Parvataneni Mohith Lalita Kumar

INFO 6205 - Fall 2021

Assignment - 2

Task:

To implement Timer Class and verify BenchmarkTest and TimerTest.

Implement Insertion Sort and run Unit Texts using InsertionSort Test.

Implement a program to run benchmarks for Random, Ordered, Reverse Sorted, Partial Sorted arrays.

Findings:

2021-09-27 00:24:42 INFO Benchmark_Timer - Begin run: Insertion sort for partially sorted array of size: 1000 with 5 runs
0.0 ms
2021-09-27 00:24:42 INFO Benchmark_Timer - Begin run: Insertion sort for randomly sorted array of size: 1000 with 5 runs
0.0 ms
2021-09-27 00:24:42 INFO Benchmark_Timer - Begin run: Insertion sort for a sorted array of size: 1000 with 5 runs
0.0 ms
2021-09-27 00:24:42 INFO Benchmark_Timer - Begin run: Insertion sort for reverse sorted array of size: 1000 with 5 runs
0.2 ms

2021-09-27 00:25:11 INFO Benchmark_Timer - Begin run: Insertion sort for partially sorted array of size: 2000 with 5 runs
0.4 ms
2021-09-27 00:25:11 INFO Benchmark_Timer - Begin run: Insertion sort for randomly sorted array of size: 2000 with 5 runs
0.2 ms
2021-09-27 00:25:11 INFO Benchmark_Timer - Begin run: Insertion sort for a sorted array of size: 2000 with 5 runs

0.0 ms
2021-09-27 00:25:11 INFO Benchmark_Timer - Begin run: Insertion sort for reverse sorted array of size: 2000 with 5 runs
1.0 ms

2021-09-27 00:25:43 INFO Benchmark_Timer - Begin run: Insertion sort for partially sorted array of size: 4000 with 5 runs
0.4 ms
2021-09-27 00:25:43 INFO Benchmark_Timer - Begin run: Insertion sort for randomly sorted array of size: 4000 with 5 runs
0.4 ms
2021-09-27 00:25:43 INFO Benchmark_Timer - Begin run: Insertion sort for a sorted array of size: 4000 with 5 runs
0.2 ms
2021-09-27 00:25:43 INFO Benchmark_Timer - Begin run: Insertion sort for reverse sorted array of size: 4000 with 5 runs
1.2 ms

