Program Structures and Algorithms
Spring 2024

NAME: Ting Guo
NUID: 002834835
GITHUB LINK:https://github.com/Nangongnuanshan/INFO6205

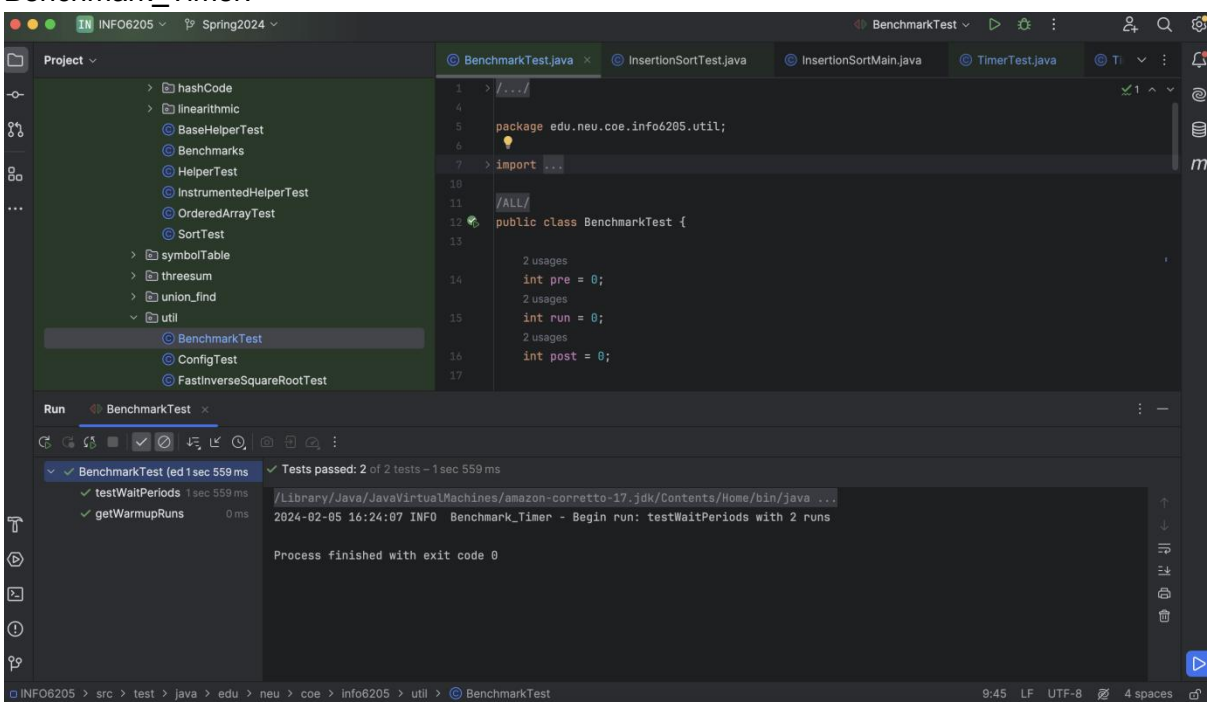
Task:Assignment 3

Explantion:
Part1: Because public Timer(){resume();} , we need getClock() at the begin of repeat to reset the ticks when calculating the finial ticks. Just time function, so we start getClock() before function.apply(t) and end getClock() after that. After pause(), to reset ticks, we need ticks -= getClock() to make sure we just time function.

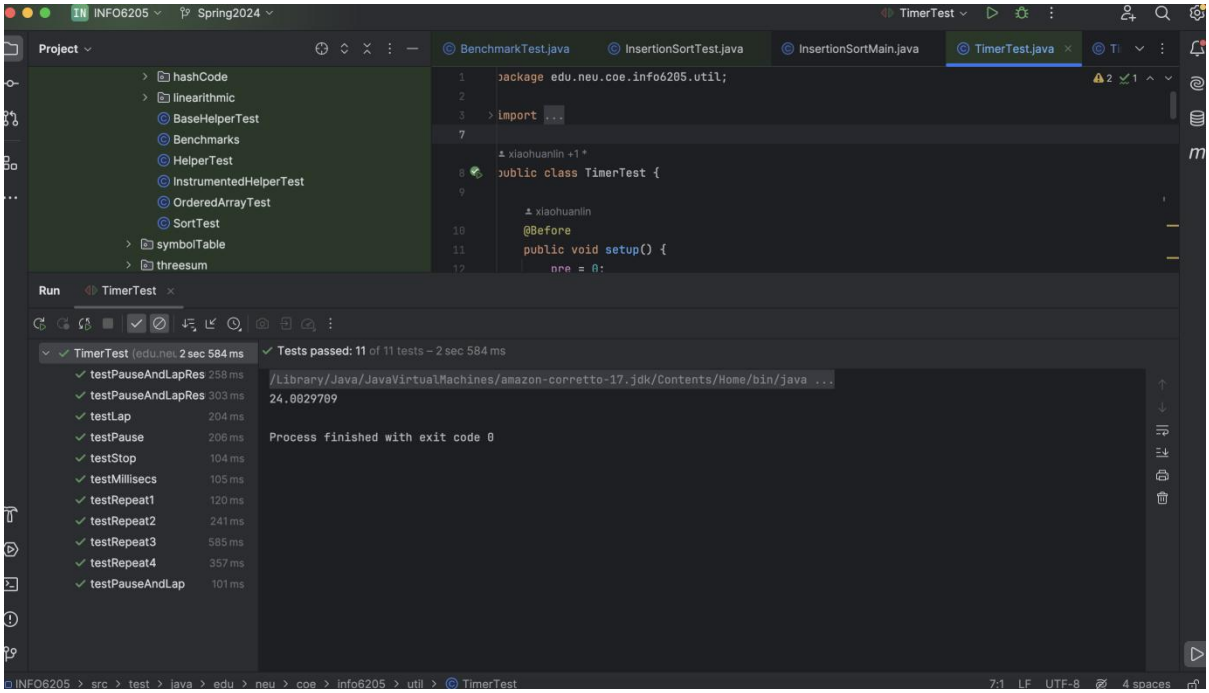
Part2: Using Helper to help realize the functions of comparison and exchange.

