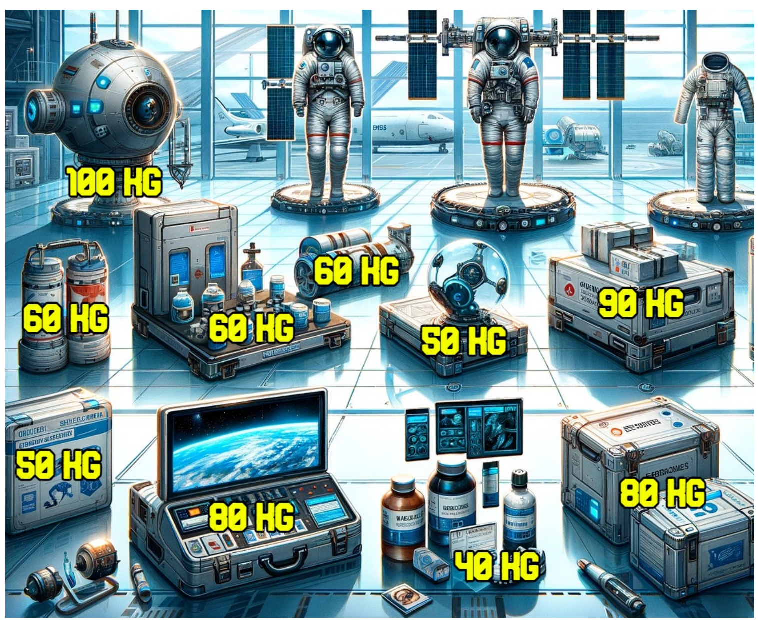
Quiz 2: Optimizing Resource Delivery to the ISS

Topics: Dynamic Programming - **A MUST use starter code**

Problem Statement

Following the successful mission of Türkiye's first astronaut, the nation has committed to further contributing to international space exploration efforts. A series of missions are planned to deliver essential resources to the International Space Station (ISS). As part of the prestigious CENG department at Hacettepe University, you have been selected by the Turkish Space Agency to optimize these critical resource delivery missions. The challenge is multifaceted: **Given a set of resources with varying mass and a spacecraft that can carry at most M kilograms, determine the most efficient way to load the spacecraft. The goal is to maximize the mass of resources delivered to the ISS while adhering to the spacecraft's capacity. Each resource is unique and must be loaded in its entirety or not at all (i.e., resources cannot be divided into lighter subparts)**.



One might consider simply loading the heaviest resources first until the spacecraft's capacity is reached. However, this greedy approach may not always yield the most effective solution. Imagine a scenario with resources of different masses as illustrated on the right and a spacecraft with a capacity of 200 kg.

The greedy strategy discussed above would result in the selection of items of weights 100 and 90 for a total of 190 kg of resources being sent to the ISS, by prioritizing the two heaviest items, while neglecting a combination of smaller, yet collectively heavier resources that could better utilize the available capacity. A better solution with three items of weights 40, 60, and 100 kg is optimal as it would use the full spacecraft capacity of 200 kg. You will use a Dynamic Programming approach to optimally solve this problem.

Hints for Dynamic Programming Approach

This problem requires the division of the main problem into smaller, manageable subproblems, which will prevent redundant calculations by memorizing intermediate results. Instead of confronting the entire problem directly, consider sequentially whether to include each resource in the spacecraft, aiming to fully utilize its capacity of M kilograms.

If it is feasible to fully load the spacecraft with the given n resources' masses denoted as $m_0, ..., m_{n-1}$, there exists a set of resources $R \subseteq \{m_0, ..., m_{n-1}\}$ that optimizes the use of space. This leads to two considerations:

- **Case 1**: If resource n is NOT included in the optimal loading solution, i.e., $m_{n-1} \notin R$, then it is possible to achieve optimal loading without the heaviest remaining resource. That is, we can then proceed to solve the same problem for $n-1$ remaining items, thus effectively reducing the problem size.

- **Case 2**: If resource n IS included in the optimal loading solution, i.e., $m_{n-1} \in R$, putting this resource aside suggests that the remaining resources can optimally fill the capacity of $M - m_{n-1}$, the spacecraft's capacity minus the mass of resource n denoted as m_{n-1}. That is, we can then proceed to solve the same problem for $M - m_{n-1}$ capacity with the remaining $n-1$ items, thus again effectively reducing the problem size to a smaller subproblem.

This approach of breaking down the problem and leveraging memorization continues until all resources have been considered. To attack this problem, we suggest you define $L(m, i)$ as a Boolean function indicating whether a spacecraft can be optimally loaded with a mass of m kilograms using the first i resources. This function is described as follows:

$$L(m,i) = \left\{ \begin{array}{ll} \texttt{TRUE} & \textbf{if } i = 0 \textbf{ and } m = 0 \\ \texttt{FALSE} & \textbf{if } i = 0 \textbf{ and } m > 0 \\ L(m, i-1) & \textbf{if } i > 0 \textbf{ and } m_{i-1} > m \\ L(m, i-1) \textbf{ or } L(m - m_{i-1}, i-1) & \textbf{otherwise} \end{array} \right\} \tag{1}$$

Input Format

The input file, which will be given as the *first command-line argument*, will consist of two lines. The first line of the input contains an integer M (capacity of the spacecraft in kg) and the number n of resources like scientific equipment, tools, food, and medical supplies. The second line contains n integers $m_0, ..., m_{n-1}$ separated by spaces, which represent the resources' masses in kg.

Sample Input:

```
200 10
40 50 50 60 60 60 100 90 80 80
```

Output Format

On the first line, output the maximum mass of the subset of given resources that can be loaded into the spacecraft without exceeding its capacity M. Below that, output the final $(M+1) \times (n+1)$-sized $L(m,i)$ matrix, where $0 \leq m \leq M$, $0 \leq i \leq n$, such that each row represents the weight starting from 0, and each column represents the resource index starting from 0. If $L(m,i)$ is `TRUE`, output `1`, and `0`, otherwise. Do not put any spaces between the digits. Print to the STDOUT.

```
m_max
L(0,0)L(0,1)...L(0,n)
L(1,0)L(1,1)...L(1,n)
...
L(M,0)L(M,1)...L(M,n)
```

Sample Output:

```
200
11111111111
00000000000
... (198 more lines here)
00001111111
```

Check `sample_output_1.txt` in the starter code for the full sample output format.

Constraints

$1 \leq M \leq 10^4$; $1 \leq n \leq 300$; $0 \leq m_0, ..., m_{n-1} \leq 10^5$

Important Rules

- **You MUST use this starter code. Do not change any part of the given starter codes, those are given to ensure that you pass the unit tests in autograding. Only complete the TODOs.**
- Test your codes using the Tur^{6}Bo Grader and finally submit them via submit.cs.hacettepe.edu.tr using the same format given below:
  - **<studentID>.zip**
    - **Quiz2.java**

Academic Integrity Policy

All work **must be done individually**. You are encouraged to discuss the given problems with your classmates, but these discussions should be carried out in an abstract way. That is, discussions related to a particular solution to a specific problem (either in actual code or in pseudocode) **will not be tolerated**. In short, turning in someone else's work (including work available on the internet, or generated by the AI tools), in whole or in part, as your own will be considered as **a violation of academic integrity**. Please note that the former condition also holds for the material found on the web as everything on the web has been written by someone else.

The submissions will be subjected to a similarity check. Any submissions that fail the similarity check will not be graded and will be reported to the ethics committee as a case of academic integrity violation, which may result in the suspension of the involved students.