Python

```python
def handle_list_cases(arr):
    if not arr:
        return arr
    return sorted(arr)
print(handle_list_cases([]))
print(handle_list_cases([1]))
print(handle_list_cases([7, 7, 7, 7]))
print(handle_list_cases([-5, -1, -3, -2, -4]))
```

Input Goes Here..

Copy

> Run

> Run+URL (Generates URL as well)

Time(sec) : 0.012

Memory(MB) : 6.1796875

Output:

Copy

```
[]
[1]
[7, 7, 7, 7]
[-5, -4, -3, -2, -1]
```

Python

```python
def selection_sort(arr):
    n = len(arr)
    for i in range(n):
        min_index = i
        for j in range(i+1, n):
            if arr[j] < arr[min_index]:
                min_index = j
        arr[i], arr[min_index] = arr[min_index], arr[i]
    return arr
print(selection_sort([5, 2, 9, 1, 5, 6]))

```

Input Goes Here..

Copy

Run

Run+URL (Generates URL as well)

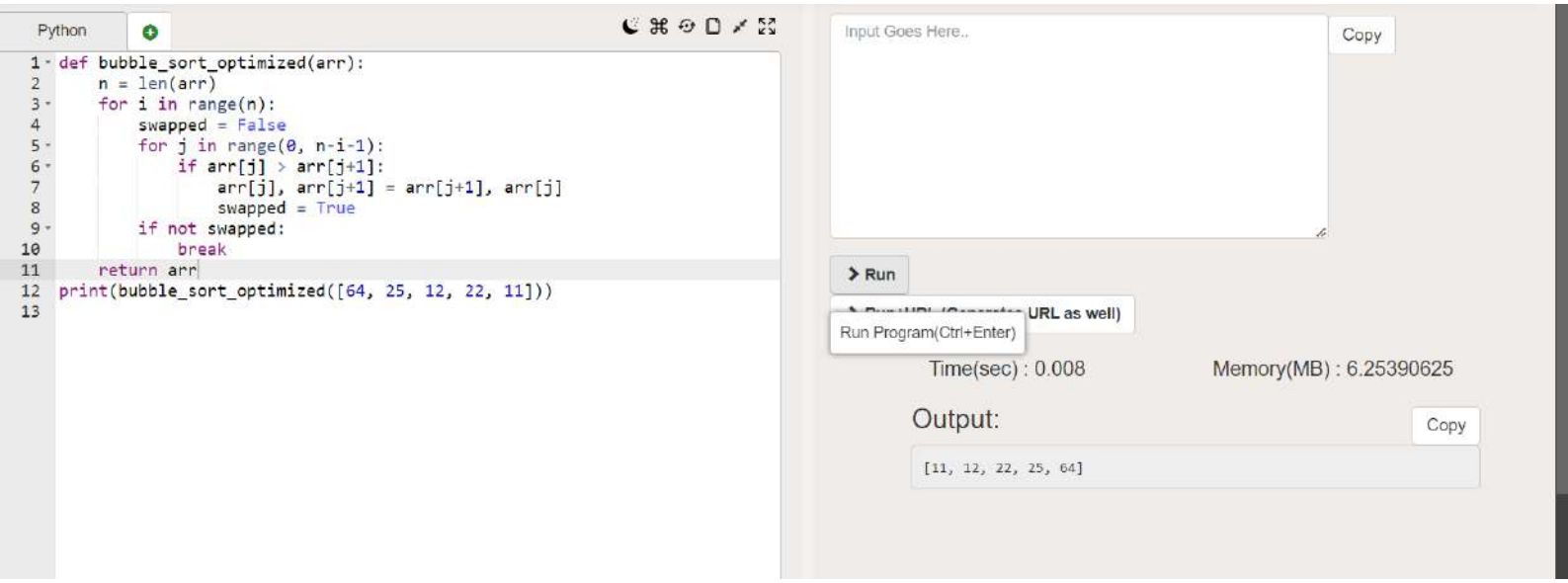
Time(sec) : 0.017

Memory(MB) : 6.19140625

Output:

Copy

```
[1, 2, 5, 5, 6, 9]
```


Python
def bubble_sort_optimized(arr):
n = len(arr)
for i in range(n):
swapped = False
for j in range(0, n-i-1):
if arr[j] > arr[j+1]:
arr[j], arr[j+1] = arr[j+1], arr[j]
swapped = True
if not swapped:
break
return arr
print(bubble_sort_optimized([64, 25, 12, 22, 11]))
Input Goes Here..
Copy
Run
Run Program(Ctrl+Enter)
Time(sec) : 0.008
Memory(MB) : 6.25390625
Output:
Copy
[11, 12, 22, 25, 64]

Python

```python
def insertion_sort(arr):
    for i in range(1, len(arr)):
        key = arr[i]
        j = i - 1
        while j >= 0 and key < arr[j]:
            arr[j + 1] = arr[j]
            j -= 1
        arr[j + 1] = key
    return arr
print(insertion_sort([3, 1, 4, 1, 5, 9, 2, 6, 5, 3]))

```

Input Goes Here..

Copy

> Run

Run Program(Ctrl+Enter)

Time(sec) : 0.011

Memory(MB) : 6.25390625

Output:

Copy

```
[1, 1, 2, 3, 3, 4, 5, 5, 6, 9]
```

Python

```python
def find_kth_missing(arr, k):
    missing = []
    current = 1
    for num in arr:
        while current < num:
            missing.append(current)
            current += 1
        current += 1
    while len(missing) < k:
        missing.append(current)
        current += 1
    return missing[k-1]
print(find_kth_missing([2, 3, 4, 7, 11], 5))

```

Input Goes Here..

Copy

> Run

> Run+URL (Generates URL as well)

Time(sec) : 0.005

Memory(MB) : 6.26171875

Output:

Copy

```
9
```

Python

```python
def find_peak_element(nums):
    left, right = 0, len(nums) - 1
    while left < right:
        mid = (left + right) // 2
        if nums[mid] > nums[mid + 1]:
            right = mid
        else:
            left = mid + 1
    return left
print(find_peak_element([1, 2, 3, 1]))

```

Input Goes Here..

Copy

> Run

> Run+URL (Generates URL as well)

Time(sec) : 0.016

Memory(MB) : 6.15625

Output:

Copy

```
2
5
```

Python

```python
def find_substring(haystack, needle):
    return haystack.find(needle)
print(find_substring("sadbutsad", "sad"))
print(find_substring("leetcode", "leeto"))

```

Input Goes Here..

Copy

> Run

> Run+URL (Generates URL as well)

Time(sec) : 0.002

Memory(MB) : 6.1484375

Output:

Copy

```
0
-1
```

Python

```python
def find_substrings(words):
    result = []
    for i in range(len(words)):
        for j in range(len(words)):
            if i != j and words[i] in words[j]:
                result.append(words[i])
                break
    return result
print(find_substrings(["mass", "as", "hero", "superhero"]))
```

Input Goes Here..

Copy

Run

Run Program(Ctrl+Enter)

Time(sec) : 0.006

Memory(MB) : 6.15234375

Output:

Copy

```
['as', 'hero']
```