Part3: Using benchmark_Timer to test run time. According the result, the efficiency of sorting Orderd Array is obviously higher than other cases. When N starts to get bigger, the speed of dealing with Reverse-Ordered Array obviously slows down, and the time is gradually much longer than others.

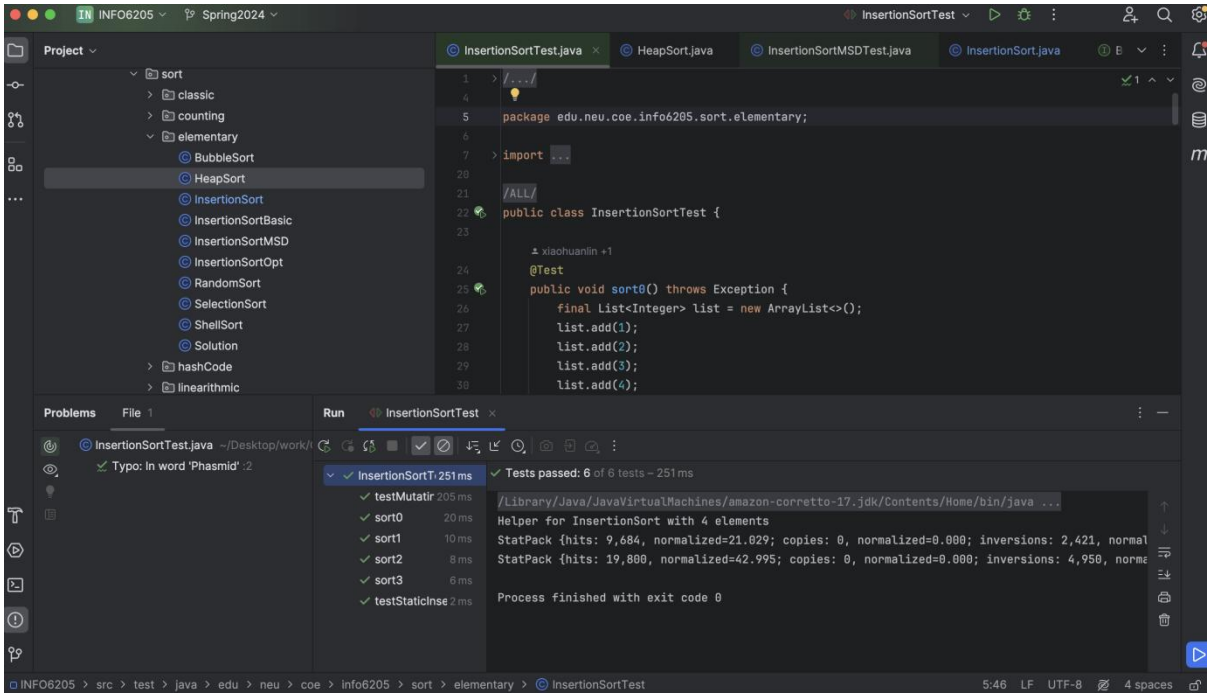
Screenshots:
Benchmark_Timer:



