**Problem 1: Dragon Tower — The Spiral Climb**

In the ancient realm of **Zerthion**, a cursed **Dragon Tower** stands tall. You, **Sir** Aryon, the chosen knight, have been tasked with reaching the top of the tower to retrieve the Flame Crystal that can save your kingdom from eternal winter.

The tower has N magical floors, each designed by the dragon to slow intruders. You can either:

* Climb 1 floor at a time through narrow stairs
* Or make a 2-floor leap using enchanted boots (which tires you more)

You must reach exactly the N-th floor to claim the Flame Crystal — overshooting means falling into a pit of lava.

Your mission: determine in how many unique ways you can reach the top of the tower.

**Input:**

* N – The total number of floors in the Dragon Tower (1 ≤ N ≤ 45)

**Constraints:**

* 1 ≤ N ≤ 45

**Input:**

N = 4

**Output:**

5

**SOLUTION :-**

**import java.util.Scanner;**

**public class DragonTower {**

**public static void main(String[] args) {**

**Scanner scanner = new Scanner(System.in);**

**int N = scanner.nextInt();**

**System.out.println(countWays(N));**

**}**

**public static int countWays(int N) {**

**if (N == 0 || N == 1) {**

**return 1;**

**}**

**int prevPrev = 1; // Represents f(0)**

**int prev = 1; // Represents f(1)**

**int current = 0;**

**for (int i = 2; i <= N; i++) {**

**current = prev + prevPrev;**

**prevPrev = prev;**

**prev = current;**

**}**

**return prev;**

**}**

**}**

**Problem 2: Magical Dice — Reach the Treasure**

**You are Aelar, a young wizard navigating the Corridor of Curses in the depths of Ebonvault Dungeon. The corridor is N tiles long, and the only way to move forward is by rolling your enchanted 6-sided die.**

* **Each roll teleports you forward by 1 to 6 tiles, depending on the outcome.**
* **You must find all possible sequences of rolls that land you exactly on tile N — going beyond it triggers a deadly trap.**

**The treasure at the end is the Orb of Infinite Knowledge, but it's protected by logic itself. You must figure out in how many ways you can reach it.**

**Input:**

**An integer N — number of tiles to reach the treasure**

**N = 3**

**Constraints:**

**1 ≤ N ≤ 25**

**Output:**

**4**

**Explanation:**

**(1,1,1), (1,2), (2,1), (3)**

**SOLUTION :-**

**import java.util.Scanner;**

**public class MagicalDice {**

**public static void main(String[] args) {**

**Scanner scanner = new Scanner(System.in);**

**int N = scanner.nextInt();**

**System.out.println(countWays(N));**

**}**

**public static int countWays(int N) {**

**if (N == 0) return 1;**

**int[] dp = new int[N + 1];**

**dp[0] = 1; // Base case**

**for (int i = 1; i <= N; i++) {**

**for (int j = 1; j <= 6; j++) {**

**if (i - j >= 0) {**

**dp[i] += dp[i - j];**

**}**

**}**

**}**

**return dp[N];**

**}**

**}**