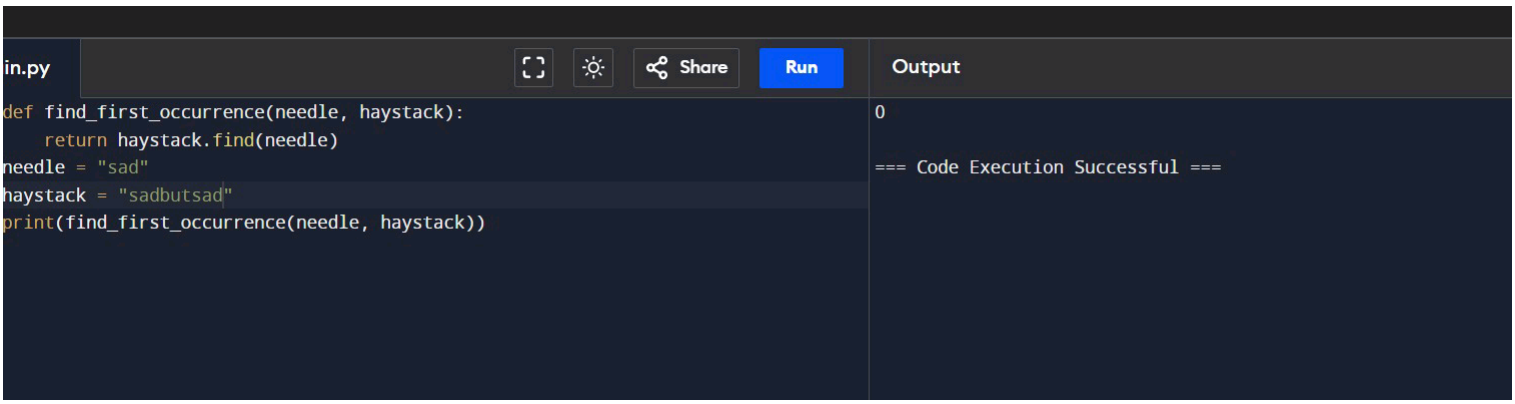
