Problem 1815: Maximum Number of Groups Getting Fresh Donuts

Problem Information

Difficulty: Hard
Acceptance Rate: 41.13%
Paid Only: No
Tags: Array, Dynamic Programming, Bit Manipulation, Memoization, Bitmask

Problem Description

There is a donuts shop that bakes donuts in batches of `batchSize`. They have a rule where they must serve **all** of the donuts of a batch before serving any donuts of the next batch. You are given an integer `batchSize` and an integer array `groups`, where `groups[i]` denotes that there is a group of `groups[i]` customers that will visit the shop. Each customer will get exactly one donut.

When a group visits the shop, all customers of the group must be served before serving any of the following groups. A group will be happy if they all get fresh donuts. That is, the first customer of the group does not receive a donut that was left over from the previous group.

You can freely rearrange the ordering of the groups. Return _the**maximum** possible number of happy groups after rearranging the groups._

Example 1:

Input: batchSize = 3, groups = [1,2,3,4,5,6] **Output:** 4 **Explanation:** You can arrange the groups as [6,2,4,5,1,3]. Then the 1st, 2nd, 4th, and 6th groups will be happy.

Example 2:

Input: batchSize = 4, groups = [1,3,2,5,2,2,1,6] **Output:** 4

Constraints:

* `1 <= batchSize <= 9` * `1 <= groups.length <= 30` * `1 <= groups[i] <= 109`

Code Snippets

C++:

```
class Solution {
public:
int maxHappyGroups(int batchSize, vector<int>& groups) {

}
};
```

Java:

```
class Solution {
public int maxHappyGroups(int batchSize, int[] groups) {

}
}
```

Python3:

```
class Solution:
def maxHappyGroups(self, batchSize: int, groups: List[int]) -> int:
```