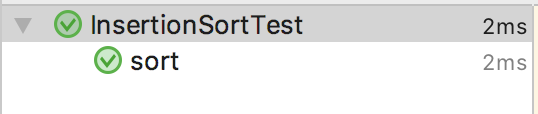
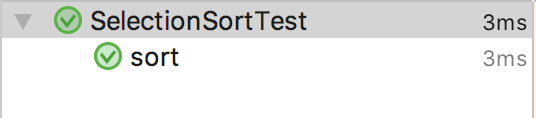
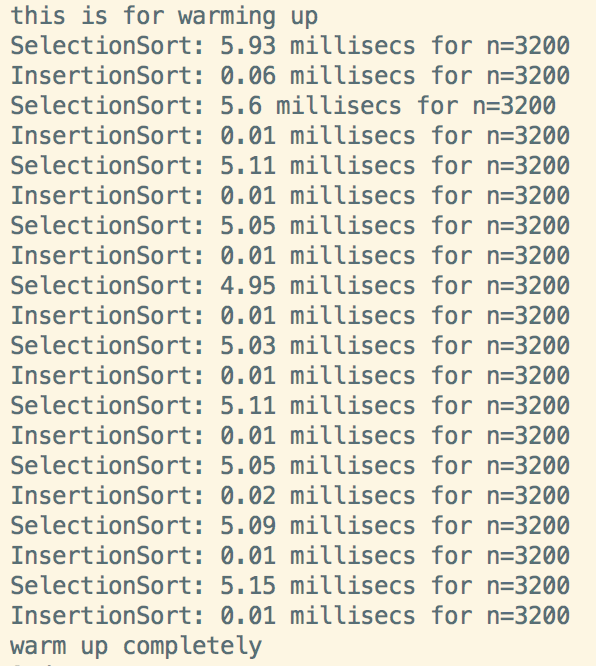
1. Run unit test





1. Run main ()







1. Observation

After warming up, the data begin to be stable.

This form is one of the data output sample:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | n=200 | | | | n=400 | | | | n=800 | | | | n=1600 | | | | n=3200 | | | |
| O | Ra | Re | Pa | O | Ra | Re | Pa | O | Ra | Re | Pa | O | Ra | Re | Pa | O | Ra | Re | Pa |
| Selection | 0.06 | 0.21 | 0.04 | 0.05 | 0.14 | 0.17 | 0.17 | 0.16 | 0.59 | 0.75 | 0.57 | 0.53 | 2.11 | 1.82 | 1.88 | 1.78 | 6.97 | 7.06 | 6.82 | 7.14 |
| Insertion | 0.01 | 0.11 | 0.08 | 0.04 | 0.01 | 0.17 | 0.35 | 0.12 | 0.02 | 0.67 | 1.47 | 0.58 | 0.02 | 3.42 | 6.5 | 1.84 | 0.02 | 10.57 | 19.96 | 5.84 |

For the same “n”, selection sort takes similar time, no matter the sequence is ordered, random or reverse. Insertion sort is influenced by the sequence’s order. If data is ordered, it seems to need little time to sort. If data is reversed, it takes much time. It’s about 2 times than random data.

If n becomes much bigger, the selection sort is faster than insertion sort if the reversed data is a lot. As n is doubling, the time grow 4 times than the former.

Partially sorted array means half sorted and half unsorted.

For n=200, the random data result seems to be wired, that’s because n is too small. After that the results become as we expect.

