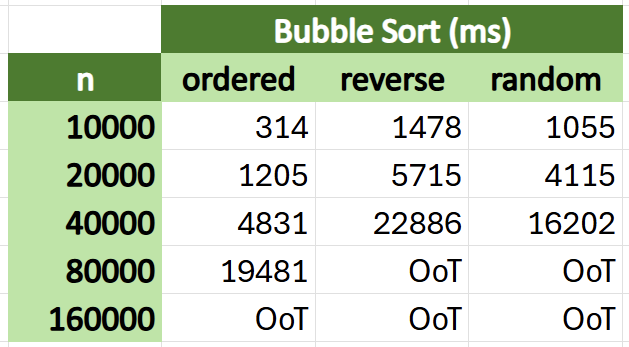
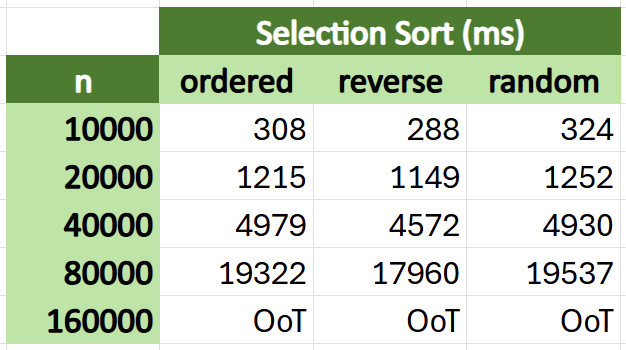
Activity 1. Bubble Sort



* Explain whether the different times obtained agree with what is expected, according to the time complexity studied.

They do. Bubble Sort has an average complexity of O(n), which is coherent with the data measured.

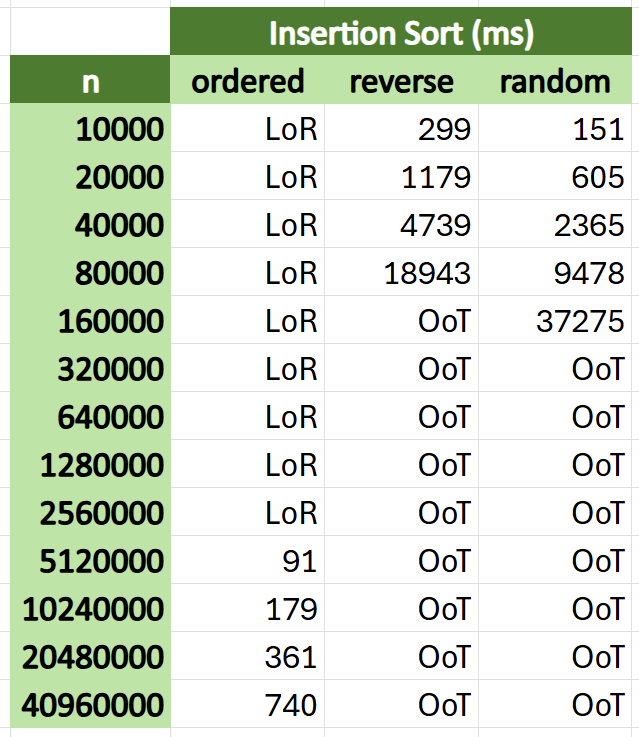
Activity 2. Selection Sort



* Explain whether the different times obtained agree with what is expected, according to the time complexity studied.

Yes, they do. Selection Sort also has an average complexity of O(n).

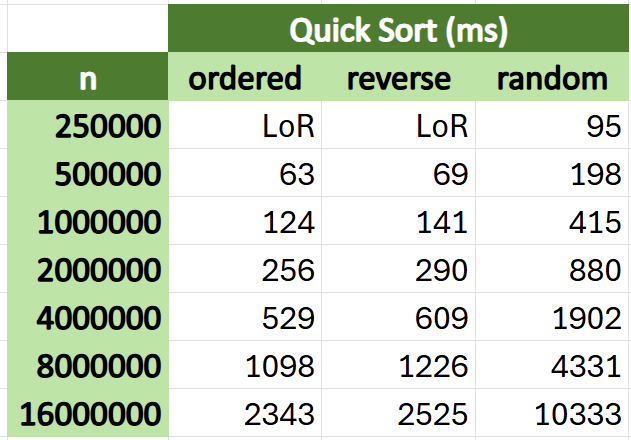
Activity 3. Insertion Sort



* Explain whether the different times obtained agree with what is expected, according to the time complexity studied.

The times show that Insertion Sort is roughly O(n) when the list is already ordered and O(n) when it’s not, just as we could have predicted.

Activity 4. Quick Sort



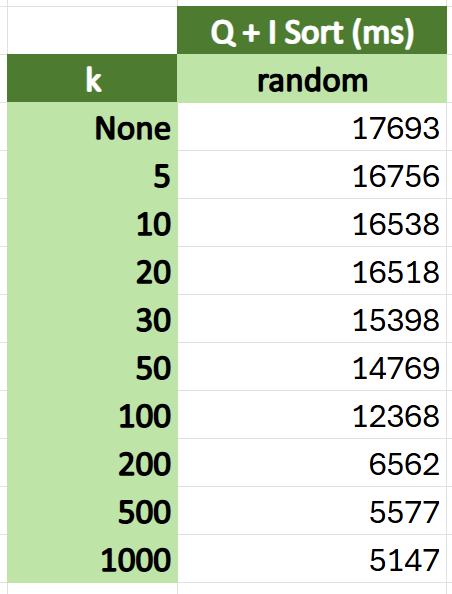
* Explain whether the different times obtained agree with what is expected, according to the time complexity studied.

In this case the times seem to follow a linear trend, as if the complexity were O(2n), so I’m not quite sure what’s happening...

* After seeing how long it takes to sort 16 million items, initially in a random order, calculate and compare (from the complexities and data in the tables above): How many days would each of those three methods (Bubble, Selection and Insertion) take to do the same?

Answer: = 2.29 x 10 ms = ***265.61 days!!***

Activity 5. Quick + Insertion Sort



* Explain conclusions obtained from the previous table

This table shows the growing effectiveness of blending the two algorithms together as we increase k, which in this case becomes especially noticeable when k >= 200. Beyond a certain value, however, Insertion Sort would presumably stop performing as well, so further testing would be required to strike the right balance.