TOPIC 5.10 : GREEDY BIN PACKING FOR MINIMUM CONTAINERS

Problem Statement  
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 program that determines the minimum number of containers required to load all items without exceeding the maximum capacity of each container, using a greedy strategy.

Algorithm

1. Start
2. Sort the list of weights in descending order
3. Initialize count = 0, current\_capacity = 0
4. For each item in weights:
   * If current\_capacity + weight ≤ max\_capacity, load it into the current container
   * Otherwise, start a new container, increment count, and set current\_capacity = weight
5. After processing all items, increment count for the last container
6. Return count
7. Stop

Input and Output  
A screenshot of a computer

AI-generated content may be incorrect.

Result  
The program successfully computes the minimum number of containers required to load all items using the greedy approach.

Performance Analysis  
Time Complexity: O(n log n) due to sorting  
Space Complexity: O(1) for counters