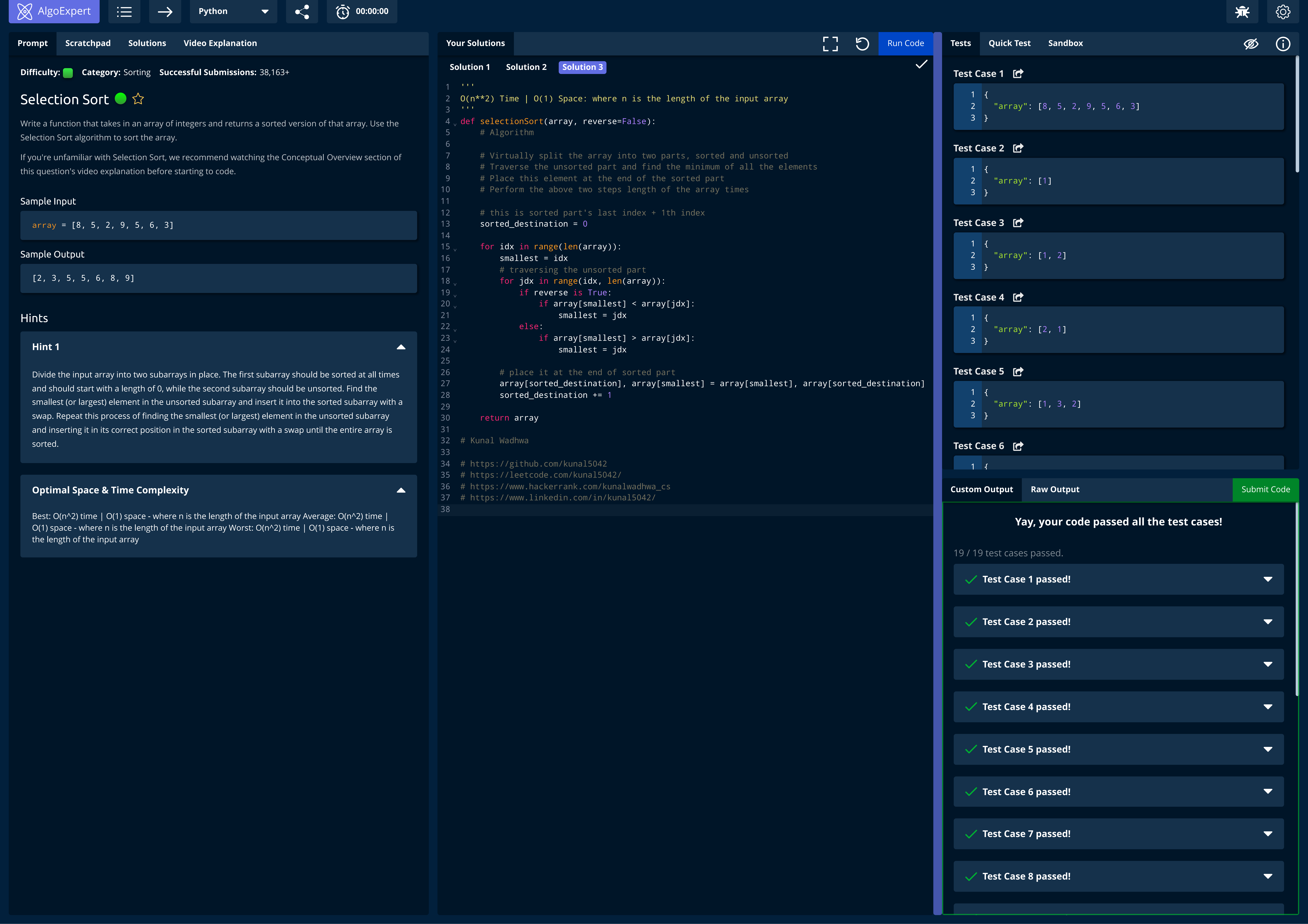

Difficulty: 🟩 **Category:** Sorting **Successful Submissions:** 38,163+

Selection Sort ● ☆

Write a function that takes in an array of integers and returns a sorted version of that array. Use the Selection Sort algorithm to sort the array.

If you're unfamiliar with Selection Sort, we recommend watching the Conceptual Overview section of this question's video explanation before starting to code.

Sample Input

```
array = [8, 5, 2, 9, 5, 6, 3]
```

Sample Output

```
[2, 3, 5, 5, 6, 8, 9]
```

Hints

Hint 1

Divide the input array into two subarrays in place. The first subarray should be sorted at all times and should start with a length of 0, while the second subarray should be unsorted. Find the smallest (or largest) element in the unsorted subarray and insert it into the sorted subarray with a swap. Repeat this process of finding the smallest (or largest) element in the unsorted subarray and inserting it in its correct position in the sorted subarray with a swap until the entire array is sorted.

Optimal Space & Time Complexity

Best: O(n^2) time | O(1) space - where n is the length of the input array Average: O(n^2) time | O(1) space - where n is the length of the input array Worst: O(n^2) time | O(1) space - where n is the length of the input array

```python
'''
O(n**2) Time | O(1) Space: where n is the length of the input array
'''
def selectionSort(array, reverse=False):
    # Algorithm

    # Virtually split the array into two parts, sorted and unsorted
    # Traverse the unsorted part and find the minimum of all the elements
    # Place this element at the end of the sorted part
    # Perform the above two steps length of the array times

    # this is sorted part's last index + 1th index
    sorted_destination = 0

    for idx in range(len(array)):
        smallest = idx
        # traversing the unsorted part
        for jdx in range(idx, len(array)):
            if reverse is True:
                if array[smallest] < array[jdx]:
                    smallest = jdx
            else:
                if array[smallest] > array[jdx]:
                    smallest = jdx

        # place it at the end of sorted part
        array[sorted_destination], array[smallest] = array[smallest], array[sorted_destination]
        sorted_destination += 1

    return array

# Kunal Wadhwa

# https://github.com/kunal5042
# https://leetcode.com/kunal5042/
# https://www.hackerrank.com/kunalwadhwa_cs
# https://www.linkedin.com/in/kunal5042/

```

Test Case 1

```
{
  "array": [8, 5, 2, 9, 5, 6, 3]
}
```

Test Case 2

```
{
  "array": [1]
}
```

Test Case 3

```
{
  "array": [1, 2]
}
```

Test Case 4

```
{
  "array": [2, 1]
}
```

Test Case 5

```
{
  "array": [1, 3, 2]
}
```

Test Case 6

```
{
```

Yay, your code passed all the test cases!

19 / 19 test cases passed.

✓ Test Case 1 passed!

✓ Test Case 2 passed!

✓ Test Case 3 passed!

✓ Test Case 4 passed!

✓ Test Case 5 passed!

✓ Test Case 6 passed!

✓ Test Case 7 passed!

✓ Test Case 8 passed!