**Problem: Stairs of Heaven**

Arjun is ready to climb the stairs of heaven. There are N steps to reach heaven. He can either take one step or two steps each time. Find out in how many distinct ways he can climb to heaven.

**Input Format**

* The first line takes an integer N (the number of steps in the stairs) as input.

**Constraints**

* 1 <= N <= 45

**Output Format**

* Print the number of distinct ways he can climb to heaven.

**Sample Input**

3

**Sample Output**

3

**Explanation**

There are three ways to climb to the top:

1. 1 step + 1 step + 1 step
2. 1 step + 2 steps
3. 2 steps + 1 step

**Extra Test Cases:**

**Test Case 1:**

INPUT:

1

OUTPUT:

1

**Test Case 2:**

INPUT:

2

OUTPUT:

2

**Test Case 3:**

INPUT:

10

OUTPUT:

89

**Test Case 4:**

arduino

Copy code

INPUT:

28

OUTPUT:

514229

**Test Case 5:**

arduino

Copy code

INPUT:

33

OUTPUT:

5702887

**Test Case 6:**

arduino

Copy code

INPUT:

45

OUTPUT:

1836311903

**Solution**

To solve this problem, we can use dynamic programming. The number of ways to reach the iii-th step is the sum of the ways to reach the (i−1)(i-1)(i−1)-th step and the (i−2)(i-2)(i−2)-th step. This is because Arjun can reach the iii-th step either from the (i−1)(i-1)(i−1)-th step or the (i−2)(i-2)(i−2)-th step.

Thus, we have:

* ways[i]=ways[i−1]+ways[i−2]ways[i] = ways[i-1] + ways[i-2]ways[i]=ways[i−1]+ways[i−2]

Where:

* ways[0]=1ways[0] = 1ways[0]=1 (no steps to climb, one way to stay where you are)
* ways[1]=1ways[1] = 1ways[1]=1 (one step, one way to climb)

Let's implement this in Python:

python

Copy code

def stairs\_of\_heaven(n):

if n == 0:

return 1

if n == 1:

return 1

ways = [0] \* (n + 1)

ways[0] = 1

ways[1] = 1

for i in range(2, n + 1):

ways[i] = ways[i - 1] + ways[i - 2]

return ways[n]

# Input

n = int(input().strip())

# Output

print(stairs\_of\_heaven(n))

This code calculates the number of distinct ways Arjun can climb to heaven given NNN steps using a dynamic programming approach.