Problem 502: IPO

Problem Information

Difficulty: Hard
Acceptance Rate: 53.19%
Paid Only: No
Tags: Array, Greedy, Sorting, Heap (Priority Queue)

Problem Description

Suppose LeetCode will start its **IPO** soon. In order to sell a good price of its shares to Venture Capital, LeetCode would like to work on some projects to increase its capital before the **IPO**. Since it has limited resources, it can only finish at most `k` distinct projects before the **IPO**. Help LeetCode design the best way to maximize its total capital after finishing at most `k` distinct projects.

You are given `n` projects where the `ith` project has a pure profit `profits[i]` and a minimum capital of `capital[i]` is needed to start it.

Initially, you have `w` capital. When you finish a project, you will obtain its pure profit and the profit will be added to your total capital.

Pick a list of **at most** `k` distinct projects from given projects to **maximize your final capital** , and return _the final maximized capital_.

The answer is guaranteed to fit in a 32-bit signed integer.

Example 1:

Input: k = 2, w = 0, profits = [1,2,3], capital = [0,1,1] **Output:** 4 **Explanation:** Since your initial capital is 0, you can only start the project indexed 0. After finishing it you will obtain profit 1 and your capital becomes 1. With capital 1, you can either start the project indexed 1 or the project indexed 2. Since you can choose at most 2 projects, you need to finish the project indexed 2 to get the maximum capital. Therefore, output the final maximized capital, which is 0 + 1 + 3 = 4.

Example 2:

Input: k = 3, w = 0, profits = [1,2,3], capital = [0,1,2] **Output:** 6

Constraints:

* `1 <= k <= 105` * `0 <= w <= 109` * `n == profits.length` * `n == capital.length` * `1 <= n <= 105` * `0 <= profits[i] <= 104` * `0 <= capital[i] <= 109`

Code Snippets

C++:

```
class Solution {
public:
int findMaximizedCapital(int k, int w, vector<int>& profits, vector<int>&
capital) {

}
};
```

Java:

```
class Solution {
public int findMaximizedCapital(int k, int w, int[] profits, int[] capital) {

}
}
```

Python3:

```
class Solution:
def findMaximizedCapital(self, k: int, w: int, profits: List[int], capital:
List[int]) -> int:
```