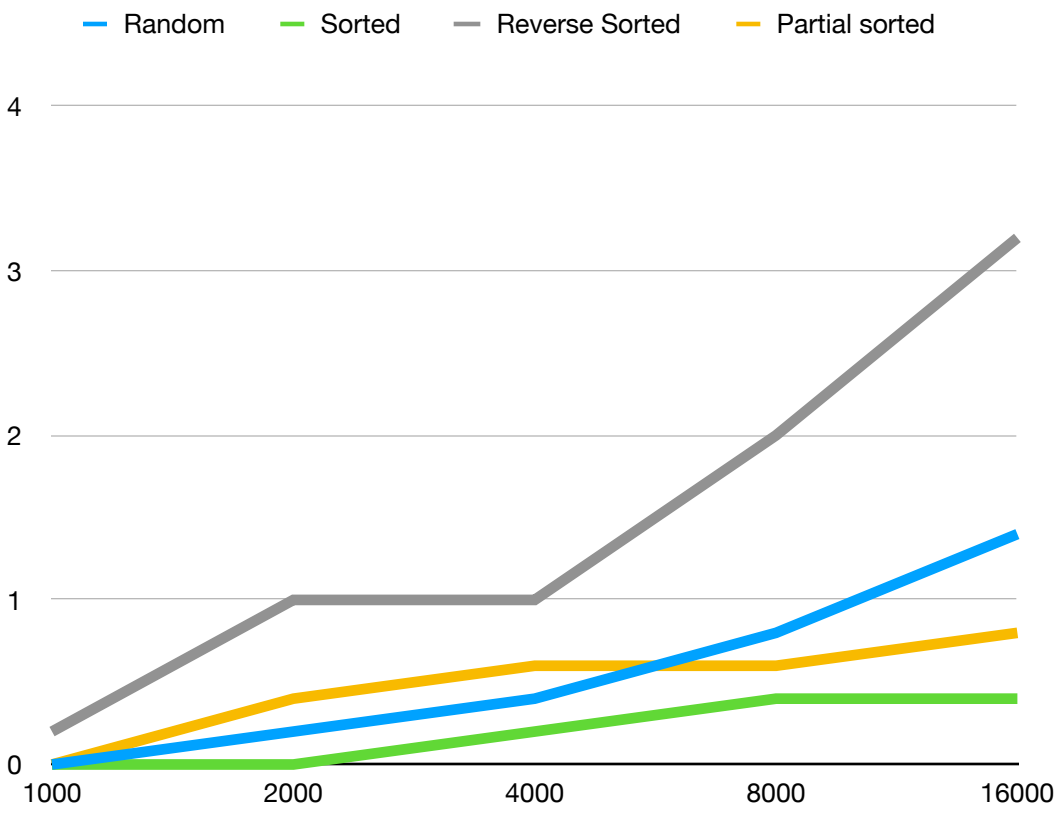
2021-09-27 00:28:33 INFO Benchmark_Timer - Begin run: Insertion sort for partially sorted array of size: 8000 with 5 runs
0.6 ms
2021-09-27 00:28:33 INFO Benchmark_Timer - Begin run: Insertion sort for randomly sorted array of size: 8000 with 5 runs
0.8 ms
2021-09-27 00:28:33 INFO Benchmark_Timer - Begin run: Insertion sort for a sorted array of size: 8000 with 5 runs
0.4 ms
2021-09-27 00:28:33 INFO Benchmark_Timer - Begin run: Insertion sort for reverse sorted array of size: 8000 with 5 runs
2.0 ms

2021-09-27 00:29:02 INFO Benchmark_Timer - Begin run: Insertion sort for partially sorted array of size: 16000 with 5 runs
0.8 ms
2021-09-27 00:29:02 INFO Benchmark_Timer - Begin run: Insertion sort for randomly sorted array of size: 16000 with 5 runs
1.4 ms
2021-09-27 00:29:02 INFO Benchmark_Timer - Begin run: Insertion sort for a sorted array of size: 16000 with 5 runs
0.4 ms

