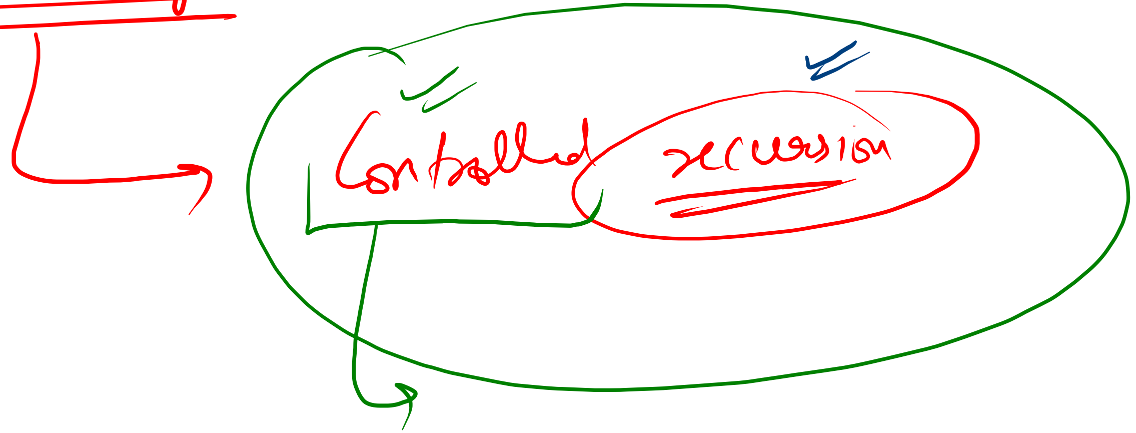
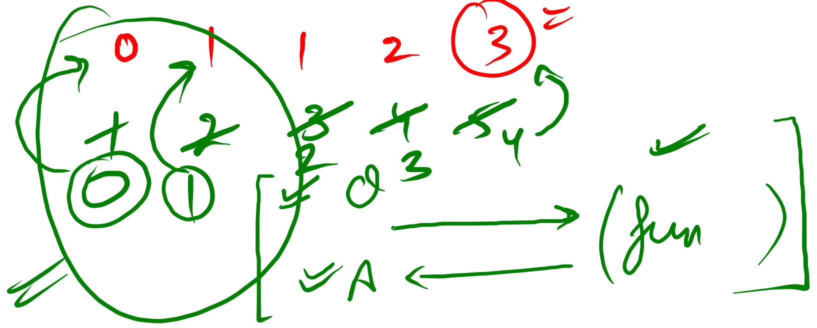
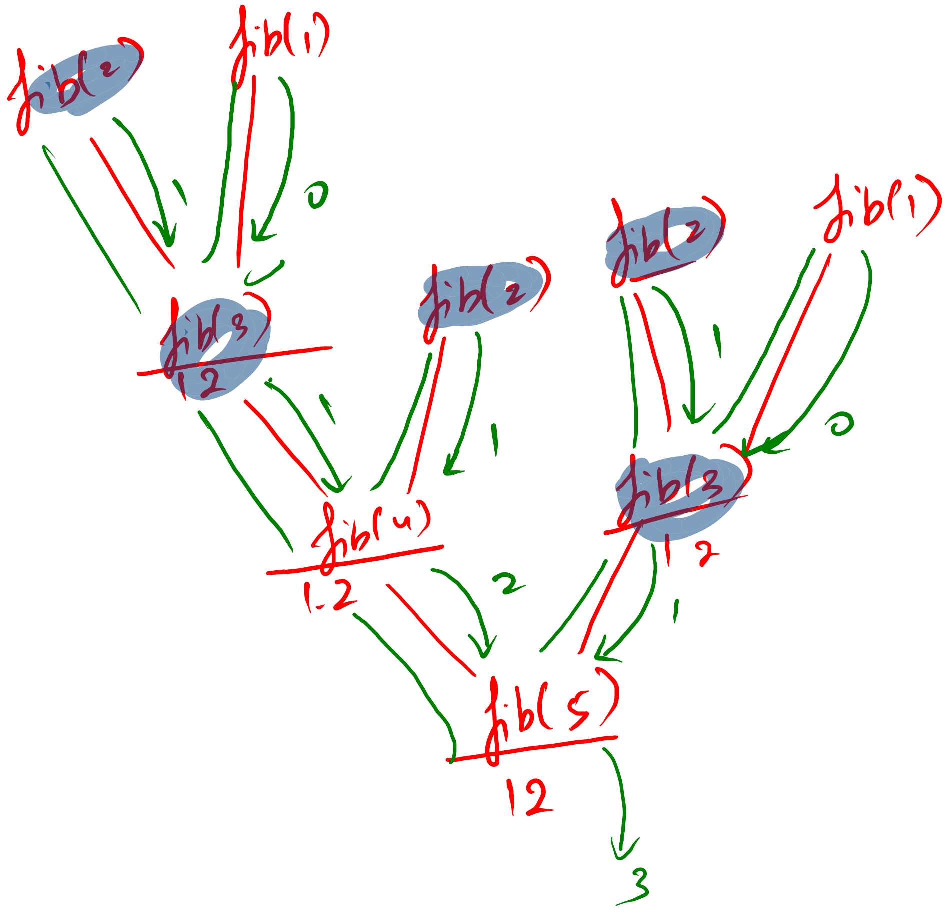
Dynamic Programming



