有 n 个气球，编号为0 到 n - 1，每个气球上都标有一个数字，这些数字存在数组 nums 中。
现在要求你戳破所有的气球。戳破第 i 个气球，你可以获得 nums[i - 1] * nums[i] * nums[i + 1] 枚硬币。 这里的 i - 1 和 i + 1 代表和 i 相邻的两个气球的序号。如果 i - 1或 i + 1 超出了数组的边界，那么就当它是一个数字为 1 的气球。
求所能获得硬币的最大数量。

回溯思路：

```
int backtrack(std::vector<int> &nums, int scores) {
    if(nums.empty()) {
        return res = std::max(res, scores);
    }

    for(int i = 0; i< nums.size();i++){
        int point = nums[i-1] * nums[i] * nums[i+1];
        int tmp = nums[i];
        // 做选择
        从nums中移除tmp
        backtrack(nums, scores+point)
        // 撤销选择
        将tmp恢复
    }
}
```

动态规划思路： 这个问题中我们每戳破一个气球nums[i]，得到的分数和该气球相邻的气球nums[i-1]和nums[i+1]是有相关性的。

1. nums[-1] = nums[n] = 1，那么我们先直接把这两个边界加进去，形成一个新的数组points;
2. 在一排气球points中，请你戳破气球0和气球n+1之间的所有气球（不包括0和n+1），使得最终只剩下气球0和气球n+1两个气球，最多能够得到多少分?
3. dp[i][j] = x表示，戳破气球i和气球j之间（开区间，不包括i和j）的所有气球，可以获得的最高分数为x。
4. 那么根据这个定义，题目要求的结果就是dp[0][n+1]的值，而 base case 就是dp[i][j] = 0，其中0 <= i <= n+1, j <= i+1，因为这种情况下，开区间(i, j)中间根本没有气球可以戳。
5. 根据刚才对dp数组的定义，如果最后一个戳破气球k，dp[i][j]的值应该为：`dp[i][j] = dp[i][k] + dp[k][j] + points[i]*points[k]*points[j]`

```
class Solution {
public:
  int maxCoins(std::vector<int>& nums) {
    int n = nums.size();
    if (n == 0) {
      return 0;
    }

    // 添加两端构造新数组
```

```
    std::vector<int> points = std::vector<int>(n + 2, 1);
    for (int i = 1; i < n + 1; i++) {
      points[i] = nums[i - 1];
    }

    // 构造dp
    std::vector<std::vector<int>> dp =
        std::vector<std::vector<int>>(n + 2, std::vector<int>(n + 2, 0));

    // 开始状态转移，i 从上到下
    for (int i = n; i >= 0; i++) {
      // j从右向左
      for (int j = i + 1; j < n + 2; j++) {
        // 最后戳破的气球
        for (int k = i + 1; k < j; k++) {
          // 择优选择
          dp[i][j] =
              std::max(dp[i][j],
                       dp[i][k] + dp[k][j] + points[i] * points[j] *
points[k]);
        }
      }
    }

    return dp[0][n + 1];
  }
};
```