Patching Array

Given a sorted integer array `nums` and an integer `n`, add/patch elements to the array such that any number in the range `[1, n]` inclusive can be formed by the sum of some elements in the array.

Return *the minimum number of patches required*.

Example 1:

```
Input: nums = [1,3], n = 6
Output: 1
Explanation:
Combinations of nums are [1], [3], [1,3], which form possible sums of: 1, 3, 4.
Now if we add/patch 2 to nums, the combinations are: [1], [2], [3], [1,3], [2,3], [1,2,3].
Possible sums are 1, 2, 3, 4, 5, 6, which now covers the range [1, 6].
So we only need 1 patch.
```

Example 2:

```
Input: nums = [1,5,10], n = 20
Output: 2
Explanation: The two patches can be [2, 4].
```

Example 3:

```
Input: nums = [1,2,2], n = 5
Output: 0
```

Constraints:

- `1 <= nums.length <= 1000`
- `1 <= nums[i] <=` 10^4
- `nums` is sorted in **ascending order**.
- `1 <= n <=` 2^{31} `- 1`

Go

```
func minPatches(nums []int, n int) int {
    generated := make(map[int]bool)
    iterate(nums, 0, &generated)

```