Exp-2.4

Title:

Insertion Sort Algorithm Handling Duplicate Elements

Aim:

To implement Insertion Sort to correctly sort arrays containing duplicates by maintaining the relative order of equal elements (stable sort).

Procedure:

1. Read input size n and array elements (including duplicates).
2. For each element from the second onward, compare it with elements before it.
3. Shift all larger elements one position to the right to make space.
4. Insert the current element in the correct position without changing relative order of duplicates.
5. Continue until the entire array is sorted.
6. Print the sorted array.

Algorithm:

1. Start
2. For i from 1 to n-1:
    - Let key = arr[i]
    - Initialize j = i - 1
    - While j >= 0 and arr[j] > key:
        - Shift arr[j] to arr[j + 1]
        - Decrement j
    - Insert key at arr[j + 1]
3. Repeat until all elements are inserted
4. End.

Input:

10

3 1 4 1 5 9 2 6 5 3

5

5 5 5 5 5

8

2 3 1 3 2 1 1 3

Output:

1 1 2 3 3 4 5 5 6 9

5 5 5 5 5

1 1 1 2 2 3 3 3

Program:

```
def insertionSort(arr):
    n = len(arr)
    for i in range(1, n):
        key = arr[i]
        j = i - 1
        # Shift elements greater than key rightwards
        while j >= 0 and arr[j] > key:
            arr[j + 1] = arr[j]
            j -= 1
        # Insert key at correct position
        arr[j + 1] = key
    return arr
```

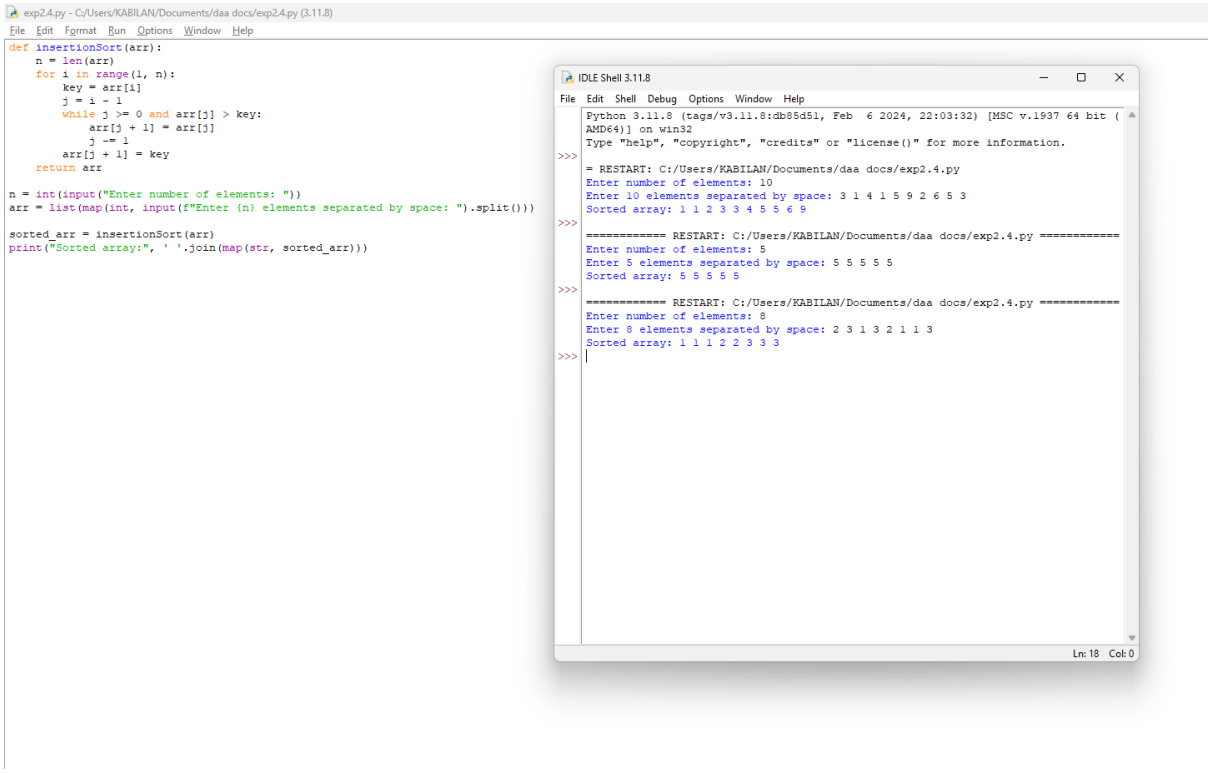
```
n = int(input("Enter number of elements: "))
arr = list(map(int, input(f"Enter {n} elements separated by space: ").split()))

sorted_arr = insertionSort(arr)
print("Sorted array:", ' '.join(map(str, sorted_arr)))
```

Performance Analysis:

Time Complexity: $O(n^2)$ or $O(n)$

Space Complexity: $O(1)$

Program Output:



Result:

Thus the given program Insertion Sort with Duplicates is executed and got output successfully.