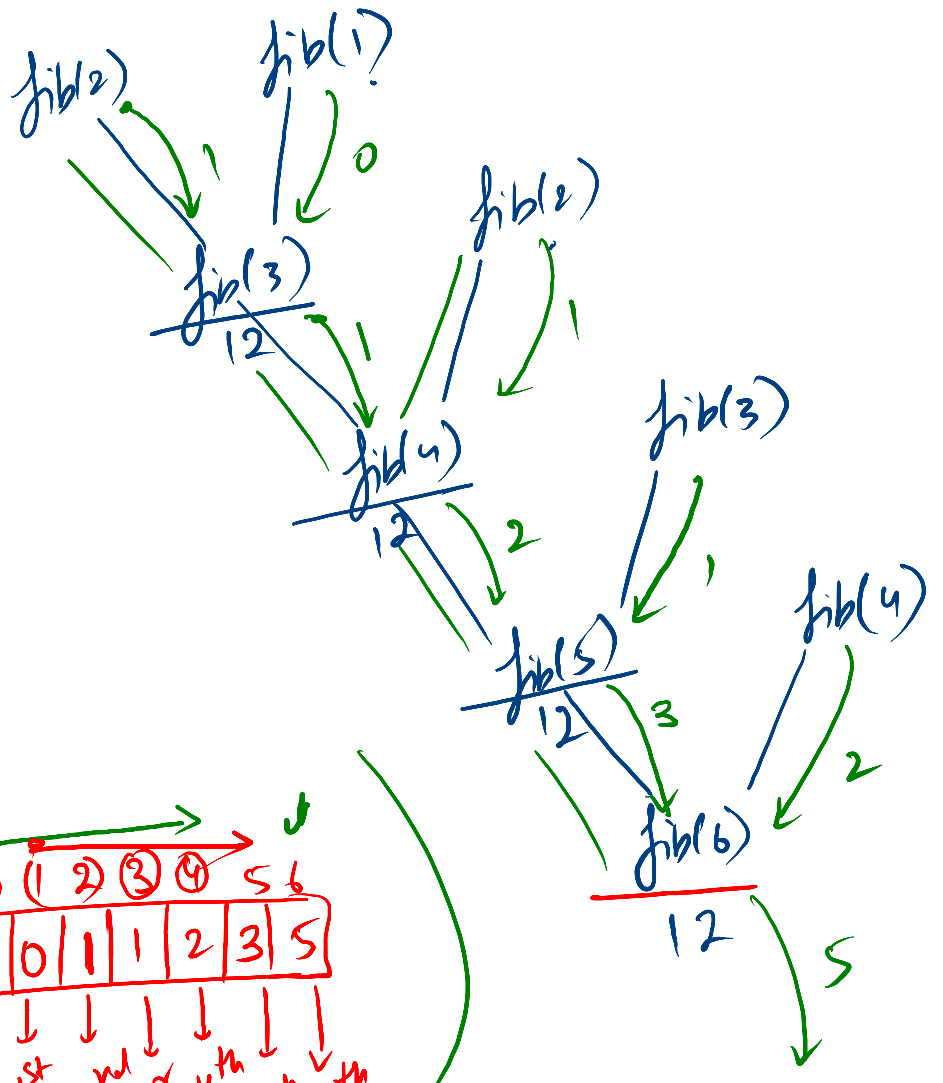
$fib(n) \Rightarrow fib(n-1) + fib(n-2)$

