Challenge: The Staircase Problem

In this lesson, you will work on an interesting problem that can be solved using dynamic programming.

We'll cover the following

Problem statement

Nick is standing next to a staircase that leads to his apartment. The staircase has `n` total steps; Nick knows he can climb anywhere between `1` and `m` steps in one jump. He thinks about how many ways there are to climb this staircase. He realizes it is a big number since there are a lot of possible combinations. So, he has asked you to write an algorithm for him that tells him the number of possible ways to climb a staircase given `n` (number of steps) and, `m` (number of steps covered in biggest jump).

Input

Your algorithm will take as input `n`, the number of steps in the staircase and, `m`, the number of steps covered in the biggest leap. Nick can jump any number of stairs between `1` and `m`.

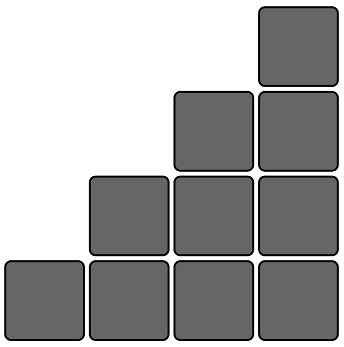
```
n = 4
m = 2
```

Output

Your algorithm will return the number of possible ways to climb the staircase.

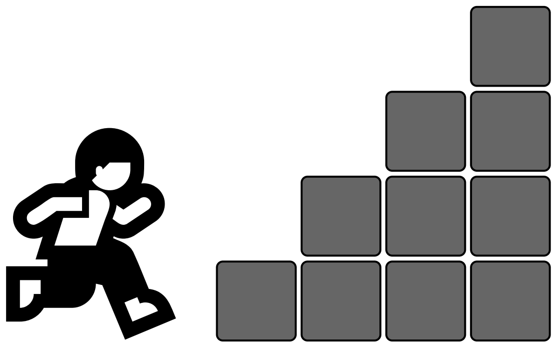
```
staircase(n, m) = 5
```

Look at the following visualization as a hint for the challenge.



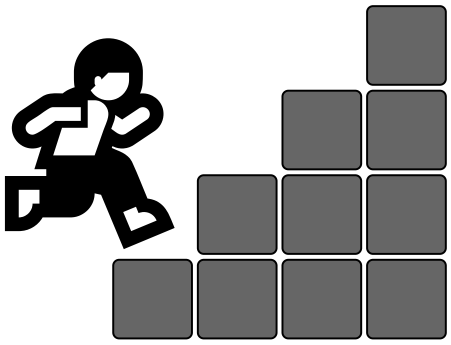
Climb staircase with n=4, and m=2

1 of 7

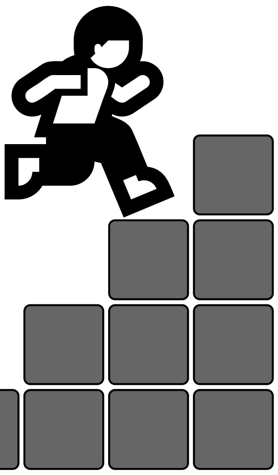


Nick can go up either one or two steps

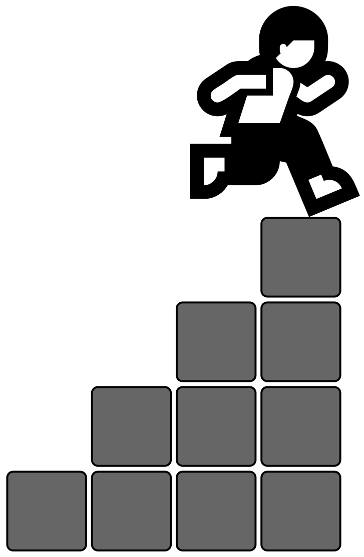
2 of 7



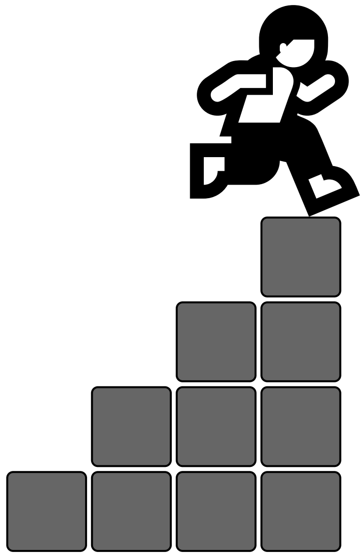
Let's say if he takes 1 step, now again he can take 1 step or 2



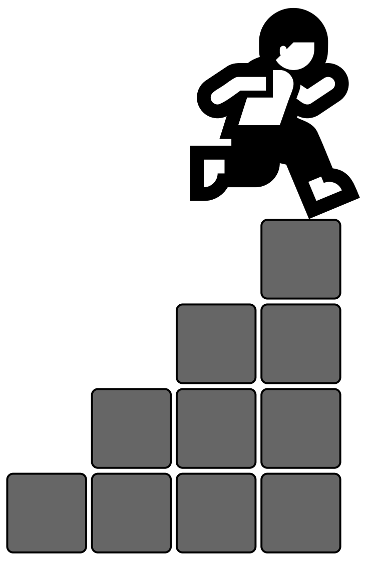
Let's say if he takes 2 step, now he can only take 1 step



This was only one way to reach the top, there could have been more ways too



This could be one way



This could be another

7 of 7

Coding challenge

The above figure shows a few possible ways to reach the top. You are required to find all possible ways to reach the top. Remember that this problem's naive solution will have a very high runtime complexity. Therefore, you must use memoization techniques to reduce the time complexity. First, write the simple naive solution and then try to figure out how you can employ memoization. Since we have enabled stress testing with larger inputs in the code, your solution may timeout if you have not memoized correctly. Set `stressTesting` to `False` to check for correctness of the solution before moving on to memoization.

Best of luck!

```
def staircase(n, m):
  # Your code goes here

  return 0

stressTesting = True
```

Hint 1 of 2

You might want to explore all the options you have at each step.

In the next lesson, we will see the solution to this problem.