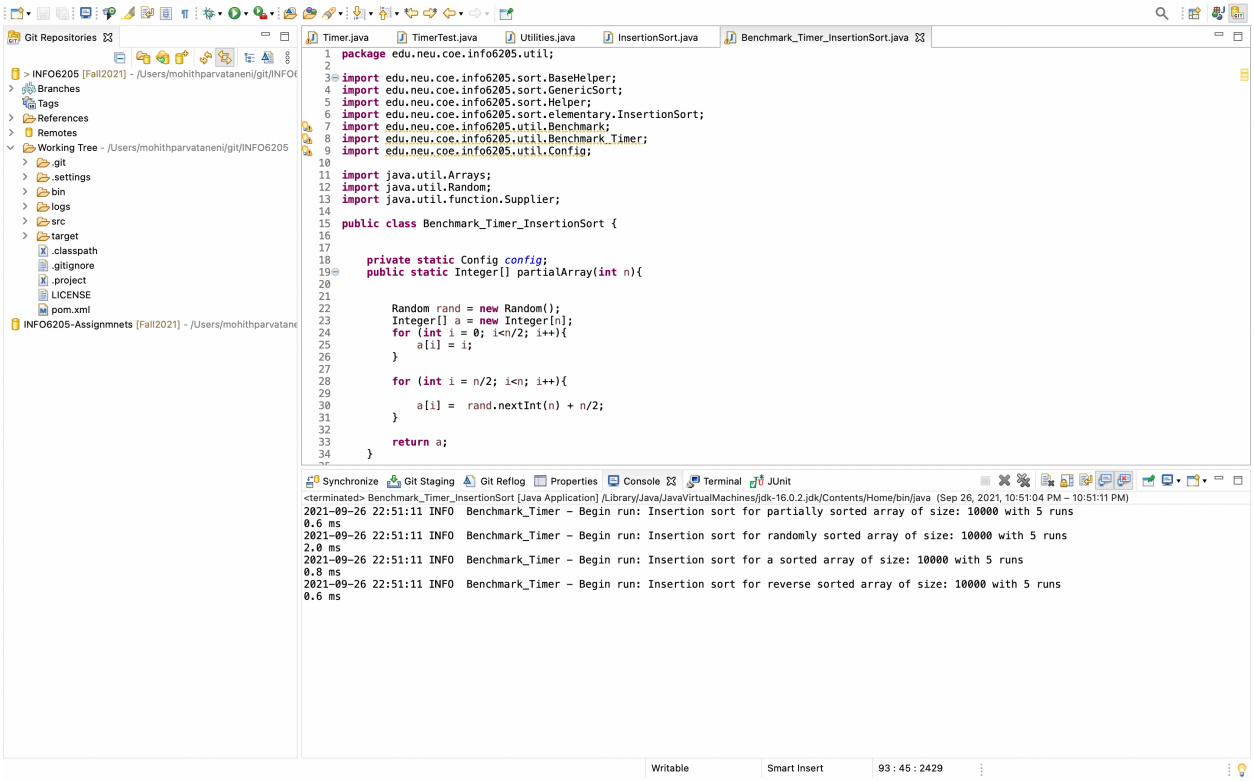
2021-09-27 00:29:02 INFO Benchmark_Timer - Begin run: Insertion sort for reverse sorted array of size: 16000 with 5 runs
3.2 ms

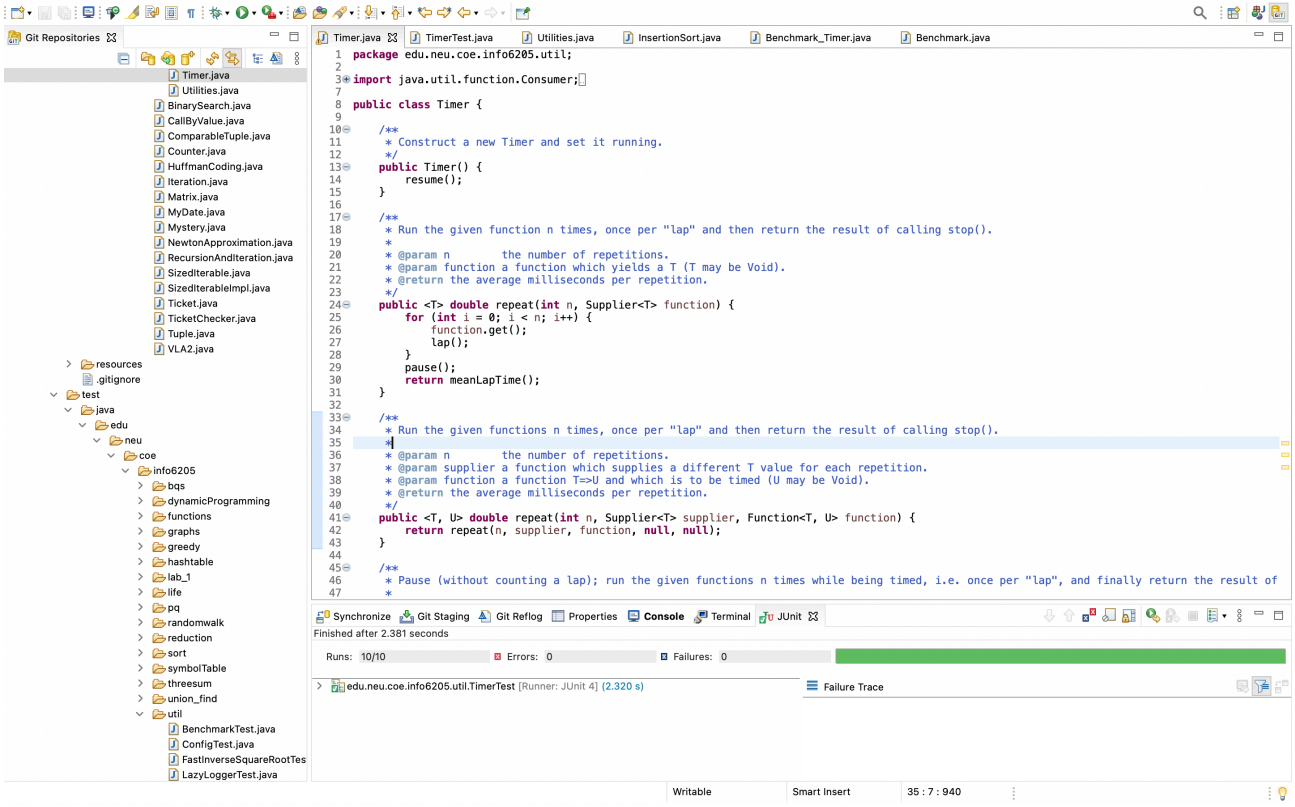
N	Random	Sorted	Reverse Sorted	Partial Sorted
1000	0	0	0.2	0
2000	0.2	0	1	0.4
4000	0.4	0.2	1	0.4
8000	0.8	0.4	2	0.6
16000	1.4	0.4	3.2	0.8

