Problem 1889: Minimum Space Wasted From Packaging

Problem Information

Difficulty: Hard
Acceptance Rate: 33.14%
Paid Only: No
Tags: Array, Binary Search, Sorting, Prefix Sum

Problem Description

You have `n` packages that you are trying to place in boxes, **one package in each box**. There are `m` suppliers that each produce boxes of **different sizes** (with infinite supply). A package can be placed in a box if the size of the package is **less than or equal to** the size of the box.

The package sizes are given as an integer array `packages`, where `packages[i]` is the **size** of the `ith` package. The suppliers are given as a 2D integer array `boxes`, where `boxes[j]` is an array of **box sizes** that the `jth` supplier produces.

You want to choose a **single supplier** and use boxes from them such that the **total wasted space** is **minimized**. For each package in a box, we define the space **wasted** to be `size of the box - size of the package`. The **total wasted space** is the sum of the space wasted in **all** the boxes.

* For example, if you have to fit packages with sizes `[2,3,5]` and the supplier offers boxes of sizes `[4,8]`, you can fit the packages of size-`2` and size-`3` into two boxes of size-`4` and the package with size-`5` into a box of size-`8`. This would result in a waste of `(4-2) + (4-3) + (8-5) = 6`.

Return _the**minimum total wasted space** by choosing the box supplier **optimally** , or _`-1` _if it is**impossible** to fit all the packages inside boxes. _Since the answer may be **large** , return it **modulo**`109 + 7`.

Example 1:

Input: packages = [2,3,5], boxes = [[4,8],[2,8]] **Output:** 6 **Explanation** : It is optimal to choose the first supplier, using two size-4 boxes and one size-8 box. The total waste is (4-2) + (4-3) + (8-5) = 6.

Example 2:

Input: packages = [2,3,5], boxes = [[1,4],[2,3],[3,4]] **Output:** -1 **Explanation:** There is no box that the package of size 5 can fit in.

Example 3:

Input: packages = [3,5,8,10,11,12], boxes = [[12],[11,9],[10,5,14]] **Output:** 9 **Explanation:** It is optimal to choose the third supplier, using two size-5 boxes, two size-10 boxes, and two size-14 boxes. The total waste is (5-3) + (5-5) + (10-8) + (10-10) + (14-11) + (14-12) = 9.

Constraints:

* `n == packages.length` * `m == boxes.length` * `1 <= n <= 105` * `1 <= m <= 105` * `1 <= packages[i] <= 105` * `1 <= boxes[j].length <= 105` * `1 <= boxes[j][k] <= 105` * `sum(boxes[j].length) <= 105` * The elements in `boxes[j]` are **distinct**.

Code Snippets

C++:

```
class Solution {
public:
int minWastedSpace(vector<int>& packages, vector<vector<int>>& boxes) {

}
};
```

Java:

```
class Solution {
public int minWastedSpace(int[] packages, int[][] boxes) {

}
}
```

Python3:

```
class Solution:
def minWastedSpace(self, packages: List[int], boxes: List[List[int]]) -> int:
```