https://course.acciojob.com/idle?question=d6662886-106d-48f6-be6a-ea491c20827a

EASY

Max Score: 30 Points

Number of paths to the end

You are given an array `A` consisting of `n` integers. You are initially positioned at the array's first index, and each element in the array represents your maximum jump length at that position.

You are required to print the number of different paths via which you can reach the end of the array.

Note

You can jump fewer steps than the maximum possible in a move.

Input Format

Input has been managed for you.

In example input, the first line contains an integer `n` denoting the length of the array.

The next line contains `n` space-separated integers denoting elements of the array.

Output Format

For each test case, return an integer denoting the number of all the paths via which you can reach the end of the array.

Example 1

Input

`5`

`2 3 1 1 4`

Output

`4`

Explanation

`There are four ways to reach from the starting position to the end.`

Example 2

Input

`6`

`2 3 0 2 2 3`

Output

`5`

Explanation

`There are five ways to reach from the starting position to the end.`

Constraints

1 <= `n` <= 20

0 <= `a[i]` <= 20

Topic Tags

- **DP**
- **Arrays**

My code

```
// in java
import java.io.*;
import java.util.*;

public class Main {

   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      int n = sc.nextInt();
      int[] arr = new int[n];
      for (int i = 0; i < arr.length; i++) {
         arr[i] = sc.nextInt();
      }

      Solution ob =new Solution();

      System.out.println(ob.alljump(arr,n));
```

```
    }

}

class Solution{

        public int solve(int index,int dp[],int a[],int n)
{
        if(index>n)
        {
                return 0;
        }
    if(index==n)
        return 1;

    if(dp[index]!=-1)
        return dp[index];

    int ans=0;
    int jumps=a[index];
    for(int i=1; i<=jumps; i++)
    {
        ans+=solve(index+i,dp,a,n);
    }
    return dp[index]=ans;


}

    public int alljump(int arr[],int n){
      //Write code here
                int dp[]=new int[n+1];
                for(int i=0; i<=n; i++)
                            dp[i]=-1;
                return solve(0,dp,arr,n);
    }
}
```