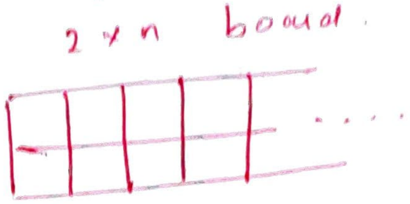
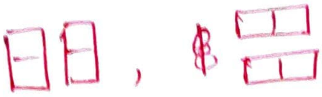
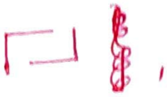
Domino Tromino Tiling Problem (Leetcode 790)

First, let us make some initial possibilities for n = 1, 2, 3, 4. . consider answer as 1 for n = 0.

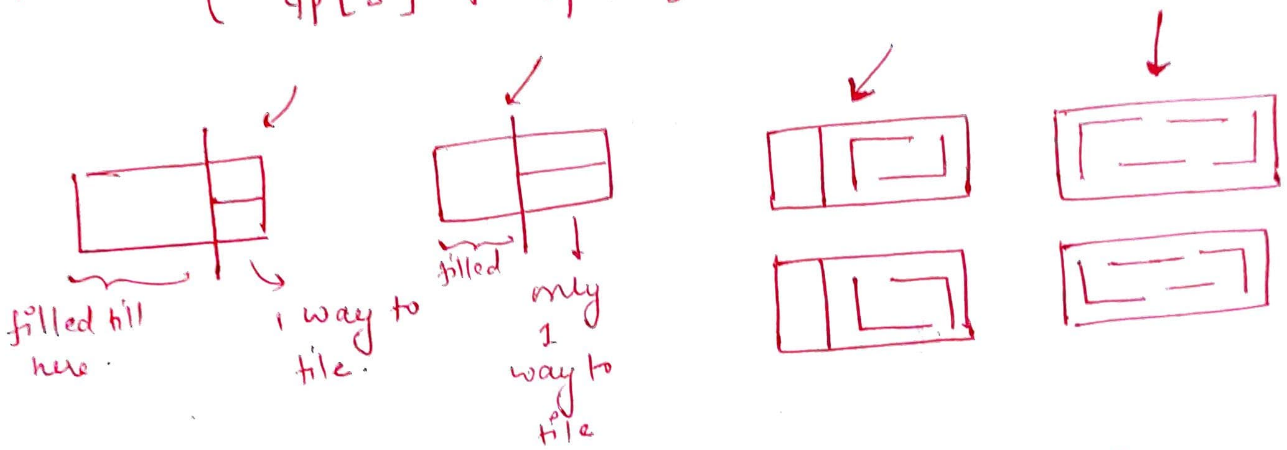


n = 1 → { } . → 1

n = 2 → {  } → { =, || } → 2

n = 3 → { |||, |=, =|,  }.
↳ 5

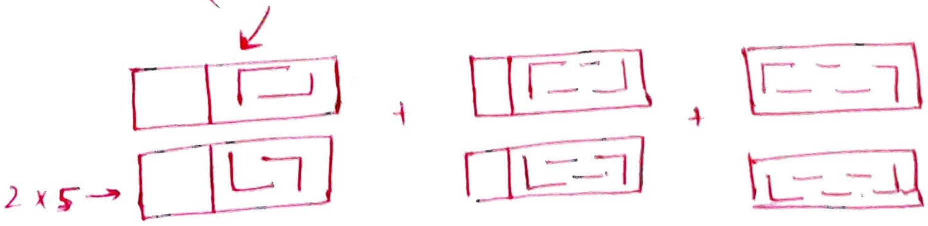
n = 4 → { dp[3] + dp[2] + 2 × dp[1] + 2 × dp[0] }.



The pattern which start and end with a tromino need to be considered for every combination below (n - 3).

For example, in case of n = 5, we can have

dp[4] + dp[3] + 2 (dp[2] + dp[1] + dp[0])



Thus,

$$dp[n] = dp[n-1] + dp[n-2] + 2\,(dp[n-3] \dots dp[0])$$

$$dp[n-1] = dp[n-2] + dp[n-3] + 2\,(dp[n-4] \dots dp[0])$$

subtracting, we get

$$dp[n] - dp[n-1] = dp[n-1] + dp[n-3]$$

$$\Rightarrow dp[n] = 2\,dp[n-1] + dp[n-3] \qquad \{ n \geq 1 \}$$

(given)

Code

```
int   numTilings (int n) {
          vector <int> dp (n+1, 0);
            if (n == 1) return 1;
            if (n == 2) return 2;

            dp[0] = 1;
            dp[1] = 1;
            dp[2] = 2;

           for (int i = 3 ; i < n+1; i++) {

                  dp[i] = 2 x dp[i-1] + dp[i-3];

           }
           return dp[n];

     }
```

Time complexity → O(N)

spece complexity → O(N)

space complexity can be reduced to O(1) by using 3 variables and updating them with each loop.

Code (n ≥ 1)

```
int numTilings (int n) {
    if (n==1) return 1;
    if (n==2) return 2;

    long long thirdlast = 1;
    long long secondlast = 1;
    long long last = 2;

    long long num = 1000000007;
    for (int i = 3; i < n+1; i++) {
        long long temp = last;
        last = (last % num) + (last % num)
                   + (thirdlast % num);

        thirdlast = secondlast;

        secondlast = temp;

    }

    return last % num
}.
```

Time complexity → O(n)

space complexity → O(1).

Also, leetcode wants answer modulo 10^9+7. so, use long long and use modulo at each step to avoid overflow.

Last two solutions were bottom up, Now, we will try Topdown approach.

Formula remains same.

$$T(n) = 2\,T(n-1) + T(n-3);$$

$(n \geq 1,\ n \leq 1001)$

Code

```
vector <int> dp (1001, 0) ;

int numTilings (int n) {

      if (n == 0) return 1;
      else if (n == 1) return 1;
      else if (n == 2) return 2;
      else if ( dp[n] != 0 ) return dp[n];

      else {
            dp [n] = 2 x numTilings (n - 1) + numTilings
                                                    (n - 3);

            return dp[n];
      }
```

Time Complexity → O(n)
space complexity → O(n)

space taken is more because recursion call·stack is also maintained.