



main.py



Share

Run

Output

```
1 empty_list = []  
2 print(empty_list)  
3
```

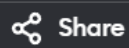
```
[]
```

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





main.py



Share

Run

Output

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

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

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



JS



main.py



Share

Run

Output

Share code

```
1 def bubble_sort_early_termination(arr):
2     n = len(arr)
3     for i in range(n):
4         already_sorted = True
5         for j in range(n - i - 1):
6             if arr[j] > arr[j + 1]:
7                 arr[j], arr[j + 1] = arr[j + 1], arr[j]
8                 already_sorted = False
9         if already_sorted:
10             break
11     return arr
12 my_list = [64, 34, 25, 12, 22, 11, 90]
13 sorted_list = bubble_sort_early_termination(my_list)
14 print(sorted_list)
15
```

[11, 12, 22, 25, 34, 64, 90]

=== Code Execution Successful ===

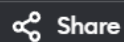


JS





main.py



Share

Run

Output

```
1 def bubble_sort(arr):
2     n = len(arr)
3     for i in range(n):
4         for j in range(0, n-i-1):
5             if arr[j] > arr[j+1]:
6                 arr[j], arr[j+1] = arr[j+1], arr[j]
7     return arr
8 input_list = [64, 25, 12, 22, 11]
9 sorted_list = bubble_sort(input_list)
10 print(sorted_list)
11
```

[11, 12, 22, 25, 64]

=== Code Execution Successful ===



JS





main.py



Share

Run

Output

```
1 def insertion_sort_with_duplicates(arr):
2     for i in range(1, len(arr)):
3         key = arr[i]
4         j = i - 1
5         while j >= 0 and key < arr[j]:
6             arr[j + 1] = arr[j]
7             j -= 1
8         arr[j + 1] = key
9     return arr
10 input_arr = [3, 1, 4, 1, 5, 9, 2, 6, 5, 3]
11 sorted_arr = insertion_sort_with_duplicates(input_arr)
12 print(sorted_arr)
13
```

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

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



JS



main.py



Share

Run

Output

```
1 def find_kth_missing(arr, k):
2     missing_set = set(range(1, arr[-1] + k + 1)) - set(arr)
3     return sorted(missing_set)[k - 1]
4 arr = [2, 3, 4, 7, 11]
5 k = 5
6 result = find_kth_missing(arr, k)
7 print(result) # Output: 9
8
```

9

=== Code Execution Successful ===

main.py



Share

Run

Output

```
1 def find_peak_element(nums):
2     left, right = 0, len(nums) - 1
3
4     while left < right:
5         mid = left + (right - left) // 2
6
7         if nums[mid] < nums[mid + 1]:
8             left = mid + 1
9         else:
10            right = mid
11
12     return left
13
```

=== Code Execution Successful ===

main.py



Share

Run

Output

```
1 def find_substrings(words):
2     result = []
3     for i in range(len(words)):
4         for j in range(len(words)):
5             if i != j and words[i] in words[j]:
6                 result.append(words[i])
7                 break
8     return list(set(result))
9 words = ["mass", "as", "hero", "superhero"]
10 print(find_substrings(words))
11
```

['hero', 'as']

=== Code Execution Successful ===