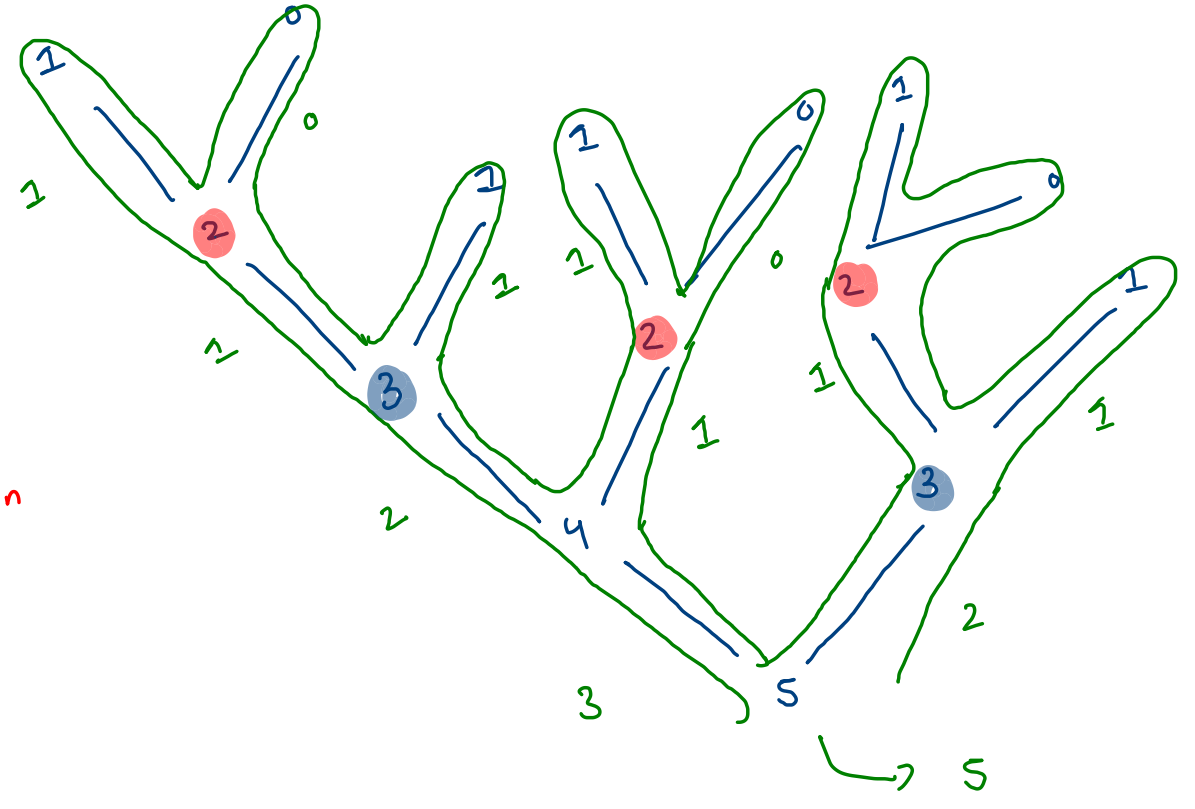
```java
public static int fib(int n) {
    if(n == 0 || n == 1) {
        return n;
    }

    int fibn = fib(n-1) + fib(n-2);
    return fibn;
}
```

