/\*

You are climbing a staircase. It takes n steps to reach the top.

Each time you can either climb 1 or 2 steps. In how many distinct ways can you climb to the top?

Example 1:

Input: n = 2

Output: 2

Explanation: There are two ways to climb to the top.

1. 1 step + 1 step

2. 2 steps

Example 2:

Input: n = 3

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

Constraints:

1 <= n <= 45

\*/

/\*\*

\* @param {number} n

\* @return {number}

\*/

// using map/hashtable for memoization

var climbStairs = function(n) {

let map = new Map();

return findSteps(0,n,map)

};

function findSteps(i,n,map){

if(i>n){

return 0;

}else if(i==n){

return 1;

}else{

if(map.has(i)){

return map.get(i);

}else {

let val = findSteps(i+1,n,map)+findSteps(i+2,n,map);

map.set(i,val);

return val;

}

}

}

// var climbStairs = function(n) {

// const memo = Array(n + 1).fill(0);

// memo[1] = 1;

// memo[2] = 2;

// for (let i = 3; i <= n; ++i) {

// memo[i] = memo[i - 1] + memo[i - 2];

// }

// return memo[n];

// };

// var climbStairs = function(n) {

// let one = 1;

// let two = 1;

// for(let i=0;i<n-1;i++){

// let temp = one;

// one = one + two;

// two = temp;

// }

// return one

// };