You are given two **0-indexed** integer arrays servers and tasks of lengths n​​​​​​ and m​​​​​​ respectively. servers[i] is the **weight** of the i​​​​​​th​​​​ server, and tasks[j] is the **time needed** to process the j​​​​​​th​​​​ task **in seconds**.

Tasks are assigned to the servers using a **task queue**. Initially, all servers are free, and the queue is **empty**.

At second j, the jth task is **inserted** into the queue (starting with the 0th task being inserted at second 0). As long as there are free servers and the queue is not empty, the task in the front of the queue will be assigned to a free server with the **smallest weight**, and in case of a tie, it is assigned to a free server with the **smallest index**.

If there are no free servers and the queue is not empty, we wait until a server becomes free and immediately assign the next task. If multiple servers become free at the same time, then multiple tasks from the queue will be assigned **in order of insertion** following the weight and index priorities above.

A server that is assigned task j at second t will be free again at second t + tasks[j].

Build an array ans​​​​ of length m, where ans[j] is the **index** of the server the j​​​​​​th task will be assigned to.

Return *the array* ans​​​​.

**Example 1:**

Input: servers = [3,3,2], tasks = [1,2,3,2,1,2]  
Output: [2,2,0,2,1,2]  
Explanation: Events in chronological order go as follows:  
- At second 0, task 0 is added and processed using server 2 until second 1.  
- At second 1, server 2 becomes free. Task 1 is added and processed using server 2 until second 3.  
- At second 2, task 2 is added and processed using server 0 until second 5.  
- At second 3, server 2 becomes free. Task 3 is added and processed using server 2 until second 5.  
- At second 4, task 4 is added and processed using server 1 until second 5.  
- At second 5, all servers become free. Task 5 is added and processed using server 2 until second 7.

**Example 2:**

Input: servers = [5,1,4,3,2], tasks = [2,1,2,4,5,2,1]  
Output: [1,4,1,4,1,3,2]  
Explanation: Events in chronological order go as follows:   
- At second 0, task 0 is added and processed using server 1 until second 2.  
- At second 1, task 1 is added and processed using server 4 until second 2.  
- At second 2, servers 1 and 4 become free. Task 2 is added and processed using server 1 until second 4.   
- At second 3, task 3 is added and processed using server 4 until second 7.  
- At second 4, server 1 becomes free. Task 4 is added and processed using server 1 until second 9.   
- At second 5, task 5 is added and processed using server 3 until second 7.  
- At second 6, task 6 is added and processed using server 2 until second 7.

**Constraints:**

* servers.length == n
* tasks.length == m
* 1 <= n, m <= 2 \* 105
* 1 <= servers[i], tasks[j] <= 2 \* 105