The **Fibonacci numbers**, commonly denoted `F(n)` form a sequence, called the **Fibonacci sequence**, such that each number is the sum of the two preceding ones, starting from `0` and `1`. That is,

```
F(0) = 0, F(1) = 1
F(n) = F(n - 1) + F(n - 2), for n > 1.
```

Given `n`, calculate `F(n)`.

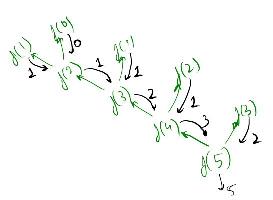
0	1	1	2	3	5	8	13	....
0	1	2	3	4	5			

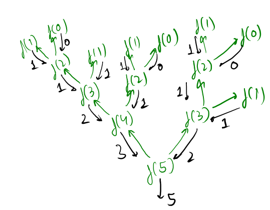
$f(n) = f(n-1) + f(n-2)$

n=5

dp

0	1	2	3	4	5
0	0	1	2	3	5





```
int recur(n) {
  if(n==0) return 0
  if(n==1) return 1

  int fnm1 = recur(n-1)
  int fnm2 = recur(n-2)

  return fnm1 + fnm2;
}
```

Tabulation

dp =

0	1	2	3	4	5
0	1	1	2	3	5

```
for(int i = 2; i<=n; i++){

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

}
```

Optimization

n = 5

3 → fnm2 = 2

4 → fnm1 = 3

5 → curr =

Good boy/Girl DP Strategy

⓪ Memo / Recur

① Start loops from base case

② Convert return → continue
  ↳ if return is having a value, save that to DP array and then continue.

③ At the place of recursion calls access dp array