



main.py



Share

Run

Output

Clear

```
1 def process_list(input_list):
2     if not input_list:
3         return []
4     elif len(input_list) == 1:
5         return input_list
6     elif all(x == input_list[0] for x in input_list):
7         return input_list
8     else:
9         return sorted(input_list)
10 print(process_list([]))
11 print(process_list([1]))
12 print(process_list([7, 7, 7, 7]))
13 print(process_list([-5, -1, -3, -2, -4]))
```

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

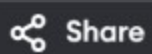
=== Code Execution Successful ===
```



JS



main.py



Run

Output

Clear

```
1 def find_kth_missing(arr, k):
2     missing_count = 0
3     current_num = 1
4     idx = 0
5     while missing_count < k:
6         if idx < len(arr) and arr[idx] == current_num:
7             idx += 1
8         else:
9             missing_count += 1
10            if missing_count == k:
11                return current_num
12            current_num += 1
13    return current_num
14 print(find_kth_missing([2, 3, 4, 7, 11], 5))
15 print(find_kth_missing([1, 2, 3, 4], 2))
16
```

9

6

=== Code Execution Successful ===



JS



main.py



Share

Run

Output

Clear

```
1 def strStr(haystack, needle):  
2     return haystack.find(needle)  
3 print(strStr("sadbutsad", "sad"))  
4 print(strStr("leetcode", "leeto"))  
5
```

0

-1

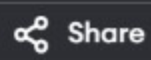
=== Code Execution Successful ===



JS



main.py



Run

Output

Clear



JS



```
1 def selection_sort(arr):
2     n = len(arr)
3     for i in range(n):
4         min_index = i
5         for j in range(i + 1, n):
6             if arr[j] < arr[min_index]:
7                 min_index = j
8         arr[i], arr[min_index] = arr[min_index], arr[i]
9     return arr
10 print(selection_sort([5, 2, 9, 1, 5, 6]))
11 print(selection_sort([10, 8, 6, 4, 2]))
12 print(selection_sort([1, 2, 3, 4, 5]))
13
```

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

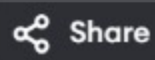
```
[2, 4, 6, 8, 10]
```

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

```
=== Code Execution Successful ===
```



main.py



Share

Run

Output

Clear



JS

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

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

```
=== Code Execution Successful ===
```



main.py



Share

Run

Output

Clear



JS

GO

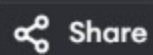
```
1 def bubble_sort(arr):
2     n = len(arr)
3     for i in range(n):
4         swapped = False
5         for j in range(0, n - i - 1):
6             if arr[j] > arr[j + 1]:
7                 arr[j], arr[j + 1] = arr[j + 1], arr[j]
8                 swapped = True
9         if not swapped:
10            break
11    return arr
12 print(bubble_sort([5, 2, 9, 1, 5, 6]))
13 print(bubble_sort([10, 8, 6, 4, 2]))
14 print(bubble_sort([1, 2, 3, 4, 5]))
15
```

```
[1, 2, 5, 5, 6, 9]
[2, 4, 6, 8, 10]
[1, 2, 3, 4, 5]
```

```
=== Code Execution Successful ===
```



main.py



Run

Output

Clear

```
1 def bubble_sort(arr):
2     n = len(arr)
3     for i in range(n):
4         swapped = False
5         for j in range(0, n - i - 1):
6             if arr[j] > arr[j + 1]:
7                 arr[j], arr[j + 1] = arr[j + 1], arr[j]
8                 swapped = True
9         if not swapped:
10             break
11     return arr
12 print(bubble_sort([64, 25, 12, 22, 11]))
13 print(bubble_sort([29, 10, 14, 37, 13]))
14 print(bubble_sort([3, 5, 2, 1, 4]))
15 print(bubble_sort([1, 2, 3, 4, 5]))
16
```

[11, 12, 22, 25, 64]

[10, 13, 14, 29, 37]

[1, 2, 3, 4, 5]

[1, 2, 3, 4, 5]

=== Code Execution Successful ===

JS