...
StatPack {hits: 9,880, normalized=21.454; copies: 0, normalized=0.000; inversions: 2,421, normalized=5.257; swaps: 2,421,
normalized=5.257; fixes: 2,421, normalized=5.257; compares: 2,519, normalized=5.470}
Process finished with exit code 0


life
pq
randomwalk
reduction
sort
classic
counting
elementary
BubbleSortTest
InsertionSortMSDTest
InsertionSortOptTest
InsertionSortTest
RandomSortTest
SelectionSortTest
ShellSortTest
hashCode
linearithmic
BaseHelperTest
Benchmarks
HelperTest
InstrumentedHelperTest
OrderedArrayTest
SortTest
symbolTable
threesum
SourceTest
ThreeSumTest
@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 supposed 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.
assertEquals(expected: 4950, compares);
final int inversions = (int) statPack.getStatistics(InstrumentedHelper.INVERSIONS).mean();
final int fixes = (int) statPack.getStatistics(InstrumentedHelper.FIXES).mean();
System.out.println(statPack);
assertEquals(inversions, fixes);
}
Run: InsertionSortTest.sort3
Tests passed: 1 of 1 test – 83 ms
InsertionSortTest (edu.neu.coe.i 83 ms
sort3 83 ms
/Library/Java/JavaVirtualMachines/jdk-18.0.2.1.jdk/Contents/Home/bin/java ...
StatPack {hits: 19,800, normalized=42.995; copies: 0, normalized=0.000; inversions: 4,950, normalized=10.749; swaps: 4,950,
normalized=10.749; fixes: 4,950, normalized=10.749; compares: 4,950, normalized=10.749}
Process finished with exit code 0

3. Unit tests for BenchmarkTest






PrivateMethodTesterTest
QuickRandomTest
SorterBenchmarkTest
StatisticsTest
StatPackTest
StopwatchTest
TimerTest
BinarySearchTest
ComparableTupleTest
HuffmanCodingTest
MyDateTest
TailCallTest
TicketTest
TupleTest
resources
target
.gitignore
no usages xiaohuanlin +1
@Test
public void getWarmupRuns() {
assertEquals(expected: 2, Benchmark_Timer.getWarmupRuns(m: 0));
assertEquals(expected: 2, Benchmark_Timer.getWarmupRuns(m: 20));
assertEquals(expected: 3, Benchmark_Timer.getWarmupRuns(m: 45));
assertEquals(expected: 6, Benchmark_Timer.getWarmupRuns(m: 100));
assertEquals(expected: 6, Benchmark_Timer.getWarmupRuns(m: 1000));
}
}
BenchmarkTest.getWarmupRuns
Tests passed: 1 of 1 test – 67 ms
BenchmarkTest (edu.neu.coe.inf 67 ms
getWarmupRuns 67 ms
/Library/Java/JavaVirtualMachines/jdk-18.0.2.1.jdk/Contents/Home/bin/java ...
Process finished with exit code 0