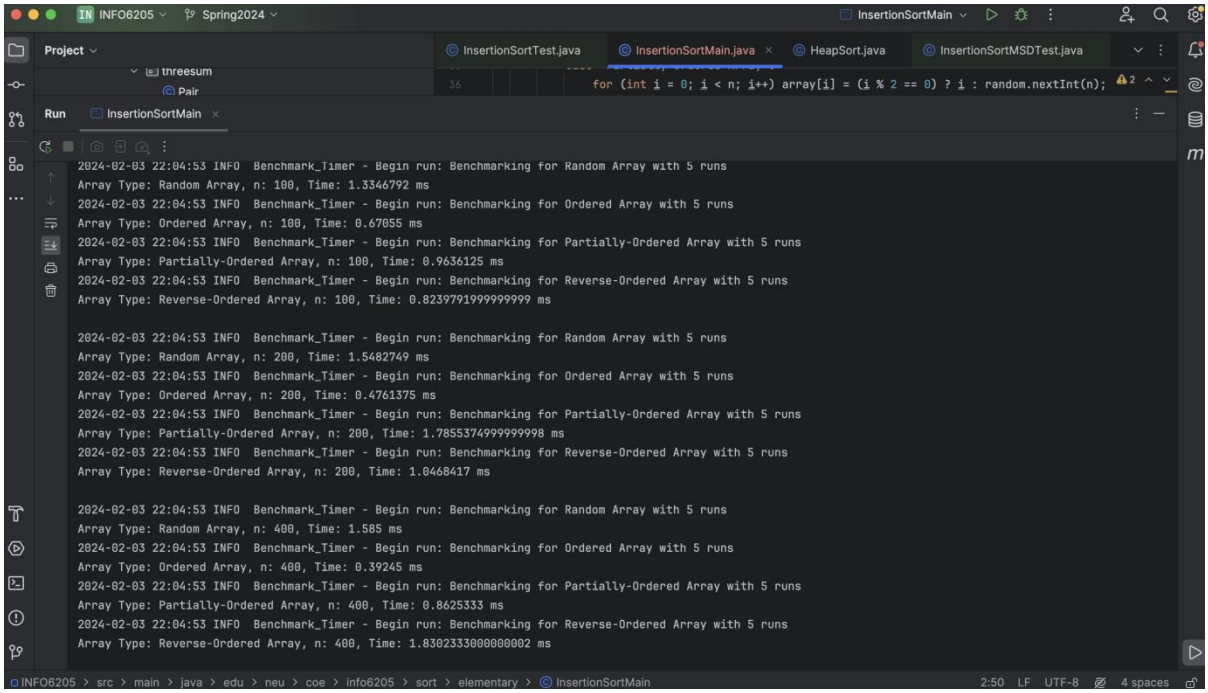
Timer:



InsertionSortTest:



Main:


INFO6205
Spring2024
InsertionSortMain
Project
threesum
Pair
InsertionSortTest.java
InsertionSortMain.java
HeapSort.java
InsertionSortMSDTest.java
for (int i = 0; i < n; i++) array[i] = (i % 2 == 0) ? i : random.nextInt(n);
Run
InsertionSortMain
2024-02-03 22:04:53 INFO Benchmark_Timer - Begin run: Benchmarking for Random Array with 5 runs
Array Type: Random Array, n: 100, Time: 1.3346792 ms
2024-02-03 22:04:53 INFO Benchmark_Timer - Begin run: Benchmarking for Ordered Array with 5 runs
Array Type: Ordered Array, n: 100, Time: 0.67055 ms
2024-02-03 22:04:53 INFO Benchmark_Timer - Begin run: Benchmarking for Partially-Ordered Array with 5 runs
Array Type: Partially-Ordered Array, n: 100, Time: 0.9636125 ms
2024-02-03 22:04:53 INFO Benchmark_Timer - Begin run: Benchmarking for Reverse-Ordered Array with 5 runs
Array Type: Reverse-Ordered Array, n: 100, Time: 0.8239791999999999 ms
2024-02-03 22:04:53 INFO Benchmark_Timer - Begin run: Benchmarking for Random Array with 5 runs
Array Type: Random Array, n: 200, Time: 1.5482749 ms
2024-02-03 22:04:53 INFO Benchmark_Timer - Begin run: Benchmarking for Ordered Array with 5 runs
Array Type: Ordered Array, n: 200, Time: 0.4761375 ms
2024-02-03 22:04:53 INFO Benchmark_Timer - Begin run: Benchmarking for Partially-Ordered Array with 5 runs
Array Type: Partially-Ordered Array, n: 200, Time: 1.7855374999999998 ms
2024-02-03 22:04:53 INFO Benchmark_Timer - Begin run: Benchmarking for Reverse-Ordered Array with 5 runs
Array Type: Reverse-Ordered Array, n: 200, Time: 1.0468417 ms
2024-02-03 22:04:53 INFO Benchmark_Timer - Begin run: Benchmarking for Random Array with 5 runs
Array Type: Random Array, n: 400, Time: 1.585 ms
2024-02-03 22:04:53 INFO Benchmark_Timer - Begin run: Benchmarking for Ordered Array with 5 runs
Array Type: Ordered Array, n: 400, Time: 0.39245 ms
2024-02-03 22:04:53 INFO Benchmark_Timer - Begin run: Benchmarking for Partially-Ordered Array with 5 runs
Array Type: Partially-Ordered Array, n: 400, Time: 0.8625333 ms
2024-02-03 22:04:53 INFO Benchmark_Timer - Begin run: Benchmarking for Reverse-Ordered Array with 5 runs
Array Type: Reverse-Ordered Array, n: 400, Time: 1.8302333000000002 ms
INFO6205 > src > main > java > edu > neu > coe > info6205 > sort > elementary > InsertionSortMain
2:50 LF UTF-8 4 spaces