<https://leetcode.com/problems/burst-balloons/>

**Think it as this is the last balloon and do bottom up like MCM.**

**Reason:** to avoid dependency on left part or on right part to solve a block of (i, j) and by considering a ballon this is the last one we are dependent on only nums[i-1] and nums[j-1] means only on the boundaries.

**Approach:** f(i, j)

1. Start with the entire block/ array.
2. Try bursting all balloons one by one by considering this is the last one which will be burst at last.
3. Return the maximum point by bursting balloons in a particular fashion.

**Recursive soln:**

1. Make [1 3 6] -> [1 1 3 6 1] for the ease of calculating points.
2. Take entire block by passing i = 1, j= n
3. Try all possible balloons to burst by taking ind = i to j, cost = multiplication of that balloon and its neighbours’ values

**cost = nums[i-1]\*nums[ind]\*nums[j+1] + f(i, ind-1) + f(ind+1, j)**

1. Return the maximum point.

**Tabulation(bottom-up):** i = n to 1 and j = 1 to n

1. Initialize dp[n+2][n+2] with 0 to counter base case.
2. Use the same recurrence relation to build dp.
3. Return dp[1][n]