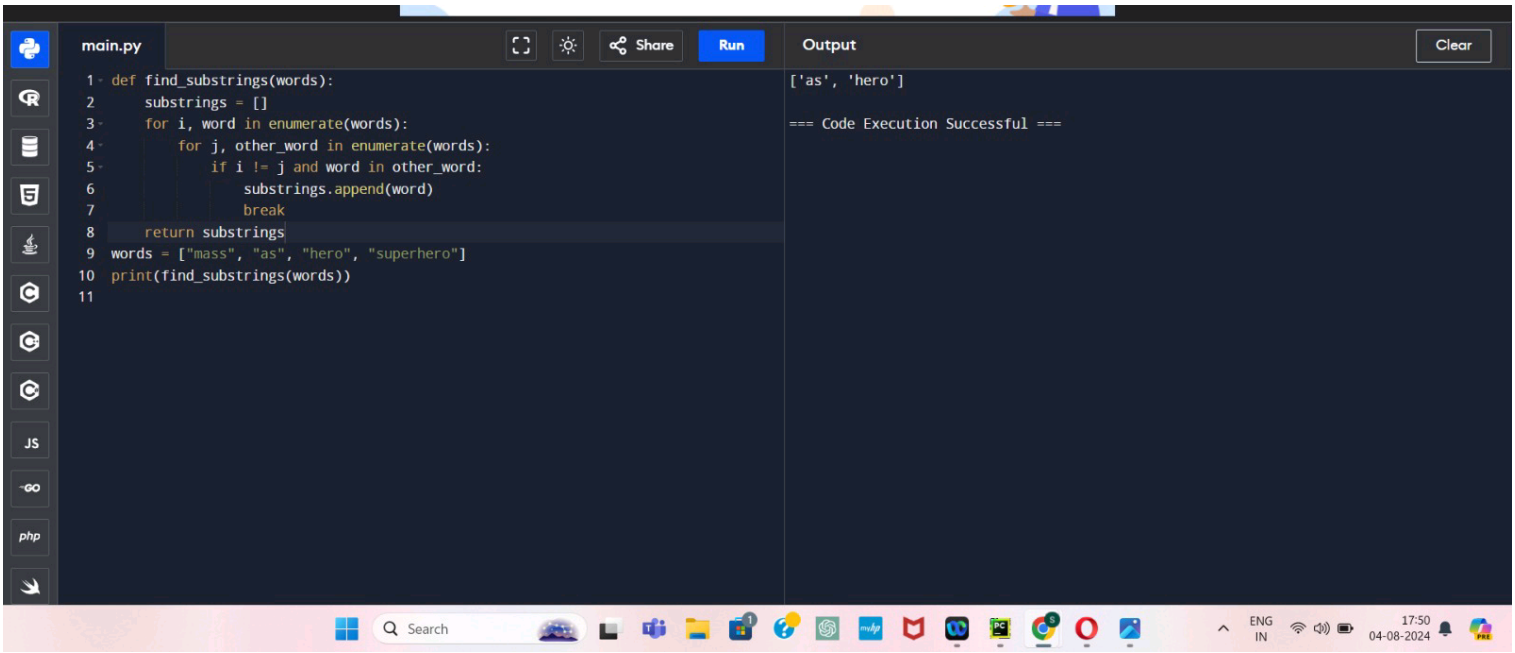
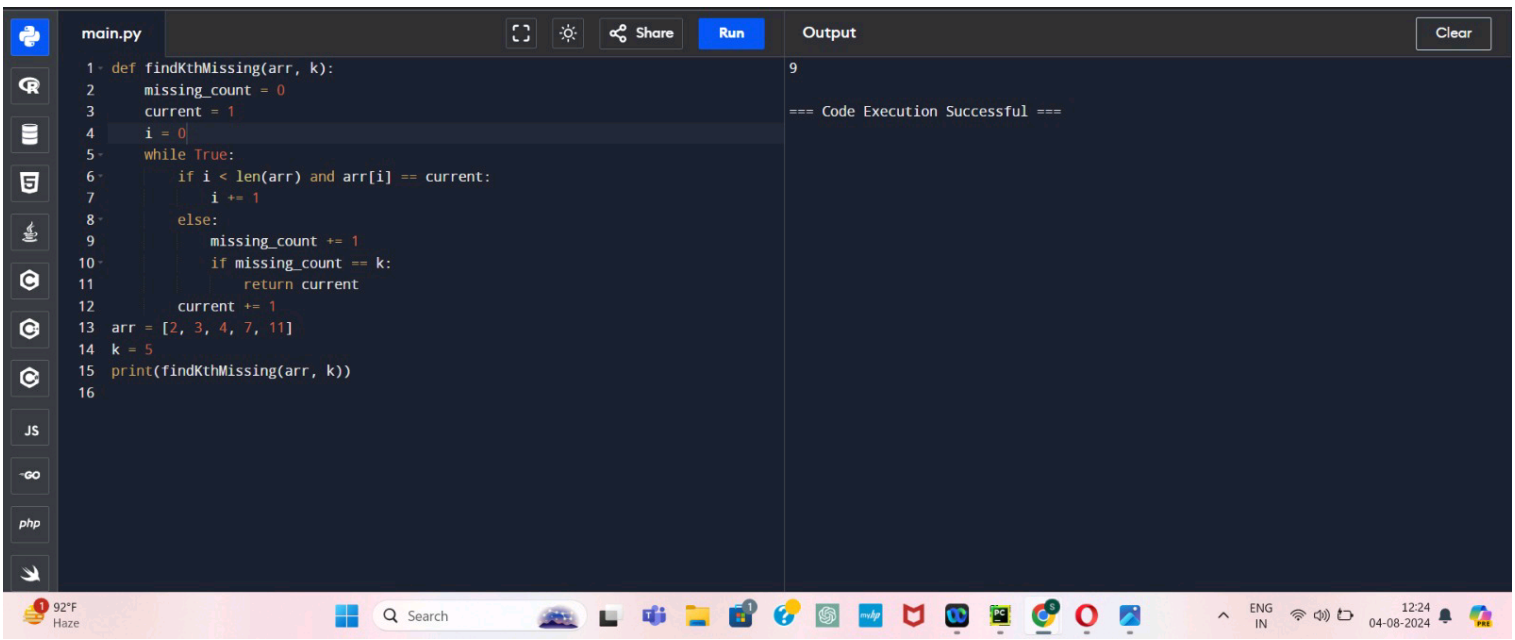
