**[Bag of Tokens](https://leetcode.com/problems/bag-of-tokens/)**

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).

**Example 2:**

**Input:** tokens = [200,100], power = 150

**Output:** 1

**Explanation:** Play token*1* (100) face-up, reducing your power to 50 and increasing your score to 1.

There is no need to play token*0*, since you cannot play it face-up to add to your score. The maximum score achievable is 1.

**Example 3:**

**Input:** tokens = [100,200,300,400], power = 200

**Output:** 2

**Explanation:** Play the tokens in this order to get a score of 2:

1. Play token*0* (100) face-up, reducing power to 100 and increasing score to 1.
2. Play token*3* (400) face-down, increasing power to 500 and reducing score to 0.
3. Play token*1* (200) face-up, reducing power to 300 and increasing score to 1.
4. Play token*2* (300) face-up, reducing power to 0 and increasing score to 2.

The maximum score achievable is 2.

**Constraints:**

* 0 <= tokens.length <= 1000
* 0 <= tokens[i], power < 104

class Solution {

public:

    int bagOfTokensScore(vector<int>& tokens, int power) {

        sort(tokens.begin(), tokens.end());

        int n = tokens.size();

        int score = 0;

        int max\_score = 0;

        int left = 0;

        int right = n - 1;

        while (left <= right) {

            if (power >= tokens[left]) {

                power -= tokens[left];

                score += 1;

                left += 1;

                max\_score = max(max\_score, score);

            } else if (score > 0) {

                power += tokens[right];

                score -= 1;

                right -= 1;

            } else {

                break;

            }

        }

        return max\_score;

    }

};

<https://leetcode.com/problems/bag-of-tokens/?envType=daily-question&envId=2024-03-04>