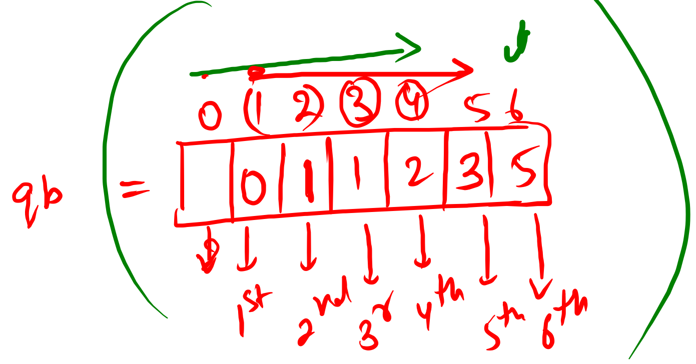
```
P  S   int  fib( int n) {
    if (n==1) return 0;
    if (n==2) return 1;

    int fibNm1 = fib(n-1);     -1
    int fibNm2 = fib(n-2);     -2
    int fibN = fibNm1 + fibNm2;

    return fibN;
}
```

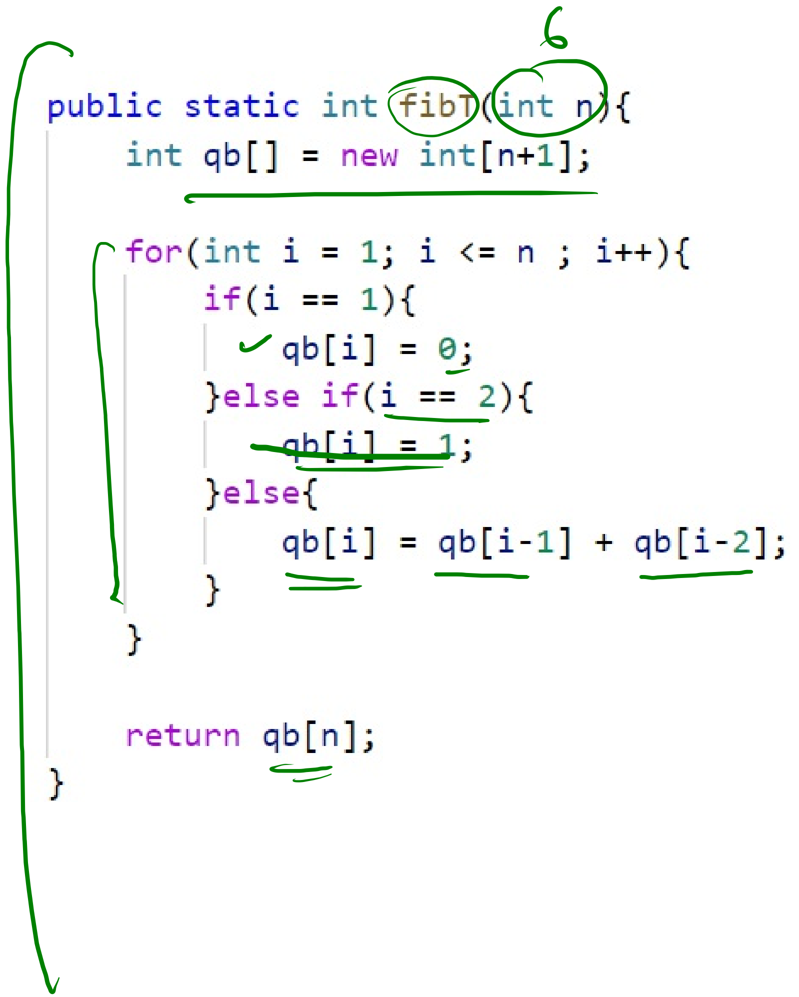




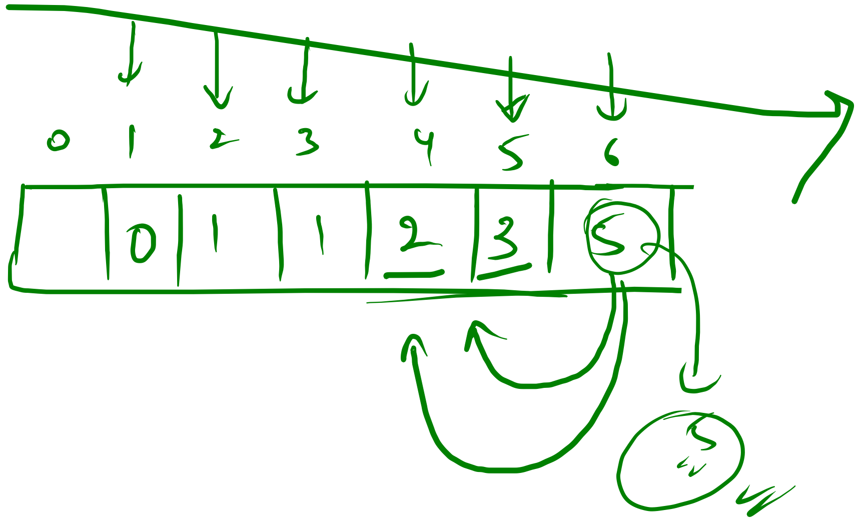
```
p    s    int    fib( int n) {

    if (n==1) return 0;
    if (n== 2) return 1;

    int fibNm1 =  fib(n-1);   -1
    int fibNm2 =  fib(n-2);   -2
    int fibN=  fibNm1 + fibNm2 ;

    return fibN;
}
```