0 1 1 2 3 5

left -> n-1

right -> n-2

(i) memoisation

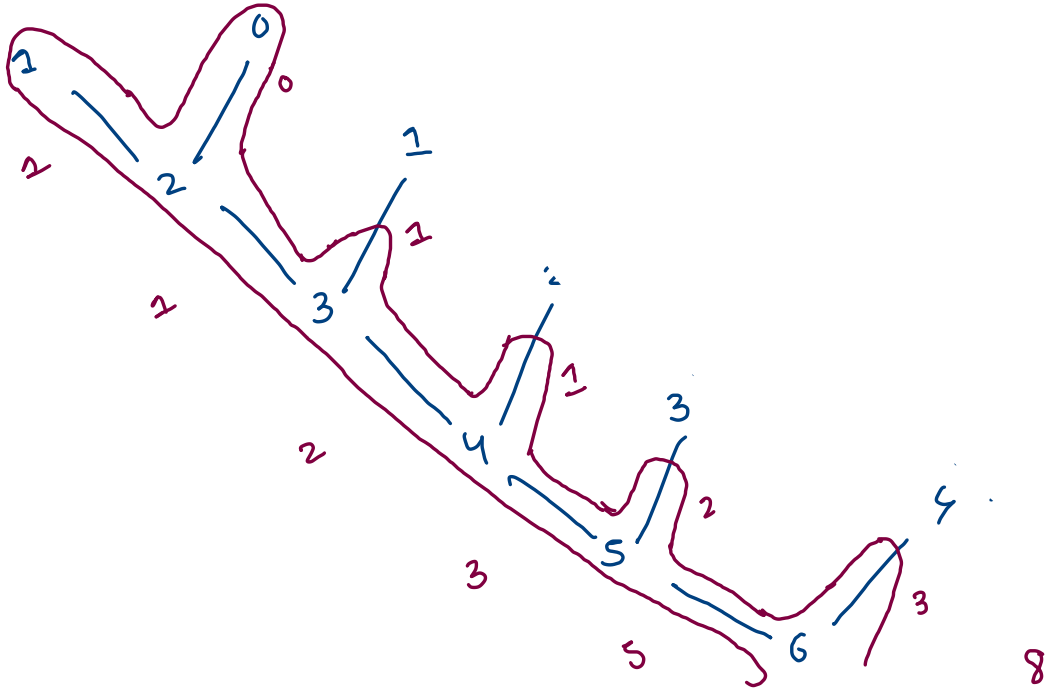


question - bank

n = 6

0	1	2	2	3	5	8
0	1	2	3	4	5	6

qb[i] -> jib(i)



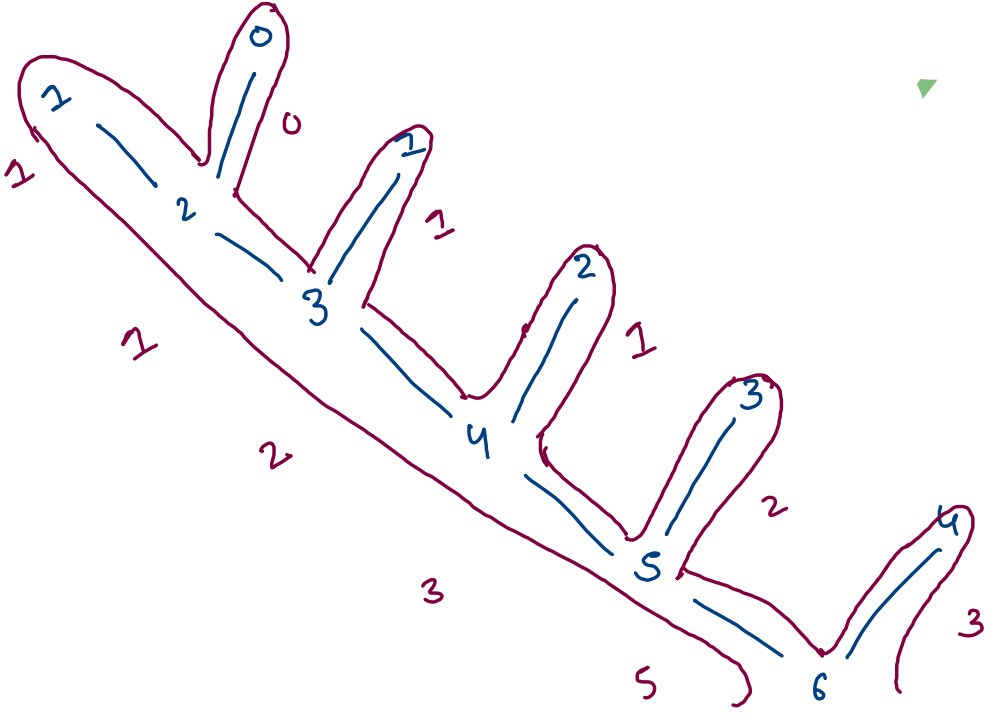
```java
public static int fib(int n,int[]qb) {
    if(n == 0 || n == 1) {
        return n;
    }

    if(qb[n] != -1) {
        return qb[n];
    }

    int fibn = fib(n-1,qb) + fib(n-2,qb);
    qb[n] = fibn;

    return fibn;
}
```

