C++ Auto

```cpp
class Solution {
public:
    int n;
    // Memoization
    int dp[2501][2501]; // Depends on Constraints
    int solve(vector<int>& nums, int index, int prev_index){
        // Base Case
        if(index >= n){
            return 0;
        }

        if(prev_index != -1 && dp[index][prev_index] != -1){
            return dp[index][prev_index];
        }

        // Condition to check whether we can take sequence or not
        int take = 0;
        if(prev_index == -1 || nums[index] > nums[prev_index]){
            take = 1 + solve(nums, index + 1, index);
        }
        // Condition to check whether we skip sequence or not
        int skip = solve(nums,index + 1,prev_index);
        if(prev_index != -1){
            dp[index][prev_index] = max(take,skip);
        }
        return max(skip,take);
    }
    int lengthOfLIS(vector<int>& nums) {
        n = nums.size();
        // Initializing with -1
        memset(dp,-1,sizeof(dp));
        return solve(nums, 0, -1);
    }
};
```


Daily Coding Challenge Completed!
Completion Streak: 109 Days
Consistency is key, see you tomorrow!