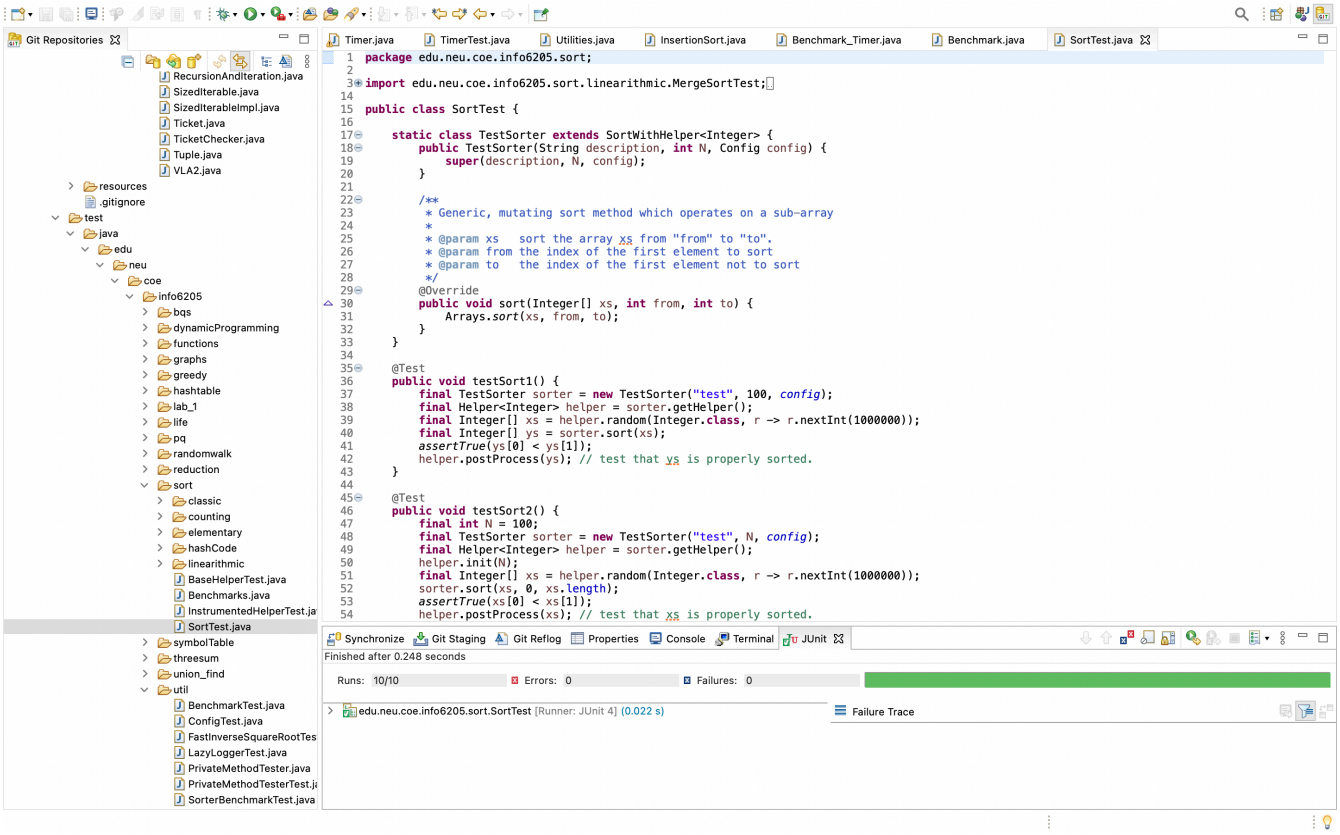
Graph:



