

Test1

The screenshot shows the PyCharm IDE with a project named 'quiz12'. The file 'main.py' is open, displaying a Python function 'find_both2' and its usage. The function takes parameters 'n', 's', 'small', and 'large'. It initializes 'small' and 'large' based on the first two elements of 's'. Then, it iterates through 's' from index 2 to 'n-1', updating 'small' and 'large' to the minimum and maximum values found so far. Finally, it returns 'small' and 'large'. The main code prompts the user for 'n' and the array 'arr', then calls 'find_both2' with 'n', 'arr', 0, and 0.

```
1 def find_both2(n, s, small, large):
2     if s[0] < s[1]:
3         small = s[0]
4         large = s[1]
5     else:
6         small = s[1]
7         large = s[0]
8
9     for i in range(3, n-1):
10        if s[i] < s[i+1]:
11            if s[i] < small:
12                small = s[i]
13            if s[i+1] > large:
14                large = s[i+1]
15        else:
16            if s[i+1] < small:
17                small = s[i+1]
18            if s[i] > large:
19                large = s[i]
20    return small, large
21
22 n = int(input('please input n:'))
23 arr = input('please input the array:')
24 arr = [int(n) for n in arr.split()]
25 print('output:', find_both2(n, arr, 0, 0))
```

The Run window shows the execution output:

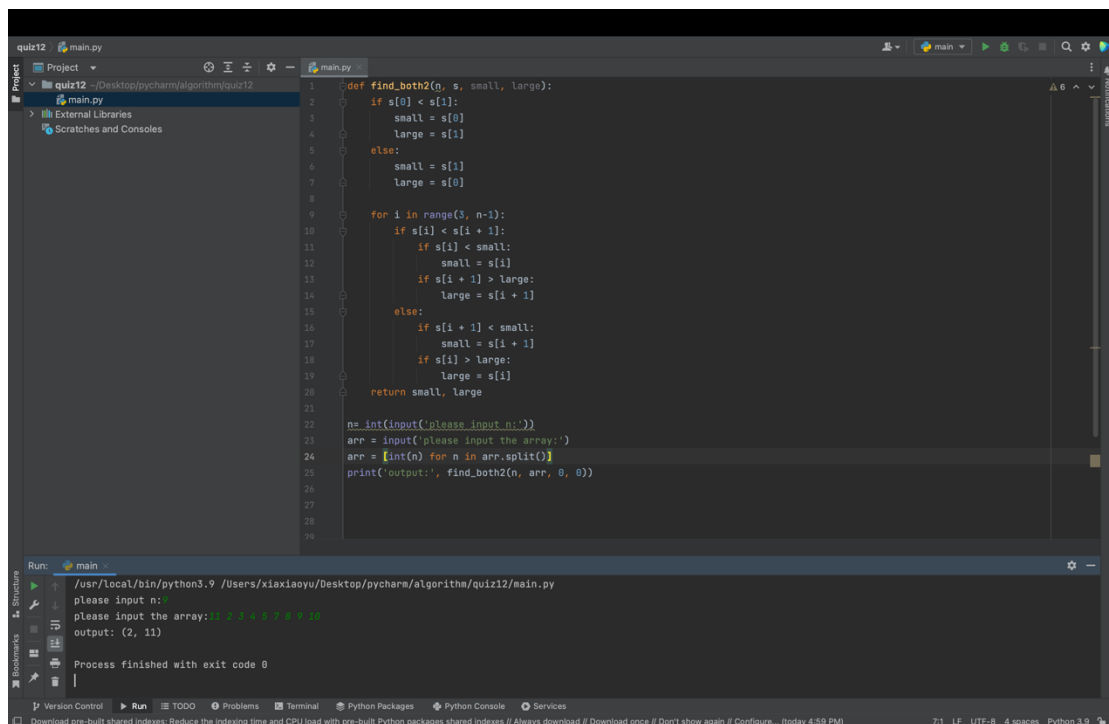
```
Run: main
/usr/local/bin/python3.9 /Users/xiaxiaoyu/Desktop/pycharm/algorithm/quiz12/main.py
please input n: 6
please input the array: 1 1 1 1 1 1
output: (1, 9)
Process finished with exit code 0
```

Test2

The screenshot shows the PyCharm IDE with the same project 'quiz12' and file 'main.py'. The code is identical to the one in Test1. The Run window shows the execution output for a different input:

```
Run: main
/usr/local/bin/python3.9 /Users/xiaxiaoyu/Desktop/pycharm/algorithm/quiz12/main.py
please input n: 6
please input the array: 1 1 1 1 1 1
output: (6, 10)
Process finished with exit code 0
```

Test3



The screenshot shows a PyCharm IDE with a Python file named `main.py`. The code defines a function `find_both2` that takes `n`, `s`, `small`, and `large` as arguments. It then takes user input for `n` and an array, and prints the output of `find_both2`. The Run console shows the execution with `n=2` and the array `[1, 2, 3, 4, 5, 6, 7, 8]`, resulting in an output of `(2, 11)`.

```
1 def find_both2(n, s, small, large):
2     if s[0] < s[1]:
3         small = s[0]
4         large = s[1]
5     else:
6         small = s[1]
7         large = s[0]
8
9     for i in range(3, n-1):
10        if s[i] < s[i+1]:
11            if s[i] < small:
12                small = s[i]
13            if s[i+1] > large:
14                large = s[i+1]
15        else:
16            if s[i+1] < small:
17                small = s[i+1]
18            if s[i] > large:
19                large = s[i]
20    return small, large
21
22 n = int(input('please input n:'))
23 arr = input('please input the array:')
24 arr = [int(n) for n in arr.split()]
25 print('output:', find_both2(n, arr, 0, 0))
26
27
28
29
```

Run: main

```
/usr/local/bin/python3.9 /Users/xiaiaoyu/Desktop/pycharm/algorithm/quiz12/main.py
please input n: 2
please input the array: 1 2 3 4 5 6 7 8
output: (2, 11)

Process finished with exit code 0
```

Time complexity:

Time complexity:

$$\frac{3n}{2} - 2 \quad \text{if } n \text{ is even}$$
$$\frac{3n}{2} - \frac{3}{2} \quad \text{if } n \text{ is odd}$$

even:

① each two keys need one comparison

So total we need $\frac{n}{2}$ comparisons.

Then we get $2 \cdot \frac{n}{2}$ information.

② we must do at least $2n-2-n = n-2$ comparisons because adversary provides us just one information.

③ $\frac{n}{2} + n - 2 = \frac{3n}{2} - 2$

adversary need the algorithm to do additional comparisons.

So Time complexity is $\frac{3n}{2} - 2$ if n is even.

odd:

① make one comparison between $s[0]$ and $s[1]$
to get initial 'small' and 'large'

② we need three types of comparison

1) the neighbor element

2) if $small < min$

3) if $large > max$.

in each type, we have $n-3$ elements to compare.

so, the number of comparisons for this part is $3 \cdot \frac{n-3}{2}$

③ make the last comparison with both min and max. $\Rightarrow 2$

so total time complexity is $1 + 3 \cdot \frac{n-3}{2} + 2 = \frac{3n}{2} - \frac{3}{2}$

So Time complexity is $\frac{3n}{2} - \frac{3}{2}$ if n is odd.