5.9 MAXIMUM WEIGHT LOADING USING GREEDY APPROACH

Question:

Given a list of item weights and the maximum capacity of a container, determine the maximum weight that can be loaded into the container using a greedy approach. The greedy approach should prioritize loading heavier items first until the container reaches its capacity.

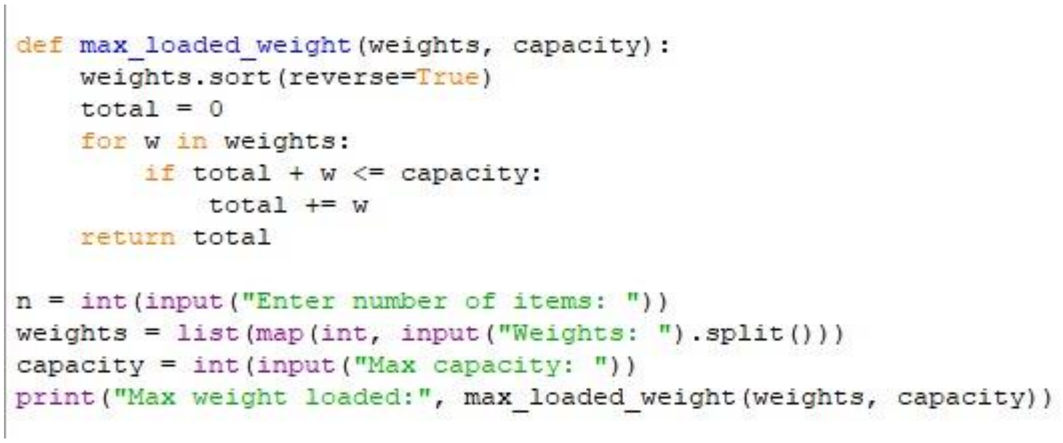
AIM

To implement a greedy algorithm that selects the heaviest items first and loads them into the container without exceeding its capacity.

ALGORITHM

1. Sort the list of item weights in **descending order**.
2. Initialize total_weight = 0.
3. Iterate through the sorted weights:
    - If total_weight + weight ≤ capacity, add the item to the container.
    - Otherwise, skip the item.
4. Return total_weight as the maximum weight loaded.

PROGRAM



```python
def max_loaded_weight(weights, capacity):
    weights.sort(reverse=True)
    total = 0
    for w in weights:
        if total + w <= capacity:
            total += w
    return total

n = int(input("Enter number of items: "))
weights = list(map(int, input("Weights: ").split()))
capacity = int(input("Max capacity: "))
print("Max weight loaded:", max_loaded_weight(weights, capacity))
```

Input:

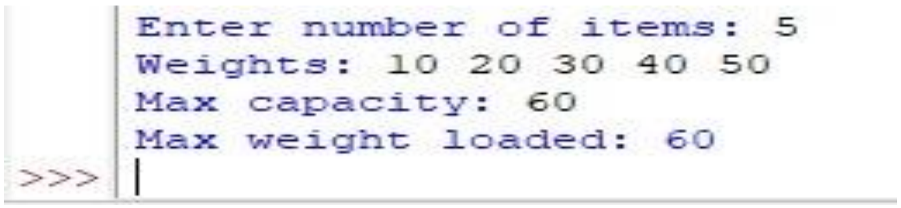
Enter number of items: 5

Weights: 10 20 30 40 50

Max capacity: 60

Max weight loaded: 60

Output:



```
Enter number of items: 5
Weights: 10 20 30 40 50
Max capacity: 60
Max weight loaded: 60
>>> |
```

RESULT:

Thus the program is successfully executed and the output is verified.

PERFORMANCE ANALYSIS:

· Time Complexity: O(n log n), due to sorting

· Space Complexity: O(1), constant extra space