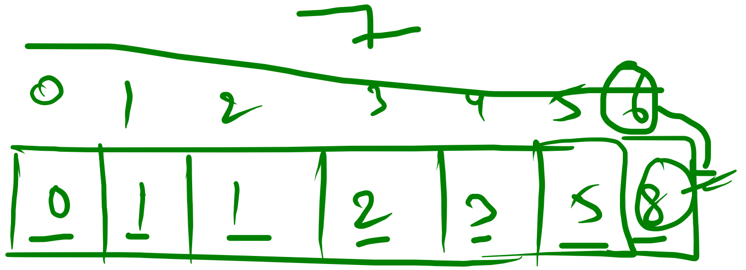


```
public static int fibT(int n){
    int qb[] = new int[n+1];

    for(int i = 1; i <= n ; i++){
        if(i == 1){
            qb[i] = 0;
        }else if(i == 2){
            qb[i] = 1;
        }else{
            qb[i] = qb[i-1] + qb[i-2];
        }
    }

    return qb[n];
}
```

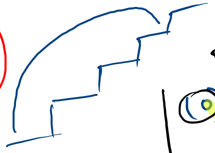


n	fib R	fib M
5	0	0
10	1	0
20	1	0
40	461	1
45	5124	1

n=6



Count



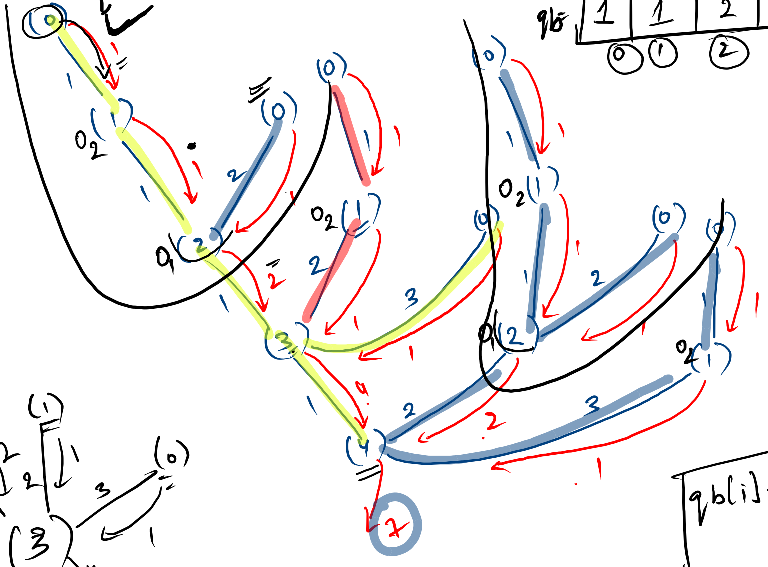
4

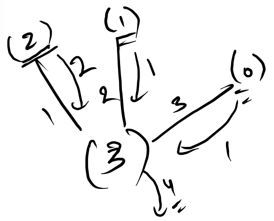
Samp

7

1 len, 2 len, 3 len ✓

qb	1	1	2	4	7
	0	1	2	3	4

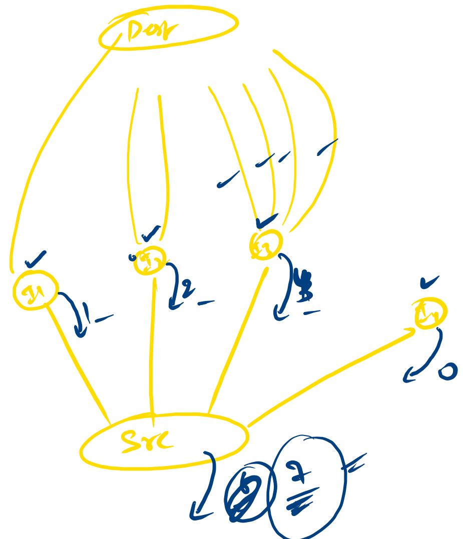




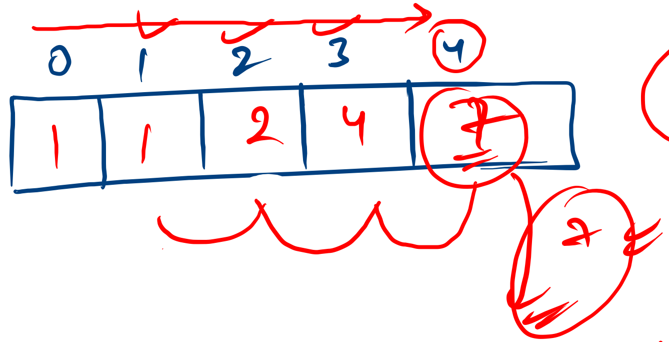
[R(D)
m ✓
T(n.w)]

