Activity 1. Direct exchange or bubble algorithm

Table 1, Bubble algorithm (WITHOUT OPTIMIZATION)

|  |  |  |  |
| --- | --- | --- | --- |
| N | T ordered | T reverse | T random |
| 10000 | 335 ms | 1776 ms | 1024 ms |
| 2\*10000 | 1278 ms | 6986 ms | 4121 ms |
| 2\*\*2\*10000 | 5129 ms | 28005 ms | 16709 ms |
| 2\*\*3\*10000 | 20841 ms | OoT(80K+) | OoT(66k) |
| 2\*\*4\*10000 | OoT | OoT | OoT |

This algorithm has a best complexity of O(n) therefore the ordered one is the fastest.The worst complexity of it is O(n^2) that is why we can see that the reverse is the slowest and the random is in between the values.

Activity 2. Selection algorithm

Table 2, Selection algorithm (WITHOUT OPTIMIZATION)

|  |  |  |  |
| --- | --- | --- | --- |
| N | T ordered | T reverse | T random |
| 10000 | 315 | 288 | 323 |
| 2\*10000 | 1253 | 1128 | 1275 |
| 2\*\*2\*10000 | 5012 | 4479 | 5101 |
| 2\*\*3\*10000 | 20115 | 18272 | 20272 |
| 2\*\*4\*10000 | OoT(70k+) | OoT(74k) | OoT(79k) |

This algorithm has always the same complexity O(n^2) that is why in the three cases the timings are the same.

Activity 3. Insertion algorithm

Table 3, Insertion algorithm (WITHOU OPTIMIZATION)

|  |  |  |  |
| --- | --- | --- | --- |
| N | T ordered | T reverse | T random |
| 10000 | LoR | 304 | 157 |
| 2\*10000 | LoR | 1173 | 606 |
| 2\*\*2\*10000 | LoR | 4816 | 2433 |
| 2\*\*3\*10000 | LoR | 19402 | 9675 |
| 2\*\*4\*10000 | LoR | OoT(76k) | 38466 |
| 2\*\*5\*10000 | LoR | OoT | OoT |
| 2\*\*6\*10000 | LoR | OoT | OoT |
| 2\*\*7\*10000 | LoR | OoT | OoT |
| 2\*\*8\*10000 | 49 | OoT | OoT |
| 2\*\*9\*10000 | 91 | OoT | OoT |
| 2\*\*10\*10000 | 185 | OoT | OoT |
| 2\*\*11\*10000 | 376 | OoT | OoT |
| 2\*\*12\*10000 | 752 | OoT | OoT |
| 2\*\*13\*10000 | 1502 | OoT | OoT |

The best case complexity of this algorithm is O (n) that is why the ordered trimmings are way smaller than the other two. The worst time complexity is O (n^2).

Activity 4. Quicksort algorithm

Table 4, Quicksort algorithm (WITHOU OPTIMIZATION)

Measured in a different PC than the previous ones.

|  |  |  |  |
| --- | --- | --- | --- |
| N | T ordered | T reverse | T random |
| 250000 | 54 | 60 | 134 |
| 2\*250000 | 110 | 122 | 282 |
| 2\*\*2\*250000 | 225 | 252 | 608 |
| 2\*\*3\*250000 | 457 | 515 | 1323 |
| 2\*\*4\*250000 | 955 | 1064 | 2980 |
| 2\*\*5\*250000 | 1927 | 2168 | 6958 |
| 2\*\*6\*250000 | 3990 | 4491 | 17786 |

For our implementation of the quicksort algorithm we have similar complexities for the ordered and the reverse one due to the way we select our pivot, because it does not matter the order of the numbers the one in the middle is always the same. For the random one this is not the case so that is why the measurements are way bigger.

The quicksort algorithm has a complexity of O (n \* log n), and it takes almost 18 seconds,

Therefore, knowing that the complexity of the other algorithms is O(n^2) these algorithms will be 666666 times slower than quicksort. That translates to 18 \* 666666 which is 140 days(approx.).

Activity 5. Quicksort + Insertion algorithm

Table 5, Quicksort + Insertion (WITHOUT OPTIMIZATION)

|  |  |
| --- | --- |
| n | T random |
| Quicksort | 2283 |
| Quicksort + Insertion(K=5) | 10669 |
| Quicksort + Insertion(K=10) | 10534 |
| Quicksort + Insertion(K=20) | 10363 |
| Quicksort + Insertion(K=30) | 10152 |
| Quicksort + Insertion(K=50) | 9830 |
| Quicksort + Insertion(K=100) | 8950 |
| Quicksort + Insertion(K=200) | 7360 |
| Quicksort + Insertion(K=500) | 10190 |
| Quicksort + Insertion(K=1000) | 18284 |