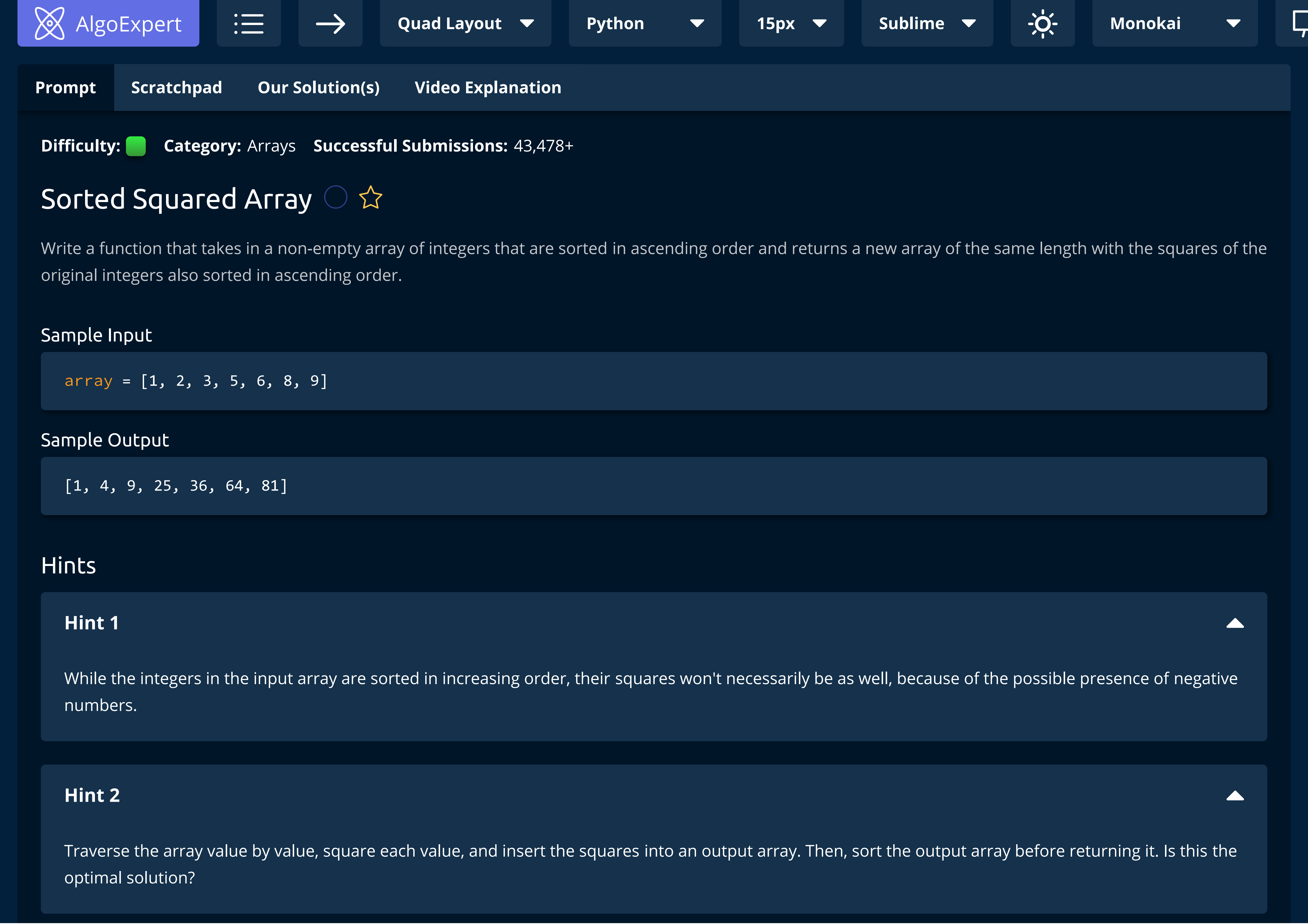


Difficulty: 🟩 **Category:** Arrays **Successful Submissions:** 43,478+

Sorted Squared Array ○ ☆

Write a function that takes in a non-empty array of integers that are sorted in ascending order and returns a new array of the same length with the squares of the original integers also sorted in ascending order.

