Task 1:

Modify my implementation of binary search so it keeps track of the number of the number of comparisons performed. Then, test it out with input lists of various sizes. Does the number of comparisons agree with the claimed O(logn) complexity?

Make sure that the list is sorted, or the algorithm will not work.

I would encourage you to pair-program these next two tasks, and switch roles after **Task 2**.

Task 2:

Watch the bubble sort animation (**note** – you'll have to change the algorithm in the dropdown on this site) and implement the `bubbleSort` function such that it sorts the input list and returns the number of comparison steps (take a look at the provided sorting.py, which shows how we keep track of, and return, the number of swaps performed). What is the algorithmic complexity if your implementation? Why?

Task 3:

Watch the selection sort animation and implement the `selectionSort` function such that it sorts the input list and returns the number of comparison steps (take a look at the provided `sorting.py`, which shows how we keep track of, and return, the number of swaps performed). What is the algorithmic complexity if your implementation? Why?