Pick-Not Pick (Include-Exclude) Problems Sheet

S.N.	Problem Name	Detailed Problem Sheet
1	Maximum Sum of Non-Adjacent Elements	**Description:** Given an array, find the maximum sum of elements such that no two elements are adjacent. **Input:** n (size of array), array of integers. **Output:** Maximum sum **Constraints:** $1 \leq n \leq 10^5$, $-10^4 \leq arr[i] \leq 10^4$ **Example:** Input: [2, 1, 4, 9] ⟶ Output: 11 **Explanation:** Pick 2 and 9 ⟶ Sum = 11 **Approach:** Pick current + dp[i-2] or skip current (dp[i-1])
2	Subset Sum Problem	**Description:** Check if there exists a subset with sum equal to the target. **Input:** n, array, target sum **Output:** true/false **Constraints:** $1 \leq n \leq 10^3$, $0 \leq target \leq 10^4$ **Example:** Input: [3, 34, 4, 12, 5, 2], target = 9 ⟶ Output: true **Explanation:** Subset {4, 5} sums to 9. **Approach:** Pick current (sum decreases) or skip current
3	0/1 Knapsack Problem	**Description:** Given weights, values, and a capacity W, maximize value by picking/not picking items. **Input:** n, arrays of weights and values, capacity W **Output:** Maximum achievable value **Constraints:** $1 \leq n \leq 10^3$, $1 \leq W \leq 10^4$ **Example:** Input: weights = [1,3,4,5], values = [1,4,5,7], W=7 ⟶ Output: 9 **Explanation:** Pick items with weight 3 and 4 ⟶ Total value = 9 **Approach:** Pick current (if weight allows) or skip current
4	Partition Equal Subset Sum	**Description:** Check if the array can be partitioned into two subsets with equal sum. **Input:** n, array **Output:** true/false **Constraints:** $1 \leq n \leq 200$, $1 \leq arr[i] \leq 100$ **Example:** Input: [1, 5, 11, 5] ⟶ Output: true **Explanation:** {1,5,5} and {11} both sum to 11. **Approach:** Reduce to subset sum with target = total_sum/2
5	House Robber Problem	**Description:** Same as problem 1, but framed in a house-robbing scenario. **Input:** n, array of money in houses **Output:** Maximum amount robbed **Constraints:** $1 \leq n \leq 10^5$, $0 \leq money \leq 10^4$ **Example:** Input: [2, 7, 9, 3, 1] ⟶ Output: 12 **Explanation:** Pick 2, 9, 1 ⟶ Sum = 12 **Approach:** Standard pick-not pick DP
6	Target Sum Problem	**Description:** Assign '+' or '−' to each element to reach target sum. **Input:** n, array, target sum **Output:** Number of ways to assign signs to reach target **Constraints:** $1 \leq n \leq 20$, $-1000 \leq arr[i] \leq 1000$

S.N.	Problem Name	Detailed Problem Sheet
		Example: Input: nums = [1,1,1,1,1], target = 3 ⟶ Output: 5 **Explanation:** 5 ways to assign signs to get 3. **Approach:** At each step, add or subtract the current element
7	Maximum Subset with No Consecutive Elements	**Description:** Find maximum sum of non-consecutive elements. **Input:** n, array **Output:** Maximum sum **Constraints:** 1 ≤ n ≤ 10^5, -10^4 ≤ arr[i] ≤ 10^4 **Example:** Input: [3, 2, 5, 10, 7] ⟶ Output: 15 **Explanation:** Pick 3, 10, 2 ⟶ Sum = 15 **Approach:** Same as house robber
8	Coin Change (Minimum Coins)	**Description:** Find minimum number of coins to make a target amount. **Input:** n, array of coins, target amount **Output:** Minimum coins needed **Constraints:** 1 ≤ n ≤ 12, 1 ≤ amount ≤ 10^4 **Example:** Input: coins = [1, 2, 5], amount = 11 ⟶ Output: 3 **Explanation:** 5+5+1 ⟶ 3 coins **Approach:** Pick current coin (stay at index) or skip to next coin
9	Longest Increasing Subsequence (LIS)	**Description:** Find the length of the longest increasing subsequence. **Input:** n, array **Output:** Length of LIS **Constraints:** 1 ≤ n ≤ 2500, -10^4 ≤ arr[i] ≤ 10^4 **Example:** Input: [10, 9, 2, 5, 3, 7, 101, 18] ⟶ Output: 4 **Explanation:** LIS is [2, 3, 7, 101] **Approach:** Pick current if it's increasing, else skip
10	Painting Fence Problem	**Description:** Given n fences and k colors, find the number of ways to paint such that no more than two adjacent fences have the same color. **Input:** n, k **Output:** Number of ways to paint **Constraints:** 1 ≤ n ≤ 10^4, 1 ≤ k ≤ 100 **Example:** Input: n = 3, k = 2 ⟶ Output: 6 **Explanation:** Various combinations avoiding 3 adjacent same colors. **Approach:** Pick same color (limited) or pick different color

Common DP Approaches Across All:

- Recursion
- Memoization (Top-Down)
- Tabulation (Bottom-Up)
- Space Optimization (for linear index problems like House Robber)

Key Characteristics:

- **Decision Tree:** Pick or not pick at each step
- **Intrinsic Value:** When you pick the current item
- **Extrinsic Value:** When you skip the current item
- **State Parameters:** Usually involve index and sometimes additional parameters like remaining weight, sum, or target.