**Part 1:**

Don't forget to check your implementation by running the unit tests in BenchmarkTest and TimerTest. If you have trouble with the exact timings in the unit tests, it's quite OK (in this assignment only) to change parameters until the tests run. Different machine architectures will result in different behavior.

**Part 1 Unit tests:**

TimerTest：

图形用户界面, 应用程序

描述已自动生成

BenchmarkTest ：

图形用户界面, 文本, 应用程序

描述已自动生成

**Part 2:**

Implement InsertionSort(in the InsertionSort class) by simply looking up the insertion code used byArrays.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 2 Unit tests:**

图形用户界面, 文本, 应用程序

描述已自动生成

**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.

**Git:**

https://github.com/ShiboLu/INFO6205-Shibo-Lu/tree/main/assignment2