Problem 983: Minimum Cost For Tickets

Problem Information

Difficulty: Medium
Acceptance Rate: 67.39%
Paid Only: No
Tags: Array, Dynamic Programming

Problem Description

You have planned some train traveling one year in advance. The days of the year in which you will travel are given as an integer array `days`. Each day is an integer from `1` to `365`.

Train tickets are sold in **three different ways** :

* a **1-day** pass is sold for `costs[0]` dollars, * a **7-day** pass is sold for `costs[1]` dollars, and * a **30-day** pass is sold for `costs[2]` dollars.

The passes allow that many days of consecutive travel.

* For example, if we get a **7-day** pass on day `2`, then we can travel for `7` days: `2`, `3`, `4`, `5`, `6`, `7`, and `8`.

Return _the minimum number of dollars you need to travel every day in the given list of days_.

Example 1:

Input: days = [1,4,6,7,8,20], costs = [2,7,15] **Output:** 11 **Explanation:** For example, here is one way to buy passes that lets you travel your travel plan: On day 1, you bought a 1-day pass for costs[0] = $2, which covered day 1. On day 3, you bought a 7-day pass for costs[1] = $7, which covered days 3, 4, ..., 9. On day 20, you bought a 1-day pass for costs[0] = $2, which covered day 20. In total, you spent $11 and covered all the days of your travel.

Example 2:

Input: days = [1,2,3,4,5,6,7,8,9,10,30,31], costs = [2,7,15] **Output:** 17 **Explanation:** For example, here is one way to buy passes that lets you travel your travel plan: On day 1, you bought a 30-day pass for costs[2] = $15 which covered days 1, 2, ..., 30. On day 31, you bought a 1-day pass for costs[0] = $2 which covered day 31. In total, you spent $17 and covered all the days of your travel.

Constraints:

* `1 <= days.length <= 365` * `1 <= days[i] <= 365` * `days` is in strictly increasing order. * `costs.length == 3` * `1 <= costs[i] <= 1000`

Code Snippets

C++:

```
class Solution {
public:
int mincostTickets(vector<int>& days, vector<int>& costs) {

}
};
```

Java:

```
class Solution {
public int mincostTickets(int[] days, int[] costs) {

}
}
```

Python3:

```
class Solution:
def mincostTickets(self, days: List[int], costs: List[int]) -> int:
```