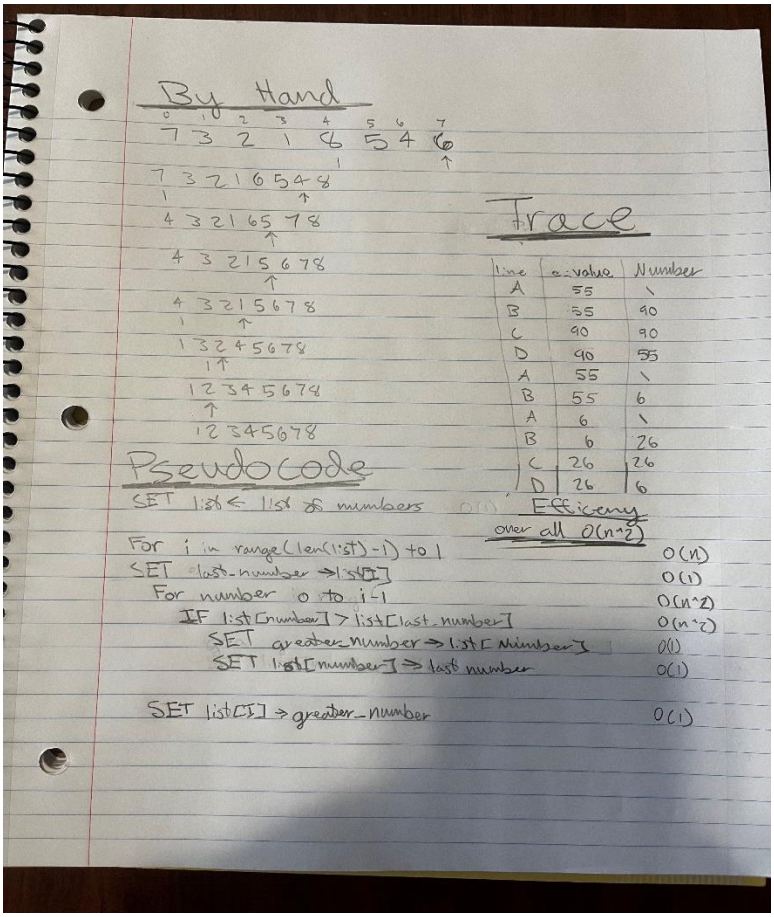

To sort a list using this algorithm. You will have a list of numbers. Start at the end of the list and get the value that is at the end of the list. Store that value in a variable, then starting at the beginning of the list check each value at each index. If the value at any index is greater than the first value at the end of the list swap those two numbers. After that repeat the same process for the entire list starting at the end of the list and moving towards the beginning

Copilot generated code:

```
# Pseudocode for the specified sorting algorithm


# Define the list
```

```
list = [list of numbers]

# Outer loop: Start from the end of the list and move towards the beginning
for i from len(list) - 1 to 1
    # Store the value at the current end of the list in a variable
    end_value = list[i]

    # Inner loop: Check each value from the start of the list up to the current position
    for j from 0 to i - 1
        # If the value at the current position is greater than the end_value, swap them
        if list[j] > end_value
            temp = list[j]
            list[j] = end_value
            end_value = temp

    # Place the end_value at the current end position
    list[i] = end_value

# The list is now sorted
```

Compare and contrast:

After looking at the code for a couple of minutes. I think it is very similar, but there are some key differences.

- Provide an analysis as to the pros and cons of the two solutions.

I think the biggest pro is, they both will most likely sort the list. Another pro is both of the algorithms are simple and easy to read and understand. A con is for the copilot algorithm it has some unnecessary variables that it uses in the second for loop. It stores the value in another variable and them it stores that variable into the list, it could just remove the second variable.

- How can your solution be improved based on what Copilot provided?

The biggest thing that it suggested I can improve is the naming of the variables. The names that I used were very long. Although that suggestion would make it easier to type and faster to create the algorithm, I do like the longer variable names because they are more descriptive of what is actually happening in the program, and I find it easier to understand.

- How can Copilot's solution be improved based on what you know?

I think that the unnecessary variable could just be removed. I don't think that there is a use for the variable temp. Other than that, I don't have any suggestions for copilot to improve.

- Does the pseudocode in Step 3 and Step 4 match the algorithm you performed in Step 1?

Yes, I think both of the pseudocode examples will complete the sorting algorithm.

Step 1 By Hand: 15 minutes

Step 2 Approach: 20

Step 3 Pseudocode: 45 minutes

Step 4 Copilot: 10 minutes

Step 5 Compare and Contrast: 25 minutes

Step 6 Update: 15 minutes

Step 7 Trace: 45 minutes

Step 8 Efficiency: 30

minutes