948. Bag of Tokens

Solved

Medium | Topics | Companies

You start with an initial **power** of `power`, an initial **score** of `0`, and a bag of tokens given as an integer array `tokens`, where each `tokens[i]` donates the value of token$_i$.

Your goal is to **maximize** the total **score** by strategically playing these tokens. In one move, you can play an **unplayed** token in one of the two ways (but not both for the same token):

- **Face-up**: If your current power is **at least** `tokens[i]`, you may play token$_i$, losing `tokens[i]` power and gaining `1` score.
- **Face-down**: If your current score is **at least** `1`, you may play token$_i$, gaining `tokens[i]` power and losing `1` score.

Return *the **maximum** possible score you can achieve after playing **any** number of tokens*.

Example 1:

> **Input:** `tokens = [100], power = 50`
>
> **Output:** `0`
>
> **Explanation:** Since your score is `0` initially, you cannot play the token face-down. You also cannot play it face-up since your power (`50`) is less than `tokens[0]` (`100`).

```
class Solution {
    func bagOfTokensScore(_ tokens: [Int], _ power: Int) -> Int {
        var pow = power
        var count = 0
        var ans = count
        var l = 0
        var r = tokens.count - 1

        let to = tokens.sorted()
        while(l<=r){
            if(to[l] <= pow ) {
                pow = pow - to[l]
                count = count + 1
                ans = max(ans,count)
                l = l + 1
            } else if(count > 0) {
```

```
                count = count - 1
                ans = max(ans,count)
                pow = pow + to[r]
                r = r - 1
            } else {
                break
            }
        }

        return ans
    }
}
```