Problem 1626: Best Team With No Conflicts

Problem Information

Difficulty: Medium
Acceptance Rate: 50.53%
Paid Only: No
Tags: Array, Dynamic Programming, Sorting

Problem Description

You are the manager of a basketball team. For the upcoming tournament, you want to choose the team with the highest overall score. The score of the team is the **sum** of scores of all the players in the team.

However, the basketball team is not allowed to have **conflicts**. A **conflict** exists if a younger player has a **strictly higher** score than an older player. A conflict does **not** occur between players of the same age.

Given two lists, `scores` and `ages`, where each `scores[i]` and `ages[i]` represents the score and age of the `ith` player, respectively, return _the highest overall score of all possible basketball teams_.

Example 1:

Input: scores = [1,3,5,10,15], ages = [1,2,3,4,5] **Output:** 34 **Explanation:** You can choose all the players.

Example 2:

Input: scores = [4,5,6,5], ages = [2,1,2,1] **Output:** 16 **Explanation:** It is best to choose the last 3 players. Notice that you are allowed to choose multiple people of the same age.

Example 3:

Input: scores = [1,2,3,5], ages = [8,9,10,1] **Output:** 6 **Explanation:** It is best to choose the first 3 players.

Constraints:

* `1 <= scores.length, ages.length <= 1000` * `scores.length == ages.length` * `1 <= scores[i] <= 106` * `1 <= ages[i] <= 1000`

Code Snippets

C++:

```
class Solution {
public:
int bestTeamScore(vector<int>& scores, vector<int>& ages) {

}
};
```

Java:

```
class Solution {
public int bestTeamScore(int[] scores, int[] ages) {

}
}
```

Python3:

```
class Solution:
def bestTeamScore(self, scores: List[int], ages: List[int]) -> int:
```