10.Given a list of item weights and a maximum capacity for each container, determine the minimum number of containers required to load all items using a greedy approach. The greedy approach should prioritize loading items into the current container until it is full before moving to the next container.

Test Case 1:

Input:

n = 7

weights = [5, 10, 15, 20, 25, 30, 35]

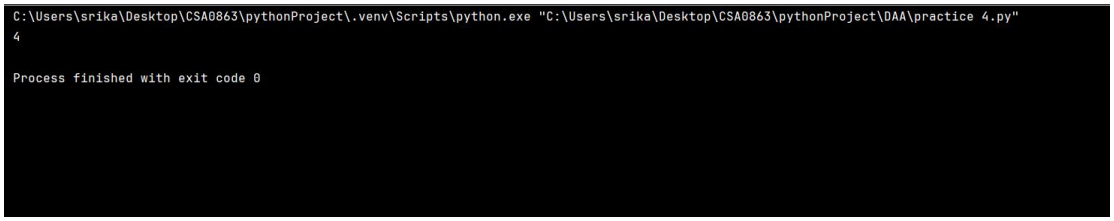
max_capacity = 50

Output: 4

Program:

```
def min_containers(weights, max_capacity):
    weights.sort(reverse=True)
    num_containers = 0
    current_capacity = max_capacity
    for weight in weights:
        if weight <= current_capacity:
            current_capacity -= weight
        else:
            num_containers += 1
            current_capacity = max_capacity - weight
    if current_capacity < max_capacity:
        num_containers += 1
    return num_containers
n = 7
weights = [5, 10, 15, 20, 25, 30, 35]
max_capacity = 50
print(min_containers(weights, max_capacity))
```

Output:



Time complexity:

O(nlogn)