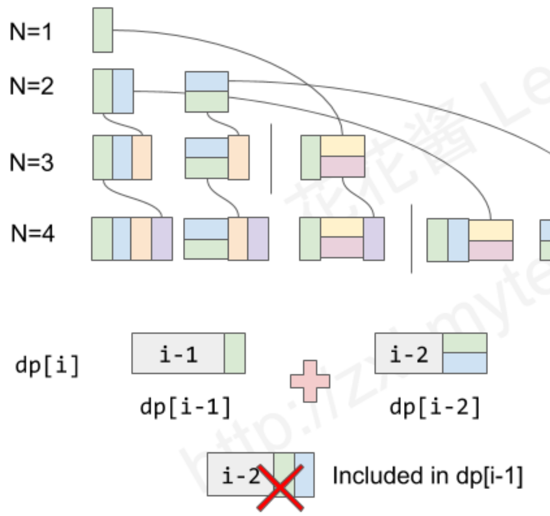
A simpler problem:
How many ways that we can cover a N*2 board using one type of domino (1x2).



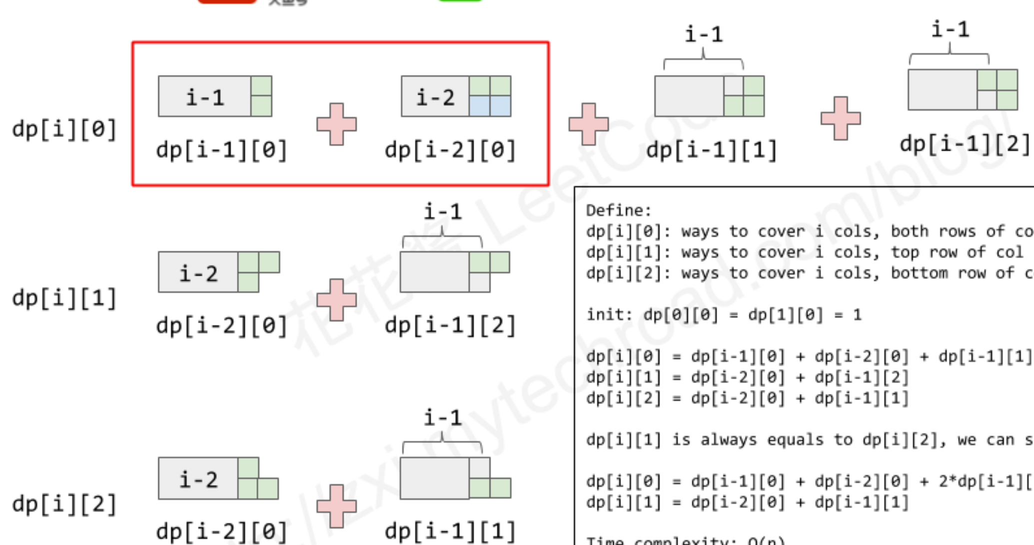

```
Define: dp[i]: ways to cover i cols
init: dp[0] = dp[1] = 1

dp[i] = dp[i-1] + dp[i-2] Fibonacci sequence!

ans: dp[n]

Time complexity: O(n)
Space complexity: O(n) -> O(1)
```


LeetCode 花花酱 huahualeetcode 790. Domino and Tromino Tiling




```
Define:
dp[i][0]: ways to cover i cols, both rows of col i are covered
dp[i][1]: ways to cover i cols, top row of col i is covered
dp[i][2]: ways to cover i cols, bottom row of col i is covered

init: dp[0][0] = dp[1][0] = 1

dp[i][0] = dp[i-1][0] + dp[i-2][0] + dp[i-1][1] + dp[i-1][2]
dp[i][1] = dp[i-2][0] + dp[i-1][2]
dp[i][2] = dp[i-2][0] + dp[i-1][1]

dp[i][1] is always equals to dp[i][2], we can simpliy

dp[i][0] = dp[i-1][0] + dp[i-2][0] + 2*dp[i-1][1]
dp[i][1] = dp[i-2][0] + dp[i-1][1]

Time complexity: O(n)
Space complexity: O(n) -> O(1)
```

http://zxi.mytechroad.com/blog/