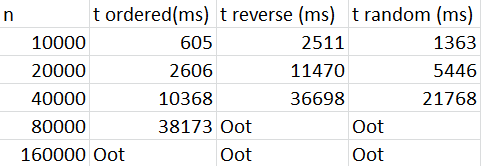
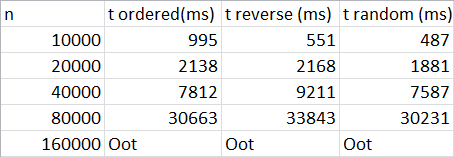
Activity 1. [Direct exchange or Bubble algorithm]

The measurements are taken with a processor i7-4790 and 8 GB DDR3



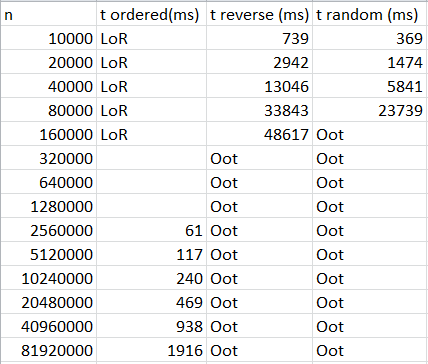
As expected, the ordered array is the one that spends less time and the reverse the one which spends more time. This is because with the ordered array it doesn`t have to do much work while with the reverse one it has to change all the positions. The random one spends less time than the reverse one as some positions will be already ordered so the algorithm has to do less work.

Activity 2. [Selection algorithm]



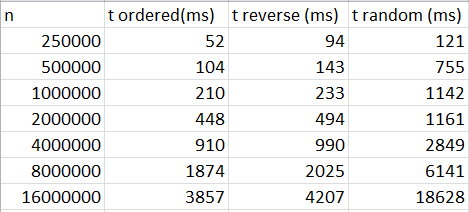
It agrees with what is expected as the selection algorithm takes the lowest element every time and put it at the beginning of the array. This is why the reverse array works better this time than the ordered array because this one requires to go to the end of the array searching for the lowest element although it is the already selected. Random needs less time as some are already ordered.

Activity 3. [Insertion algorithm]



As we expected the times are reasonable because this algorithm inserts each element into its position inside a sub vector already ordered so for the ordered vector requires less time than the random one and the reverse one as it is already ordered so all interactions are omitted. The random spends less time than the reverse one as is completely the other way around, in the reverse one it has to be inserting the element one by one traversing the entire vector while in the random some elements are already ordered so it has to do less interactions.

Activity 4. [Quicksort algorithm]



As expected, these algorithms spend little time in any kind of array, showing us that it is the most effective sorting algorithms in terms of performance. We can see that the random vector spends a little more time than the others two this is probably because it requires more interactions.

We can see how many days would each of those three methods (Bubble, Selection and Insertion) take in doing the same by doing a rule of 3 in each case taking in account the size of 16M

We use the formula t2 = k^n\*t1 where k = n2/n1

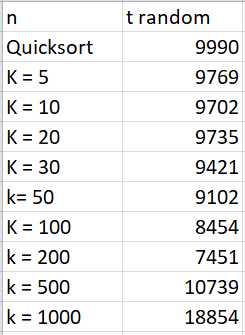
Bubble -> t = (16M/10000)^2\* 1363ms = 3489280000ms = 40,39 days

Selection -> t = (16M/10000)^2\*487ms = 1246720000ms = 14,3 days

Insertion -> t = (16M/10000)^2\*369ms = 944640000ms = 10,9 days

Activity 5. [Quicksort + Insertion algorithm]

This time the measurements are done with a processor i5-12600KF and 32GB DDR5



After taking the measurements, we can conclude that the range in where we can see the difference in terms of improving the performance is between k = 50 and k = 200. So we can see that it is interesting to mix algorithms to get a better performance.