[R(D)
m(n.w)
T ✓]

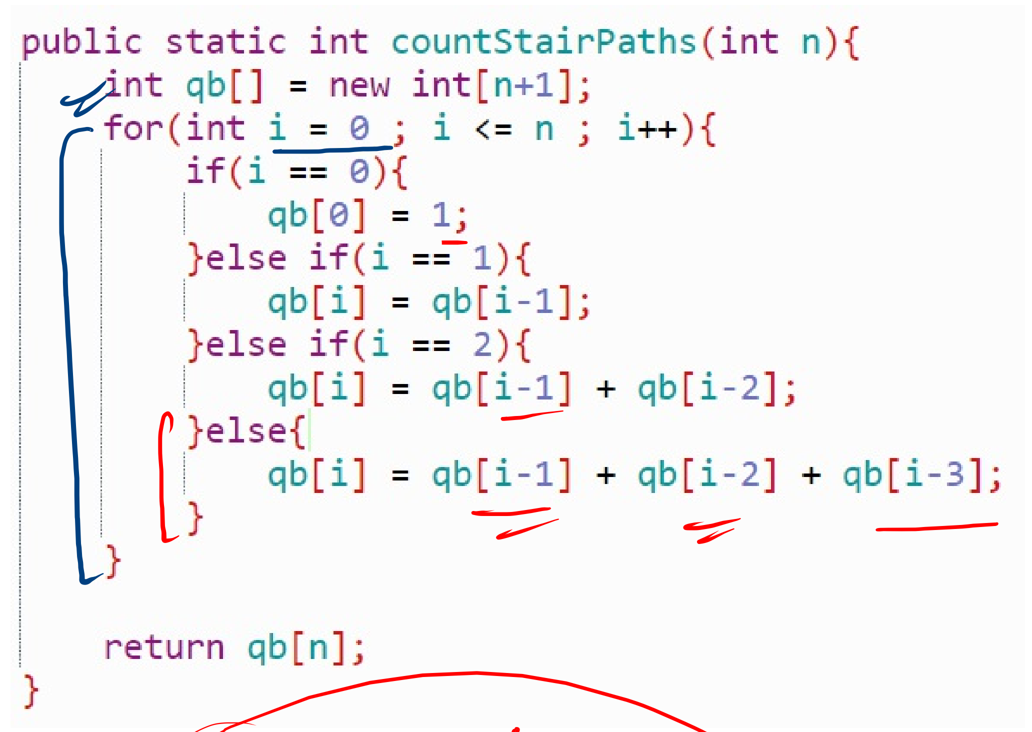
qb[i] => no. of ways to climb ith stairs



