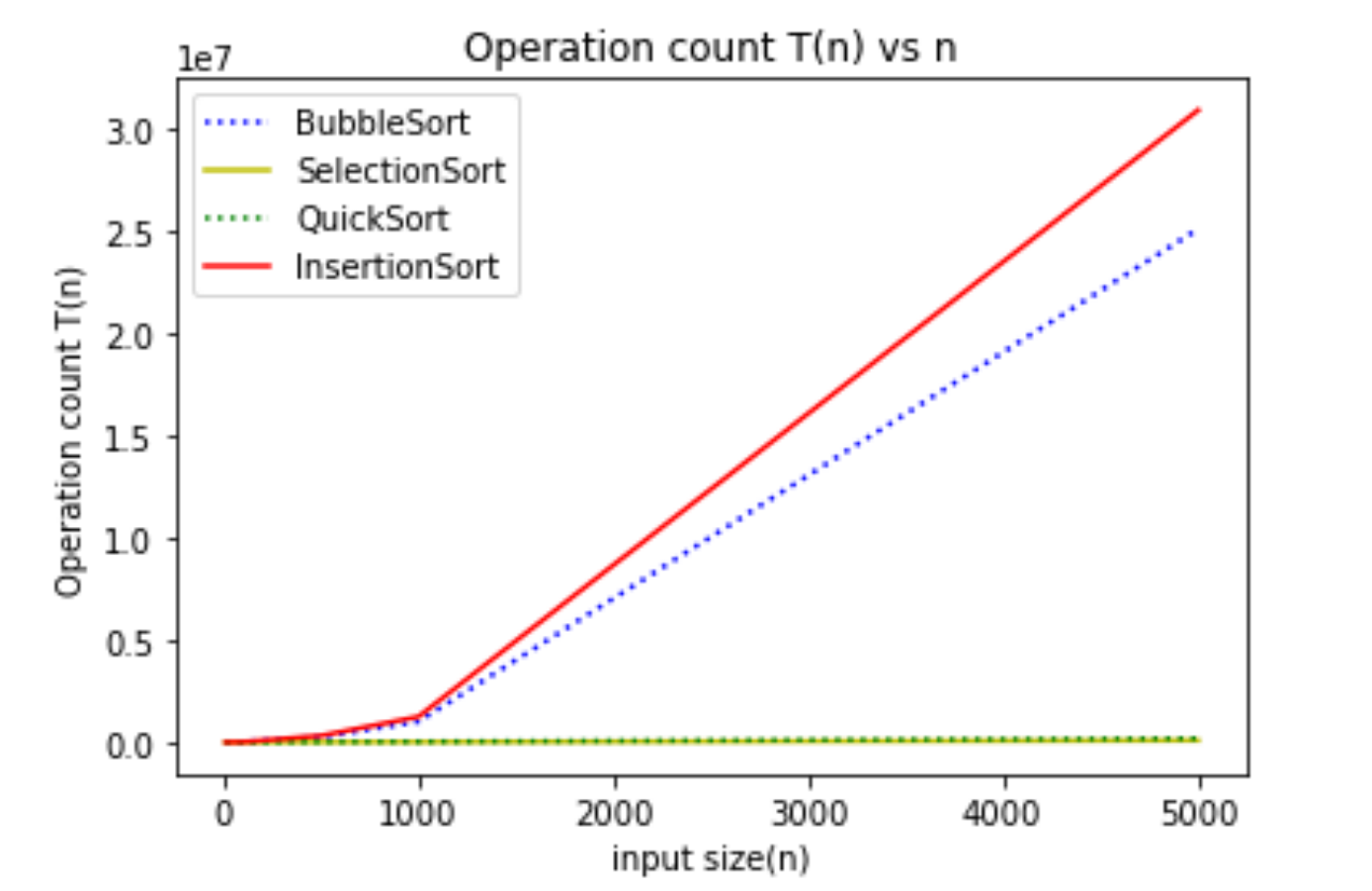
Step 1: Implementation for sorting algorithms are given in sort.py fie.

Step 2:

Bubble Sort: For the given standard bubble sort implementation, the running time is O(n^2) for all best , Worst , and Average cases. This implementation does not depend on the order of elements in the list. Two for loops still have to run and they give O(n^2)

Step 3:

As shown in the graph below, quick sort and selection sort take approximately the same amount of operations whereas bubble sort and insertion sort stay close to each other with respect to operation counts.



Step 4:

As shown in the graph below, execution time kind of resembles the graph in step 3. By inspecting the graph below , these sorting algorithms can be listed in terms of their runtime from low to high as

1. Quick Sort – best performer
2. Selection Sort
3. Insertion Sort
4. Bubble Sort - worst performer

A picture containing text, screenshot, line, plot

Description automatically generated

**Note on running the program:**

I used Spyder IDE to run my program. Please put sort.py, plot.py and test.py in one folder and run test.py

to generate the graphs shown here. I have only used input size up to 5000 as it takes much longer with higher values.