203) 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
Test Case 2:
Input:
n = 8
weights = [10, 20, 30, 40, 50, 60, 70, 80]
max_capacity = 100
Output: 6

AIM: To write a python program to the greedy approach should prioritize loading items into the current container until it is full before moving to the next container.

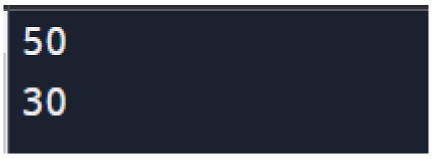
PROGRAM:
```
def max_weight_greedy(weights, max_capacity):
    # Sort weights in descending order
    weights.sort(reverse=True)

    current_weight = 0

    for weight in weights:
        if current_weight + weight <= max_capacity:
            current_weight += weight
        else:
            break

    return current_weight

n1 = 5
weights1 = [10, 20, 30, 40, 50]
max_capacity1 = 60
print(max_weight_greedy(weights1, max_capacity1))
```

OUTPUT: 

```
50
30
```

TIME COMPLEXITY: O(n logn)