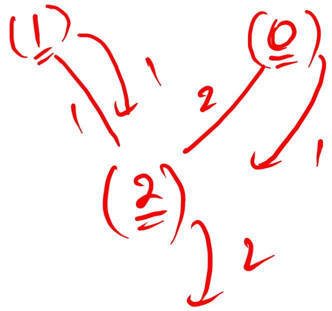
4

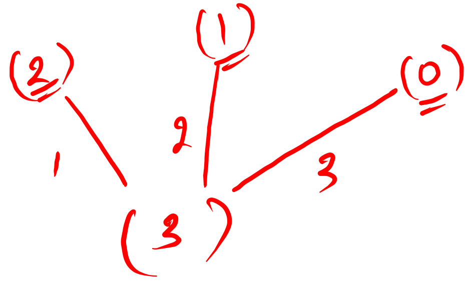


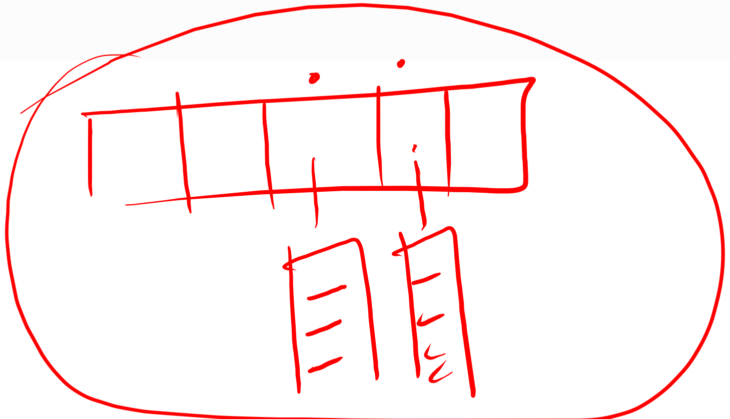
Tabulation



```
public static int countStairPaths(int n){
    int qb[] = new int[n+1];
    for(int i = 0 ; i <= n ; i++){
        if(i == 0){
            qb[0] = 1;
        }else if(i == 1){
            qb[i] = qb[i-1];
        }else if(i == 2){
            qb[i] = qb[i-1] + qb[i-2];
        }else{
            qb[i] = qb[i-1] + qb[i-2] + qb[i-3];
        }
    }

    return qb[n];
}
```





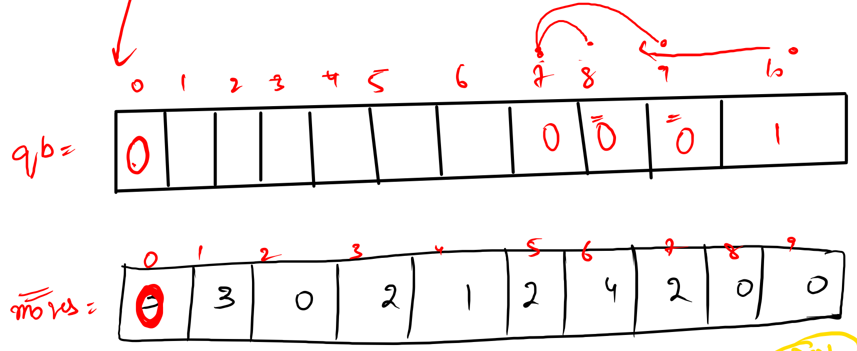


6 → 10

0 → 10

qb[i] ⇒ no. of path to reach dest from current stair, pathCount ⇒ ith stair → dest[10th]

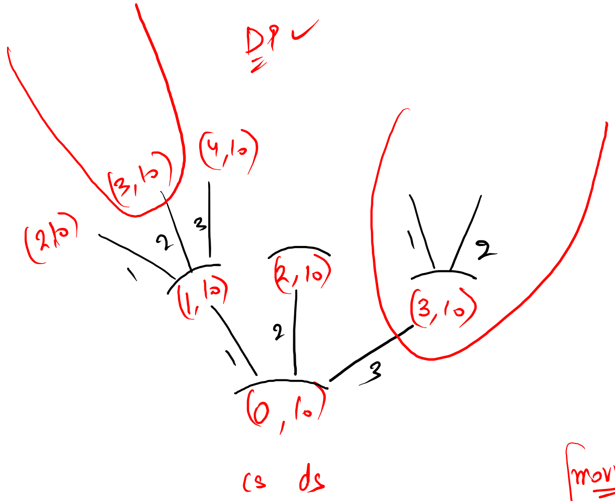
= 1 = "''"

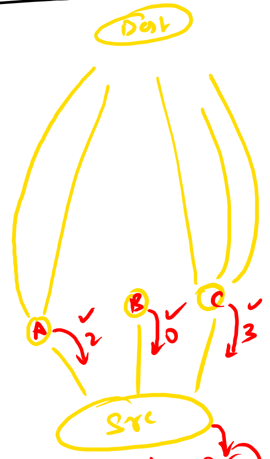


```
for(int i = n ; i >= 0 ; i--){
    if(i == n){
        qb[n] = 1; // base case
    }else{
        int maxJmp = moves[i];
        if(maxJmp == 0){
            qb[i] = 0;
        }
    }
}
```

```
10
3
3
0
2
1
2
4
2
0
0
```