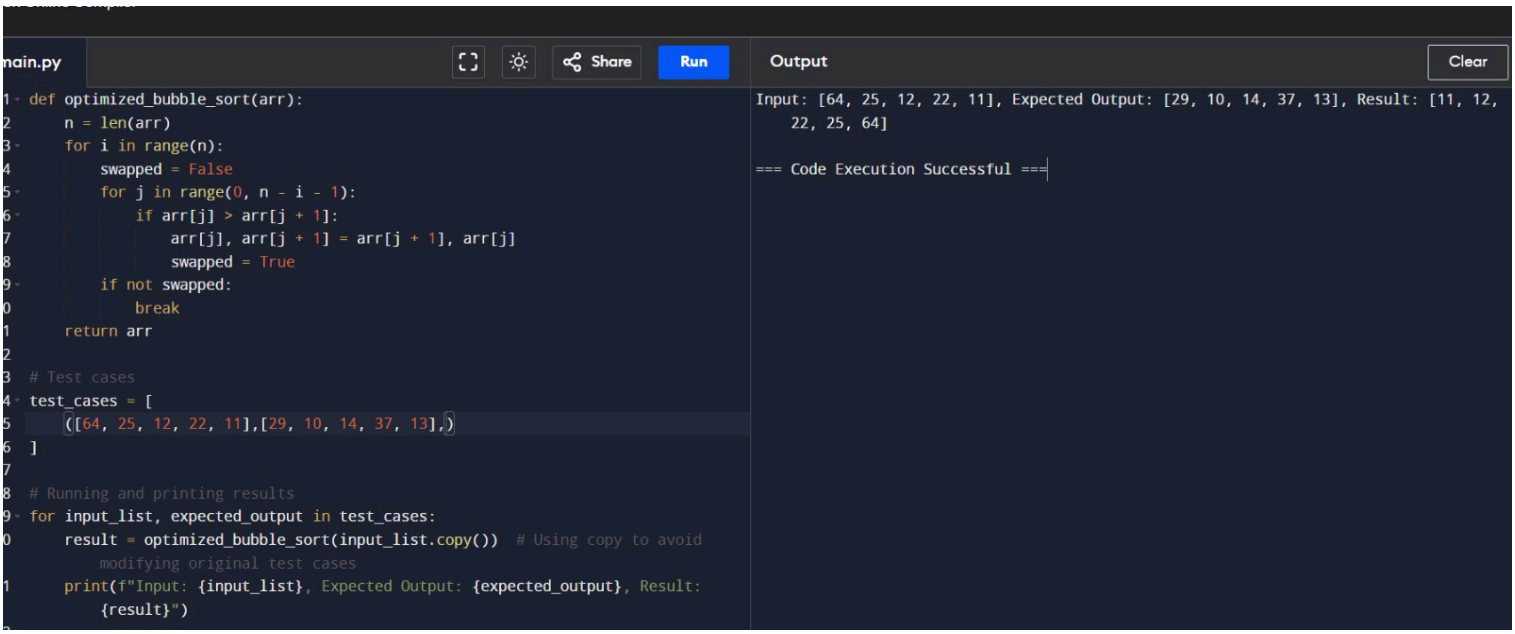

