Activity 1. Direct exchange or Bubble algorithm

|  |  |  |  |
| --- | --- | --- | --- |
| n | t ordered | t reverse | t random |
| 10000 | 331 | 1625 | 1134 |
| 20000 | 1290 | 6478 | 4499 |
| 40000 | 5026 | 26326 | 18284 |
| 80000 | 20360 | OoT | OoT |
| 160000 | OoT | OoT | OoT |

Bubble sort has a complexity of n2 in all cases. The complexity match times.

Ordered:

n = 10000 t = 331 // n =20000 t = 1290. Since the complexity is square, by growing twice the problem size the times should be 4 times grater and they are.

Reversed:

Same explanation, 6478/1625 is more or less 4, which makes sense.

Random:

Same explanation, 1134/4499 is more or less 4, makes sense.

Activity 2. Selection algorithm

|  |  |  |  |
| --- | --- | --- | --- |
| n | t ordered | t reverse | t random |
| 10000 | 325 | 292 | 320 |
| 20000 | 1270 | 1170 | 1285 |
| 40000 | 5084 | 4687 | 5134 |
| 80000 | 20499 |  | 20757 |
| 160000 | OoT | OoT | OoT |

This is the same case as Bubble algorithm, the complexity is n2 and times prove it.

For instance, n = 20000 t = 1270 // n = 40000 t = 4687, the division is more or less 4 so it follow quadratic complexity(n grows by 2 time by 4)

Activity 3. Insertion algorithm

|  |  |  |  |
| --- | --- | --- | --- |
| n | t ordered | t reverse | t random |
| 10000 | LoR | 291 | 150 |
| 20000 | LoR | 1171 | 600 |
| 40000 | LoR | 4705 | 2352 |
| 80000 | LoR | 19500 | 9551 |
| 160000 | LoR | OoT | 39195 |
| 320000 | LoR | OoT | OoT |
| 640000 | LoR | OoT | OoT |
| 1280000 | LoR | OoT | OoT |
| 2560000 | 50 | OoT | OoT |
| 5120000 | 100 | OoT | OoT |
| 10240000 | 204 | OoT | OoT |
| 20480000 | 399 | OoT | OoT |
| 40960000 | 803 | OoT | OoT |
| 81920000 | 1599 | OoT | OoT |

In the case of this algorithm, it average complexity is n2, but in the best case (that is, when it is ordered) the complexity is O(n).

This is very good shown in the measurements.

Ordered:  
for instance, when n = 10240000 t = 204 // n = 20480000 t = 399 both the problem size and time is getting multiplied by n so it follows the complexity.

Reversed and random:

As they are not the best case, they have a quadratic complexity.

Let’s take n = 20000 t = 1171 // n = 4000 t = 4705 we can see how the problem size is multiplied by 2 and time by 4 (22), it follows the complexity.

Activity 4. Quicksort algorithm

|  |  |  |  |
| --- | --- | --- | --- |
| n | t ordered | t reverse | t random |
| 250000 | LoR | LoR | 102 |
| 500000 | 62 | 74 | 210 |
| 1000000 | 130 | 152 | 449 |
| 2000000 | 279 | 311 | 964 |
| 4000000 | 556 | 643 | 2092 |
| 8000000 | 1151 | 1308 | 4738 |
| 16000000 | 2395 | 2698 | 11558 |

Quicksort algorithm has an average complexity of O(nlogn).

In this case, as we are using a good pivot (median of three) we get this complexity.

For instance, n = 8000000 t = 4738 // n = 16000000 t = 11558 we can se how by growing the problem size by 2 time is growing a bit more than twice.

Activity 4. Quicksort + Insertion algorithm