1) (5 pts) ANL (Algorithm Analysis)

What is the **worst-case** Big O runtime for the following function, in terms of the input parameter, **n**? (You may assume that the array pointed to by list is of length **n**.) In order to receive full credit, you must use words to explain your reasoning AND arrive at the correct answer.

```
int mystery(int* list, int n) {

    int i = 0, j = 1;

    if (n < 2) return 0;

    while (j < n) {

        while (i < j && list[i] < list[j]) i++;
        j++;
    }

    return i;
}
```

REASON:

It's impossible for either i or j to exceed n, since i will never get greater than j and j will never get greater than n. There's O(1) extra work before either i or j are incremented. This means that the inner while loop never runs more than n times **total** and the outer while loop never runs more than n times **total**. In fact, structurally, as long as n >= 2, the outer while loop runs precisely n – 1 times. It follows that the runtime of this code is simply **O(n).**

RUN-TIME: (n)

Grading: **1 pt for the correct answer**
4 pts for the reason: 1 pt to argue that outer loop runs <= n times.
3 pts to argue that the inner loop never runs more than n times.

If the answer is wrong, max score is 1/5 for arguing that the outer loop runs no more than n times.