<https://leetcode.com/problems/palindrome-partitioning-ii/>

Front dp pattern - tryout all partitions from the start.

Similar to palindrome partitioning where you count how many palindrome partitioning can be done, here we will be finding which way has min number of partitions.

f(i, s) - min partitions till i

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

1. Start from i=0
2. Try partitions from i to j WHERE j = i to n. If any i to j substring is palindrome find for the remaining string and add 1 to it.

**partition = 1+ minPartition(j+1, s)**

1. Return min partitions.

**Tabulation(bottom-up):** i = n-2 to 0 and j = i to n

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