1. 132. Sort an array of integers using the bubble sort technique. Analyze its time complexity using Big-O notation. Write the code

PROGRAM:-

```
def bubble_sort(nums):
    n = len(nums)
    for i in range(n):
        # Track whether any swap is made in this pass
        swapped = False
        for j in range(0, n-i-1):
            if nums[j] > nums[j+1]:
                nums[j], nums[j+1] = nums[j+1], nums[j]
                swapped = True
        # If no swap is made, the list is already sorted
        if not swapped:
            break
    return nums


# Test Cases
# Test Case 1
input1 = [64, 34, 25, 12, 22, 11, 90]
print(f"Input: {input1}\nSorted: {bubble_sort(input1)}\n")


# Test Case 2
input2 = [5, 1, 4, 2, 8]
print(f"Input: {input2}\nSorted: {bubble_sort(input2)}\n")


# Test Case 3
```

input3 = [3, 7, 3, 5, 2, 5, 9, 2]

print(f"Input: {input3}\nSorted: {bubble_sort(input3)}\n")

OUTPUT:-

```
Input: [64, 34, 25, 12, 22, 11, 90]
Sorted: [11, 12, 22, 25, 34, 64, 90]

Input: [5, 1, 4, 2, 8]
Sorted: [1, 2, 4, 5, 8]

Input: [3, 7, 3, 5, 2, 5, 9, 2]
Sorted: [2, 2, 3, 3, 5, 5, 7, 9]


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

TIME COMPLEXITY:-O(n^2)