<https://leetcode.com/problems/partition-array-for-maximum-sum/>

Front dp pattern - tryout all partitions from the start.

f(i, arr) - max possible sum till i

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

1. Start from i=0
2. Try partitions ind from i to min(n, i+k), keep a maxSum variable to store max sum for every partition for ind. Keep some variable mx to store max element and len to calc length.

**int temp = mx\*len + f(ind+1)**

Sum = max(sum, temp)

1. Return max sum.

**Tabulation(bottom-up):** i = n-1 to 0 and j = i to n

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