Output /Testing Screenshot :






Git Repositories
RecursionAndIteration.java
SizedIterable.java
SizedIterableImpl.java
Ticket.java
TicketChecker.java
Tuple.java
VLA2.java
resources
.gitignore
test
java
edu
neu
coe
info6205
bqs
dynamicProgramming
functions
graphs
greedy
hashtable
lab_1
life
pq
randomwalk
reduction
sort
classic
counting
elementary
hashCode
linearithmic
BaseHelperTest.java
Benchmarks.java
InstrumentedHelperTest.ja
SortTest.java
symbolTable
threesum
union_find
util
BenchmarkTest.java
ConfigTest.java
FastInverseSquareRootTes
LazyLoggerTest.java
PrivateMethodTester.java
PrivateMethodTesterTest.ja
SorterBenchmarkTest.java
Timer.java
TimerTest.java
Utilities.java
InsertionSort.java
Benchmark_Timer.java
Benchmark.java
SortTest.java
1 package edu.neu.coe.info6205.sort;
2
3 import edu.neu.coe.info6205.sort.linearithmic.MergeSortTest;
14
15 public class SortTest {
16
17 static class TestSorter extends SortWithHelper<Integer> {
18 public TestSorter(String description, int N, Config config) {
19 super(description, N, config);
20 }
21
22 /**
23 * Generic, mutating sort method which operates on a sub-array
24 *
25 * @param xs sort the array xs from "from" to "to".
26 * @param from the index of the first element to sort
27 * @param to the index of the first element not to sort
28 */
29 @Override
30 public void sort(Integer[] xs, int from, int to) {
31 Arrays.sort(xs, from, to);
32 }
33 }
34
35 @Test
36 public void testSort1() {
37 final TestSorter sorter = new TestSorter("test", 100, config);
38 final Helper<Integer> helper = sorter.getHelper();
39 final Integer[] xs = helper.random(Integer.class, r -> r.nextInt(1000000));
40 final Integer[] ys = sorter.sort(xs);
41 assertTrue(ys[0] < ys[1]);
42 helper.postProcess(ys); // test that ys is properly sorted.
43 }
44
45 @Test
46 public void testSort2() {
47 final int N = 100;
48 final TestSorter sorter = new TestSorter("test", N, config);
49 final Helper<Integer> helper = sorter.getHelper();
50 helper.init(N);
51 final Integer[] xs = helper.random(Integer.class, r -> r.nextInt(1000000));
52 sorter.sort(xs, 0, xs.length);
53 assertTrue(xs[0] < xs[1]);
54 helper.postProcess(xs); // test that xs is properly sorted.
Synchronize
Git Staging
Git Reflog
Properties
Console
Terminal
JUnit
Finished after 0.248 seconds
Runs: 10/10
Errors: 0
Failures: 0
edu.neu.coe.info6205.sort.SortTest [Runner: JUnit 4] (0.022 s)
Failure Trace