Day 20: Sorting

Objective
Today, we're discussing a simple sorting algorithm called *Bubble Sort*. Check out the Tutorial tab for learning materials and an instructional video!

Consider the following version of Bubble Sort:

```
for (int i = 0; i < n; i++) {
    int numberOfSwaps = 0;

    for (int j = 0; j < n - 1; j++) {
        if (a[j] > a[j + 1]) {
            swap(a[j], a[j + 1]);
            numberOfSwaps++;
        }
    }

    if (numberOfSwaps == 0) {
        break;
    }
}
```

Task
Given an array, a, of size n containing distinct elements $a[0], a[1], \ldots, a[n-1]$, sort array a in ascending order using the *Bubble Sort* algorithm above. Once sorted, print the following 3 lines:

1. `Array is sorted in` $numSwaps$ `swaps.`
   where $numSwaps$ is the number of swaps that took place.
2. `First Element:` $firstElement$
   where $firstElement$ is the *first* element in the sorted array.
3. `Last Element:` $lastElement$
   where $lastElement$ is the *last* element in the sorted array.

Hint: To complete this challenge, you will need to add a variable that keeps a running tally of *all* swaps that occur during execution.

Input Format

The first line contains an integer, n, denoting the number of elements in array a.
The second line contains n space-separated integers describing a, where the i^{th} integer is $a[i]$, $\forall\, i \in [0,\ n-1]$.

Constraints

- $2 \leq n \leq 600$
- $1 \leq a[i] \leq 2 \times 10^6$, $\forall\, i \in [0,\ n-1]$

Output Format

There should be 3 lines of output:

1. `Array is sorted in` $numSwaps$ `swaps.`
   where $numSwaps$ is the number of swaps that took place.

2. **`First Element:`** $firstElement$
   where $firstElement$ is the *first* element in the sorted array.

3. **`Last Element:`** $lastElement$
   where $lastElement$ is the *last* element in the sorted array.

Sample Input 0

```
3
1 2 3
```

Sample Output 0

```
Array is sorted in 0 swaps.
First Element: 1
Last Element: 3
```

Sample Input 1

```
3
3 2 1
```

Sample Output 1

```
Array is sorted in 3 swaps.
First Element: 1
Last Element: 3
```

Explanation

Sample Case 1:
The array is already sorted, so 0 swaps take place and we print the necessary 3 lines of output shown above.

Sample Case 2:
The array is *not sorted*, and its initial values are: $\{3, 2, 1\}$. The following 3 swaps take place:

1. $\{3, 2, 1\} \rightarrow \{2, 3, 1\}$
2. $\{2, 3, 1\} \rightarrow \{2, 1, 3\}$
3. $\{2, 1, 3\} \rightarrow \{1, 2, 3\}$

At this point the array is sorted and we print the necessary 3 lines of output shown above.