Course Name: Algorithm Design and Analysis
Course ID: 180086081200P1002H

Chapter Name: Dynamic Programming

For problems 1-5, you should do at least the following things:

1. Describe your algorithm in natural language AND pseudo-code;
2. Describe the optimal substructure and DP equation;
3. Prove the correctness of your algorithm;
4. Analyse the complexity of your algorithm.

---

1. Money Robbing

   A robber is planning to rob houses along a street. Each house has a certain amount of money stashed, the only constraint stopping you from robbing each of them is that adjacent houses have security system connected and it will automatically contact the police if two adjacent houses were broken into on the same night.

   (a) Given a list of non-negative integers representing the amount of money of each house, determine the maximum amount of money you can rob tonight without alerting the police.

   (b) What if all houses are arranged in a circle?

2. Ugly Number

   An ugly number is a positive integer whose prime factors are limited to 2, 3, and 5. Given an integer $\boldsymbol{n}$, return the $\boldsymbol{n^{th}}$ ugly number.

   (a) Using a brute-force algorithm to solve this problem, analyze the time complexity of your implemented brute-force algorithm and explain why the algorithm's time complexity is O(n^2), where n is the number of points.

   (b) Propose an improved algorithm to solve this problem with a time complexity better than the brute-force algorithm. Describe the algorithm's idea and analyze its time complexity.

3. Unique Binary Search Trees

   Given n, how many structurally unique BST's (binary search trees) that store values 1...n?

   Note: Given $n = 3$, there are a total of 5 unique BST's:

4. Largest Divisible Subset

   Given a set of distinct positive integers, find the largest subset such that every pair (S_i, S_j) of elements in this subset satisfies: $S_i\%S_j = 0$ or $S_j\%S_i = 0$. Please return the largest size of the subset.

   Note: $S_i\%S_j = 0$ means that S_i is divisible by S_j.

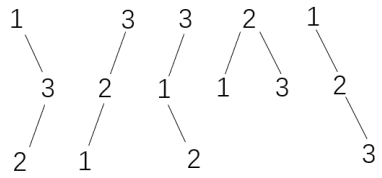

Figure 1: BST

5. Target Sum

   You are given an integer array nums and an integer target.

   You want to build an expression out of nums by adding one of the symbols '+' and '-' before each integer in nums and then concatenate all the integers.

   For example, if nums = [2, 1], you can add a '+' before 2 and a '-' before 1 and concatenate them to build the expression "+2-1".

   Return the number of different expressions that you can build, which evaluates to target.

   Example:

   Input: nums = [1,1,1,1,1], target = 3

   Output: 5

   Explanation: There are 5 ways to assign symbols to make the sum of nums be target 3.

   -1 + 1 + 1 + 1 + 1 = 3

   +1 - 1 + 1 + 1 + 1 = 3

   +1 + 1 - 1 + 1 + 1 = 3

   +1 + 1 + 1 - 1 + 1 = 3

   +1 + 1 + 1 + 1 - 1 = 3