DP ✓





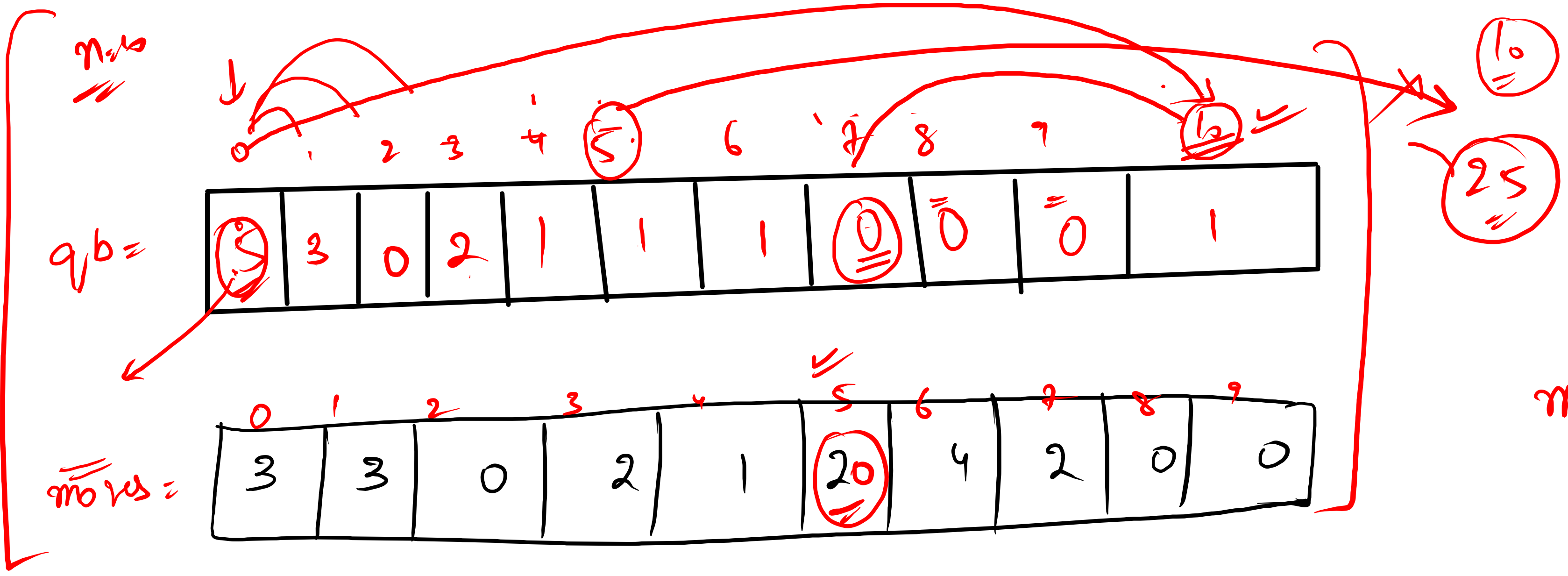
moves[i] ⇒ max jump from ith stair

→ Strg = (Meaning)