n = 6

qb

0	1	2	3	4	5	6
-1	-1	~~-1~~ 1	~~-1~~ 2	~~-1~~ 3	~~-1~~ 5	~~-1~~ 8

memoisation



8

$n = 6$

II. tabulation

(i) create storage

(ii) assign meaning

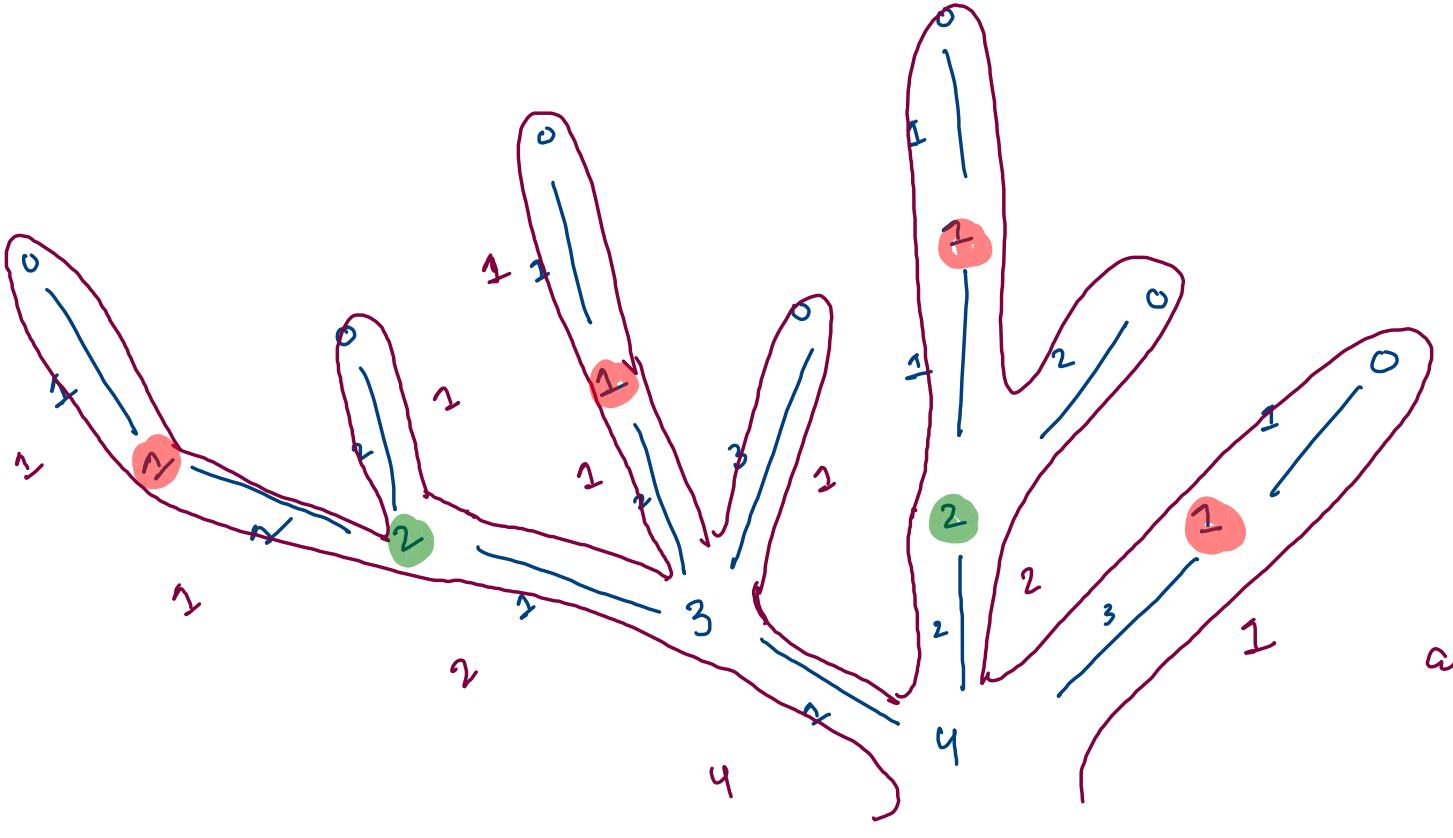
(iii) travel and solve from smallest problem to largest