in.py
Share
Run
Output
def find_first_occurrence(needle, haystack):
return haystack.find(needle)
needle = "sad"
haystack = "sadbutsad"
print(find_first_occurrence(needle, haystack))
0
=== Code Execution Successful ===

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


main.py
Share
Run
Output
Clear
def find_substrings(words):
substrings = []
for i, word in enumerate(words):
for j, other_word in enumerate(words):
if i != j and word in other_word:
substrings.append(word)
break
return substrings
words = ["mass", "as", "hero", "superhero"]
print(find_substrings(words))
['as', 'hero']
=== Code Execution Successful ===
Search
ENG
IN
17:50
04-08-2024


main.py
Share
Run
Output
Clear
def findKthMissing(arr, k):
missing_count = 0
current = 1
i = 0
while True:
if i < len(arr) and arr[i] == current:
i += 1
else:
missing_count += 1
if missing_count == k:
return current
current += 1
arr = [2, 3, 4, 7, 11]
k = 5
print(findKthMissing(arr, k))
9
=== Code Execution Successful ===
92°F
Haze
Search
ENG
IN
12:24
04-08-2024


main.py
Share
Run
Output
Clear
def optimized_bubble_sort(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
# Test cases
test_cases = [
([64, 25, 12, 22, 11],[29, 10, 14, 37, 13],)
]
# Running and printing results
for input_list, expected_output in test_cases:
result = optimized_bubble_sort(input_list.copy()) # Using copy to avoid modifying original test cases
print(f"Input: {input_list}, Expected Output: {expected_output}, Result: {result}")
Input: [64, 25, 12, 22, 11], Expected Output: [29, 10, 14, 37, 13], Result: [11, 12, 22, 25, 64]
=== Code Execution Successful ===

main.py

Share Run

```python
def insertion_sort(arr):
    n = len(arr)
    for i in range(1, n):
        key = arr[i]
        j = i - 1
        while j >= 0 and arr[j] > key:
            arr[j + 1] = arr[j]
            j -= 1
        arr[j + 1] = key
    return arr
test_cases = [
    ([3, 1, 4, 1, 5, 9, 2, 6, 5, 3], [1, 1, 2, 3, 3, 4, 5, 5, 6, 9]),
]
for input_list, expected_output in test_cases:
    result = insertion_sort(input_list.copy())
    print(f"Input: {input_list}, Expected Output: {expected_output}, Result:
        {result}")

```

Output Clear

```
Input: [3, 1, 4, 1, 5, 9, 2, 6, 5, 3], Expected Output: [1, 1, 2, 3, 3, 4, 5, 5, 6, 9]
    , Result: [1, 1, 2, 3, 3, 4, 5, 5, 6, 9]

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

main.py

Share Run

