Problem 879: Profitable Schemes

Problem Information

Difficulty: Hard
Acceptance Rate: 48.13%
Paid Only: No
Tags: Array, Dynamic Programming

Problem Description

There is a group of `n` members, and a list of various crimes they could commit. The `ith` crime generates a `profit[i]` and requires `group[i]` members to participate in it. If a member participates in one crime, that member can't participate in another crime.

Let's call a **profitable scheme** any subset of these crimes that generates at least `minProfit` profit, and the total number of members participating in that subset of crimes is at most `n`.

Return the number of schemes that can be chosen. Since the answer may be very large, **return it modulo** `109 + 7`.

Example 1:

Input: n = 5, minProfit = 3, group = [2,2], profit = [2,3] **Output:** 2 **Explanation:** To make a profit of at least 3, the group could either commit crimes 0 and 1, or just crime 1. In total, there are 2 schemes.

Example 2:

Input: n = 10, minProfit = 5, group = [2,3,5], profit = [6,7,8] **Output:** 7 **Explanation:** To make a profit of at least 5, the group could commit any crimes, as long as they commit one. There are 7 possible schemes: (0), (1), (2), (0,1), (0,2), (1,2), and (0,1,2).

Constraints:

* `1 <= n <= 100` * `0 <= minProfit <= 100` * `1 <= group.length <= 100` * `1 <= group[i] <= 100` * `profit.length == group.length` * `0 <= profit[i] <= 100`

Code Snippets

C++:

```
class Solution {
public:
int profitableSchemes(int n, int minProfit, vector<int>& group, vector<int>&
profit) {

}
};
```

Java:

```
class Solution {
public int profitableSchemes(int n, int minProfit, int[] group, int[] profit)
{

}
}
```

Python3:

```
class Solution:
def profitableSchemes(self, n: int, minProfit: int, group: List[int], profit:
List[int]) -> int:
```