0	1	1	2	3	5	8
0	1	2	3	4	5	6

dp

dp[i] -> fib(i)

$$\boxed{dp[i] = dp[i-1] + dp[i-2]}$$

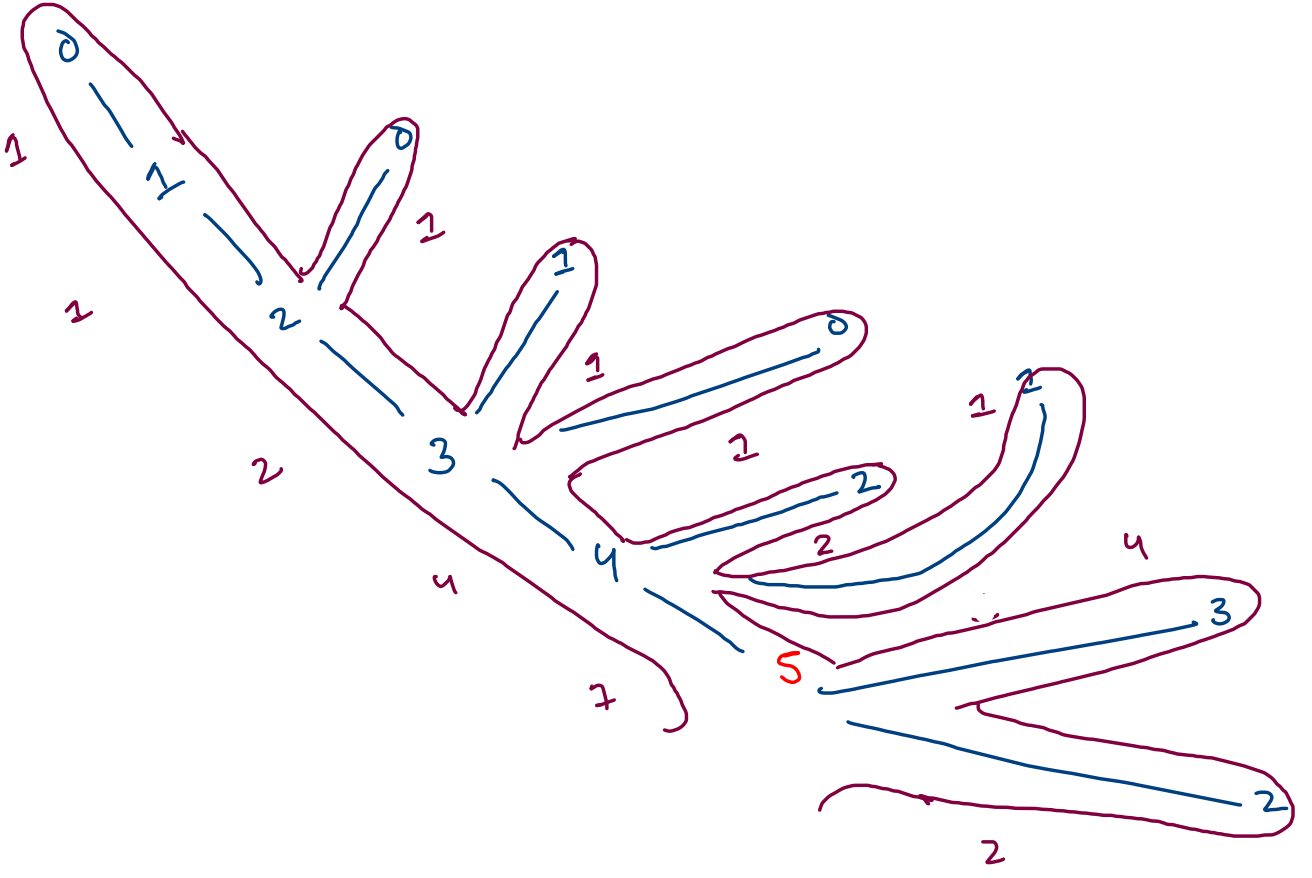


ans = 7

```java
public static int climbStairs_mem(int n,int[]qb) {
    if(n == 0) {
        return 1;
    }

    if(qb[n] != -1) {
        return qb[n];
    }

    int ways = 0;

    if(n-1 >= 0) {
        ways += climbStairs_mem(n-1,qb);
    }

    if(n-2 >= 0) {
        ways += climbStairs_mem(n-2,qb);
    }

    if(n-3 >= 0) {
        ways += climbStairs_mem(n-3,qb);
    }

    qb[n] = ways;
    return ways;
}
```

n = 5

	1	2	4	7	13
0	1	2	3	4	5



$n = 5$

tabulation

(i) create storage

(ii) assign meaning to storage

(iii) travel and solve

1	1	2	4	7	13
0	1	2	3	4	5

$dp[i] \rightarrow$ i to 0 ways

$$dp[i] = dp[i-1] + dp[i-2] + dp[i-3].$$