PROBLEM

Maximize dot product

Easy Accuracy: 13.27% Submissions: 25K+ Points: 2

Given two arrays **a** and **b** of positive integers of size **n** and **m** where **n >= m**, the task is to maximize the dot product by inserting zeros in the second array but you cannot disturb the order of elements.

Dot product of array a and b of size n is **a[0]*b[0] + a[1]*b[1] + ... + a[n-1]*b[n-1].**

Example 1:

Input:
n = 5, a[] = {2, 3, 1, 7, 8}
m = 3, b[] = {3, 6, 7}
Output:
107
Explanation:
We get maximum dot product after inserting 0 at first and third positions in second array.
Therefore b becomes {0, 3, 0, 6, 7}.
Maximum dot product = 2*0 + 3*3 + 1*0 + 7*6 + 8*7 = 107.

Example 2:

Input:
n = 3, a[] = {1, 2, 3}
m = 1, b[] = {4}
Output:
12
Explanation:
We get maximum dot product after inserting 0 at first and second positions in second array.
Therefore b becomes {0, 0, 4}.
Maximum Dot Product = 1*0 + 2*0 + 3*4 = 12.

Your Task:
You don't need to read input or print anything. Complete the function **maxDotProduct()** which takes **n, m**, array **a** and **b** as input parameters and returns the maximum value.

Expected Time Complexity: O(n*m)
Expected Auxiliary Space: O(n*m)

Constraints:
$1 \leq m \leq n \leq 10^3$
$1 \leq a[i], b[i] \leq 10^3$

CODE

```
class Solution:

    def maxDotProduct(self, n, m, a, b):

        # dp[i][j] = max(dp[i-1][j-1] + a[i]*b[j], dp[i-1][j])

        dp = [[0]*(m+1) for _ in range(n+1)]


        for i in range(n):

            for j in range(min(m, i+1)):

                dp[i+1][j+1] = max(dp[i][j] + a[i]*b[j], dp[i][j+1])


        return dp[n][m]

        # code here
```