→ direction of solution

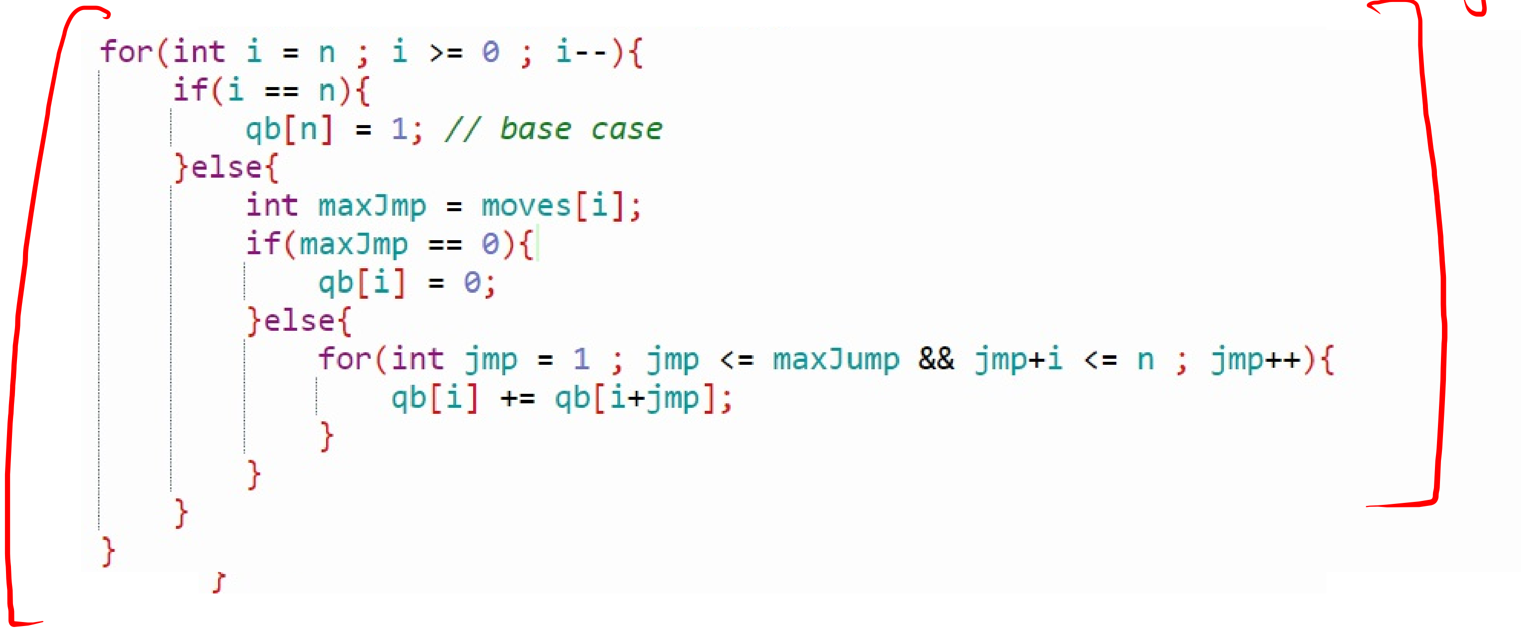
Easy → Diff

→ Traverse & solve.



max Jump = 3

jump = 1, 2, 3


```
for(int i = n ; i >= 0 ; i--){
    if(i == n){
        qb[n] = 1; // base case
    }else{
        int maxJmp = moves[i];
        if(maxJmp == 0){
            qb[i] = 0;
        }else{
            for(int jmp = 1 ; jmp <= maxJump && jmp+i <= n ; jmp++){
                qb[i] += qb[i+jmp];
            }
        }
    }
}
```

0	1	2	3	4	5	6	7	8	9
3	3	0	2	1	2	4	2	0	0

= move

H.W.

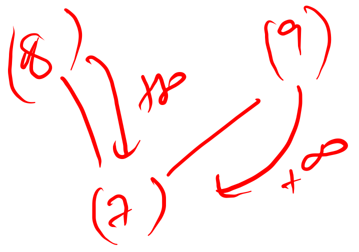
S: 0

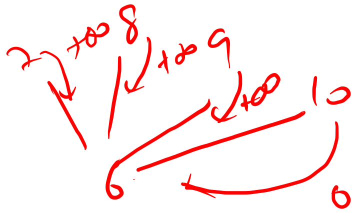
0 → Dest → 0 steps

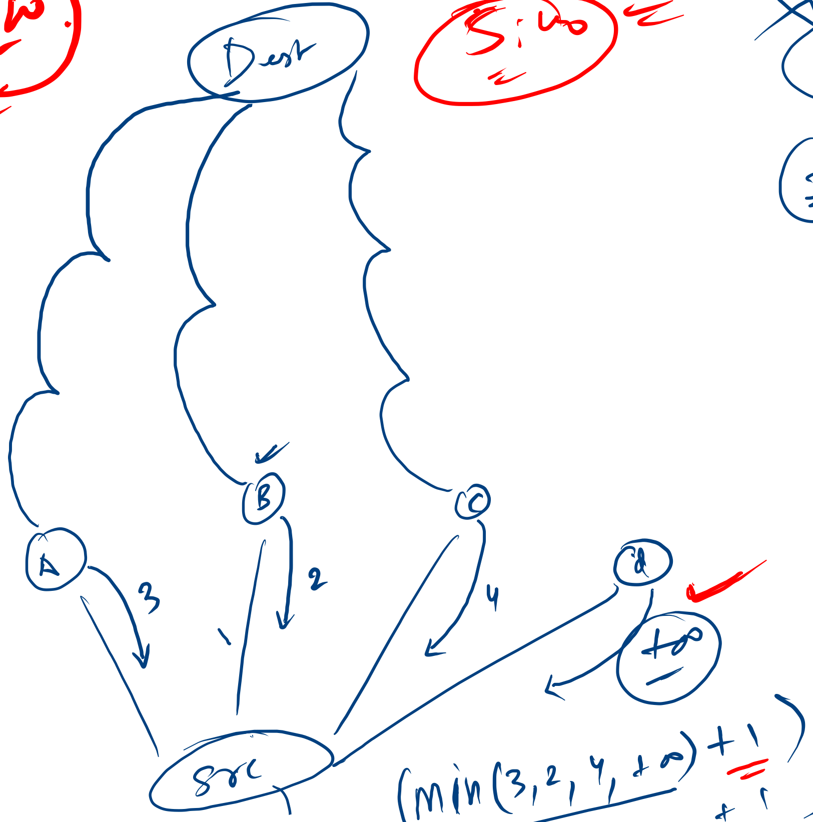
+∞

0	1	2	3	4	5	6	7	8	9	10
4	4	+∞	3	3	2	1	+∞	+∞	+∞	0

Ans







$\min(3,2,4,+\infty)+1$

2 + 1

3