Problem 2431: Maximize Total Tastiness of Purchased Fruits

Problem Information

Difficulty: Medium
Acceptance Rate: 64.69%
Paid Only: Yes
Tags: Array, Dynamic Programming

Problem Description

You are given two non-negative integer arrays `price` and `tastiness`, both arrays have the same length `n`. You are also given two non-negative integers `maxAmount` and `maxCoupons`.

For every integer `i` in range `[0, n - 1]`:

* `price[i]` describes the price of `ith` fruit. * `tastiness[i]` describes the tastiness of `ith` fruit.

You want to purchase some fruits such that total tastiness is maximized and the total price does not exceed `maxAmount`.

Additionally, you can use a coupon to purchase fruit for **half of its price** (rounded down to the closest integer). You can use at most `maxCoupons` of such coupons.

Return _the maximum total tastiness that can be purchased_.

Note that:

* You can purchase each fruit at most once. * You can use coupons on some fruit at most once.

Example 1:

Input: price = [10,20,20], tastiness = [5,8,8], maxAmount = 20, maxCoupons = 1 **Output:** 13 **Explanation:** It is possible to make total tastiness 13 in following way: - Buy first fruit without coupon, so that total price = 0 + 10 and total tastiness = 0 + 5. - Buy second fruit with coupon, so that total price = 10 + 10 and total tastiness = 5 + 8. - Do not buy third fruit, so that total price = 20 and total tastiness = 13. It can be proven that 13 is the maximum total tastiness that can be obtained.

Example 2:

Input: price = [10,15,7], tastiness = [5,8,20], maxAmount = 10, maxCoupons = 2 **Output:** 28 **Explanation:** It is possible to make total tastiness 20 in following way: - Do not buy first fruit, so that total price = 0 and total tastiness = 0. - Buy second fruit with coupon, so that total price = 0 + 7 and total tastiness = 0 + 8. - Buy third fruit with coupon, so that total price = 7 + 3 and total tastiness = 8 + 20. It can be proven that 28 is the maximum total tastiness that can be obtained.

Constraints:

* `n == price.length == tastiness.length` * `1 <= n <= 100` * `0 <= price[i], tastiness[i], maxAmount <= 1000` * `0 <= maxCoupons <= 5`

Code Snippets

C++:

```
class Solution {
public:
int maxTastiness(vector<int>& price, vector<int>& tastiness, int maxAmount,
int maxCoupons) {

}
};
```

Java:

```
class Solution {
public int maxTastiness(int[] price, int[] tastiness, int maxAmount, int
maxCoupons) {

}
}
```

Python3:

```
class Solution:
def maxTastiness(self, price: List[int], tastiness: List[int], maxAmount:
int, maxCoupons: int) -> int:
```