1. You are given the price of a stock on each of the last \( n \) days \((p_1, p_2, \ldots, p_n)\). If you had bought the stock on day \( b \) and sold it on day \( s \) (with \( 1 \leq b < s \leq n \)), you would have generated a profit of \( p_s - p_b \). Design an \( \mathcal{O}(n) \) dynamic programming algorithm that determines which days \( b \) and \( s \) you should have bought and then sold the stock for maximum profit.

2. You are given an \( n \times n \) matrix of integers, representing the values of every location in a grid. You must find a path from the top-left square to the bottom-right square which only moves right or down at each step. Design an \( \mathcal{O}(n^2) \) dynamic programming algorithm to find the path with the maximum sum value.

3. Design an \( \mathcal{O}(n^2) \) dynamic programming algorithm to find the longest palindromic contiguous substring of an input string.