```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]
        print(f"Pass {i + 1}: {arr}")
    return arr
random_array = [5, 2, 9, 1, 5, 6]
print("Sorting a Random Array:")
print("Original array:", random_array)
sorted_random_array = selection_sort(random_array.copy())
print("Sorted array:", sorted_random_array)
print("\n")
reverse_sorted_array = [10, 8, 6, 4, 2]
print("Sorting a Reverse Sorted Array:")
print("Original array:", reverse_sorted_array)
sorted_reverse_array = selection_sort(reverse_sorted_array.copy())
print("Sorted array:", sorted_reverse_array)
print("\n")
```

Output

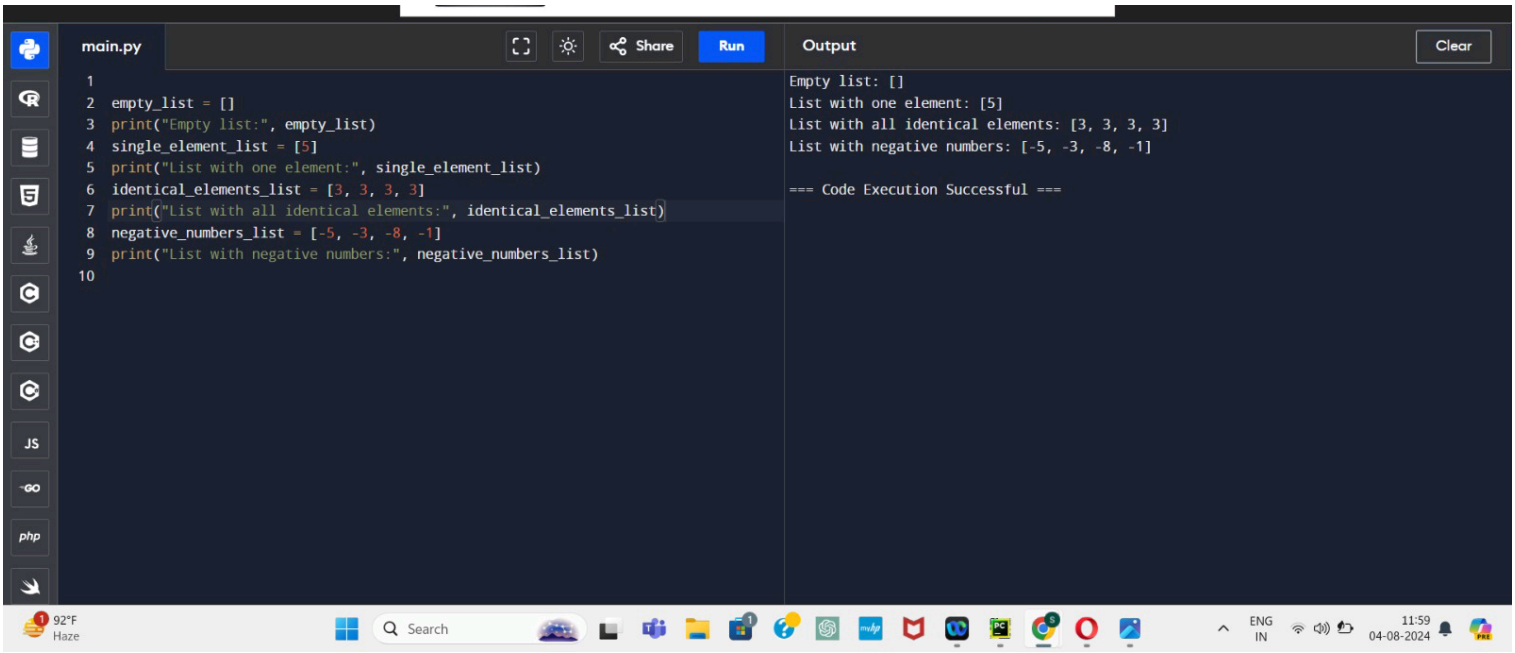
Clear

```
Sorting a Random Array:
Original array: [5, 2, 9, 1, 5, 6]
Pass 1: [1, 2, 9, 5, 5, 6]
Pass 2: [1, 2, 9, 5, 5, 6]
Pass 3: [1, 2, 5, 9, 5, 6]
Pass 4: [1, 2, 5, 5, 9, 6]
Pass 5: [1, 2, 5, 5, 6, 9]
Pass 6: [1, 2, 5, 5, 6, 9]
Sorted array: [1, 2, 5, 5, 6, 9]


Sorting a Reverse Sorted Array:
Original array: [10, 8, 6, 4, 2]
Pass 1: [2, 8, 6, 4, 10]
Pass 2: [2, 4, 6, 8, 10]
Pass 3: [2, 4, 6, 8, 10]
Pass 4: [2, 4, 6, 8, 10]
Pass 5: [2, 4, 6, 8, 10]
Sorted array: [2, 4, 6, 8, 10]



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


main.py
Share
Run
Output
Clear
1
2 empty_list = []
3 print("Empty list:", empty_list)
4 single_element_list = [5]
5 print("List with one element:", single_element_list)
6 identical_elements_list = [3, 3, 3, 3]
7 print("List with all identical elements:", identical_elements_list)
8 negative_numbers_list = [-5, -3, -8, -1]
9 print("List with negative numbers:", negative_numbers_list)
10
Empty list: []
List with one element: [5]
List with all identical elements: [3, 3, 3, 3]
List with negative numbers: [-5, -3, -8, -1]
=== Code Execution Successful ===
JS
-GO
php
92°F
Haze
Search
ENG
IN
11:59
04-08-2024