Sample Input

```
array = [1, 2, 3, 5, 6, 8, 9]
```

Sample Output

```
[1, 4, 9, 25, 36, 64, 81]
```

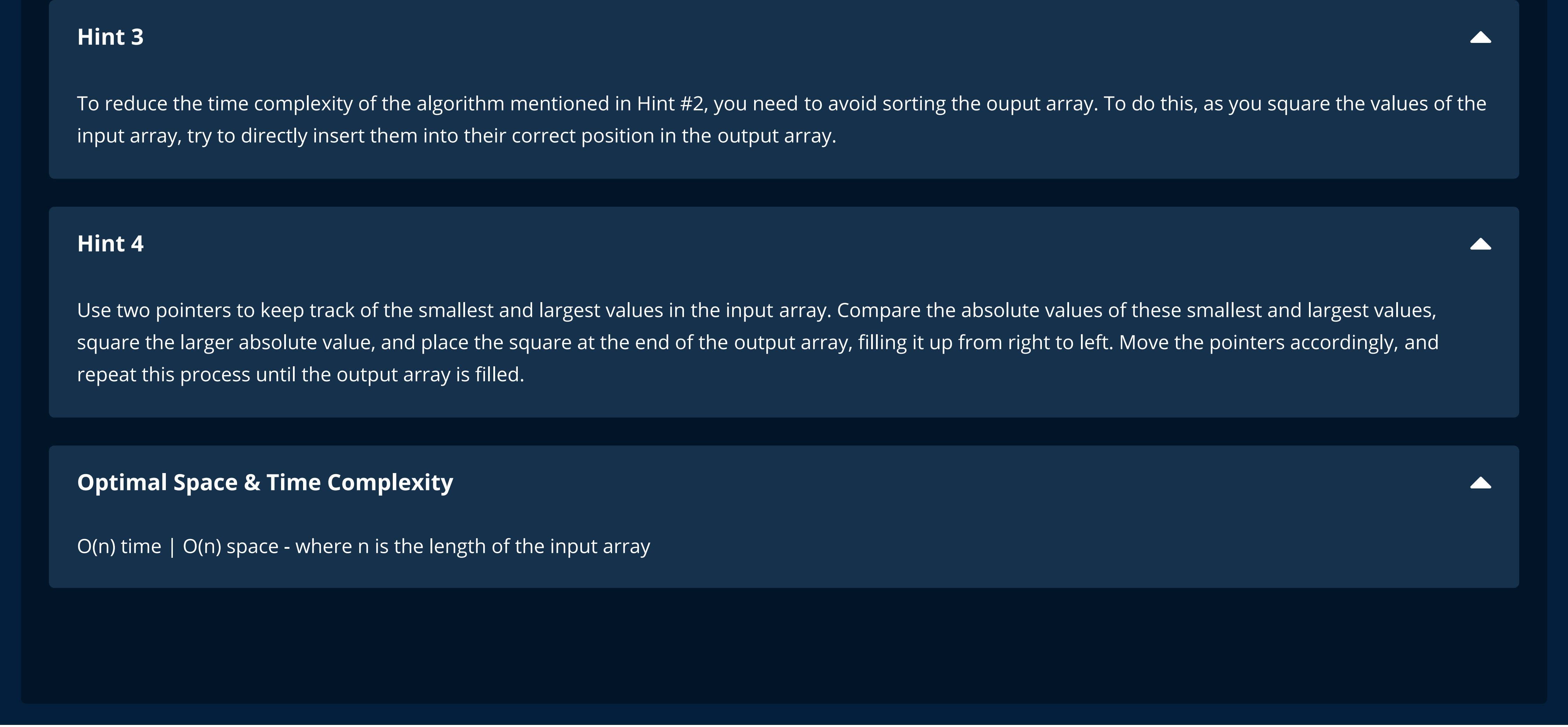
Hints

Hint 1

While the integers in the input array are sorted in increasing order, their squares won't necessarily be as well, because of the possible presence of negative numbers.

Hint 2

Traverse the array value by value, square each value, and insert the squares into an output array. Then, sort the output array before returning it. Is this the optimal solution?



Hint 3

To reduce the time complexity of the algorithm mentioned in Hint #2, you need to avoid sorting the ouput array. To do this, as you square the values of the input array, try to directly insert them into their correct position in the output array.

Hint 4

Use two pointers to keep track of the smallest and largest values in the input array. Compare the absolute values of these smallest and largest values, square the larger absolute value, and place the square at the end of the output array, filling it up from right to left. Move the pointers accordingly, and repeat this process until the output array is filled.

Optimal Space & Time Complexity

O(n) time | O(n) space - where n is the length of the input array