**Burst Balloons**

**Hard**

You are given n balloons, indexed from 0 to n - 1. Each balloon is painted with a number on it represented by an array nums. You are asked to burst all the balloons.

If you burst the ith balloon, you will get nums[i - 1] \* nums[i] \* nums[i + 1] coins. If i - 1 or i + 1 goes out of bounds of the array, then treat it as if there is a balloon with a 1 painted on it.

Return *the maximum coins you can collect by bursting the balloons wisely*.

**Example 1:**

**Input:** nums = [3,1,5,8]

**Output:** 167

**Explanation:**

nums = [3,1,5,8] --> [3,5,8] --> [3,8] --> [8] --> []

coins = 3\*1\*5 + 3\*5\*8 + 1\*3\*8 + 1\*8\*1 = 167

**Example 2:**

**Input:** nums = [1,5]

**Output:** 10

**Constraints:**

* n == nums.length
* 1 <= n <= 500
* 0 <= nums[i] <= 100

class Solution {

public:

    int maxCoins(vector<int>& a) {

        int n=a.size();

        a.insert(a.begin(), 1);

        a.push\_back(1);

        int dp[n+2][n+2];

        memset(dp, 0, sizeof(dp));

        for (int i=1; i<n+1; i++) {

            int l=1, r=i;

            while (r<=n) {

                for (int j=l; j<=r; j++) {

                    int val=a[l-1]\*a[j]\*a[r+1]+dp[l][j-1]+dp[j+1][r];

                    dp[l][r]=max(dp[l][r], val);

                }

                l++; r++;

            }

        }

        return dp[1][n];

    }

};