merge sort $\rightarrow$ $n \log(n)$

quick n^2

count $\rightarrow$ $O(n + k)$

stable sorting

"abc" "abd" "ABC",

abc, ABC, abd

ABC, abc, abd

0 ... 9

arr 3' 4' 7 3'' 2 4'' 6

max min

size

$\boxed{max-min+1}$

freq →

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

i

d

indFreq

ind sorted

O(n)

map

$\boxed{data-min}$

$\boxed{5-2 \to 3}$

7-min = 2 + 5 freq

0	1	2	3	4	5
~~1~~ 0	~~3~~ 2	~~5~~ ~~4~~ 3	5	~~6~~ 5	~~7~~ 6
2	3	4	5	6	7

O(n)

3 → 2

4 → 2

0 → 1

2 → 1

1 → 1

7 → 1

sorted →

0	1	2	3	4	5	6
2	3'	3''	4'	4''	6	7

is stable

5
7 -2 4 1 3

```
int freq[] = new int[max-min+1];

for(int val: arr){
    int ind = val-min;;
    freq[ind]++;
}

for(int i=1;i<freq.length;i++){
    freq[i] += freq[i-1];
}

int sorted[] = new int[arr.length];

for(int i=arr.length-1;i>=0;i--){
    int data = arr[i];
    int freqInd = data-min;
    int sortedInd = freq[freqInd]-1;
    sorted[sortedInd] = data;
    freq[freqInd]--;
}

for(int i=0;i<arr.length;i++){
    arr[i] = sorted[i];
}
```

1-(-2) = 3

1

3

1

	0	1	2	3	4	5
arr =	7	-2	4	1	3'	3''

6 3 +2 0

min = -2

max = 7

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

	0	1	2	3	4	5	6	7	8	9
freq =	1	1	1	~~2~~ 1	2	~~4~~ ~~3~~ 2	5	5	5	6

	0	1	2	3	4	5
sorted		1	3'	3''		

-2 1 3' 3'' 4 7

data 0 ... 10^{18}

arr = [12 13 7 8 9 4 20 10001]

$\downarrow$

[0, 10^{18}]

[4 7 8 9 12 13 20 10001]

7 3 4 2 6 8

7 6 5 4 2 1

3
2
4
7

2
3
4
7

5 7
1 2
2 3
4 7
3 2
1 1

8
1 1
1 2
3 2
2 3
5 7
4 7

1 1
1 2
2 3
3 2
4 7
5 7

2 3
1 3
4 2
4 1

1 3
2 3
4 2
4 1

1 2 3 4
3 7 8 9

2 3
1 3
4 2
4 1

1 3
2 3
4 2
4 1

0023
0013
0042
0041

1000

1000
41
42
23
13

1000
0013
0023
0041
0042

0 23
0 41
0 42
0 13

1000

0013
0023
0041
0042
1000

data → num/exp % 10

100

10 10 1

102345

000234

exp → 100

102 3~~45~~

$$\frac{102345}{100} =$$

$1023 \% 10 = \boxed{3}$

max

places

300 → 10 lakh

T

n log(n)

tuk

3 4 7 9

0 1 2 3 4 5 6 7

0 1 3 4

0 1 2 3 4

1 1 0 1 1

15 Lakh → 100

T

100x log(100)

5
12041996
20101996
05061997
12041989
11081987

D	M	Y
12	04	1996
20	10	1996
05	06	1997
12	04	1989
11	08	1987

old

11	08	1987
12	04	1989
12	04	1996
20	10	1996
05	06	1997

new

0 2502

DD	MM	YYYY
11	10	2000
12	10	1958

D	MM	
12	10	1958
11	10	2000

11	10	2000
12	10	1958

1000
1000
1000

2000
2000

100 10 1

0 2 3 4

2

exp = 1 ✓

= 10 ✓

100 ✓

1000 ✓

$\boxed{\text{num/exp} > 0}$

Recursion → problem $\xrightarrow{\text{solve}}$ DP

↓

fibonacci

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

0 1 2 3

0 1 1 2 3 5 8 . . . . .

qb

2 → 1

fib(n)

dp

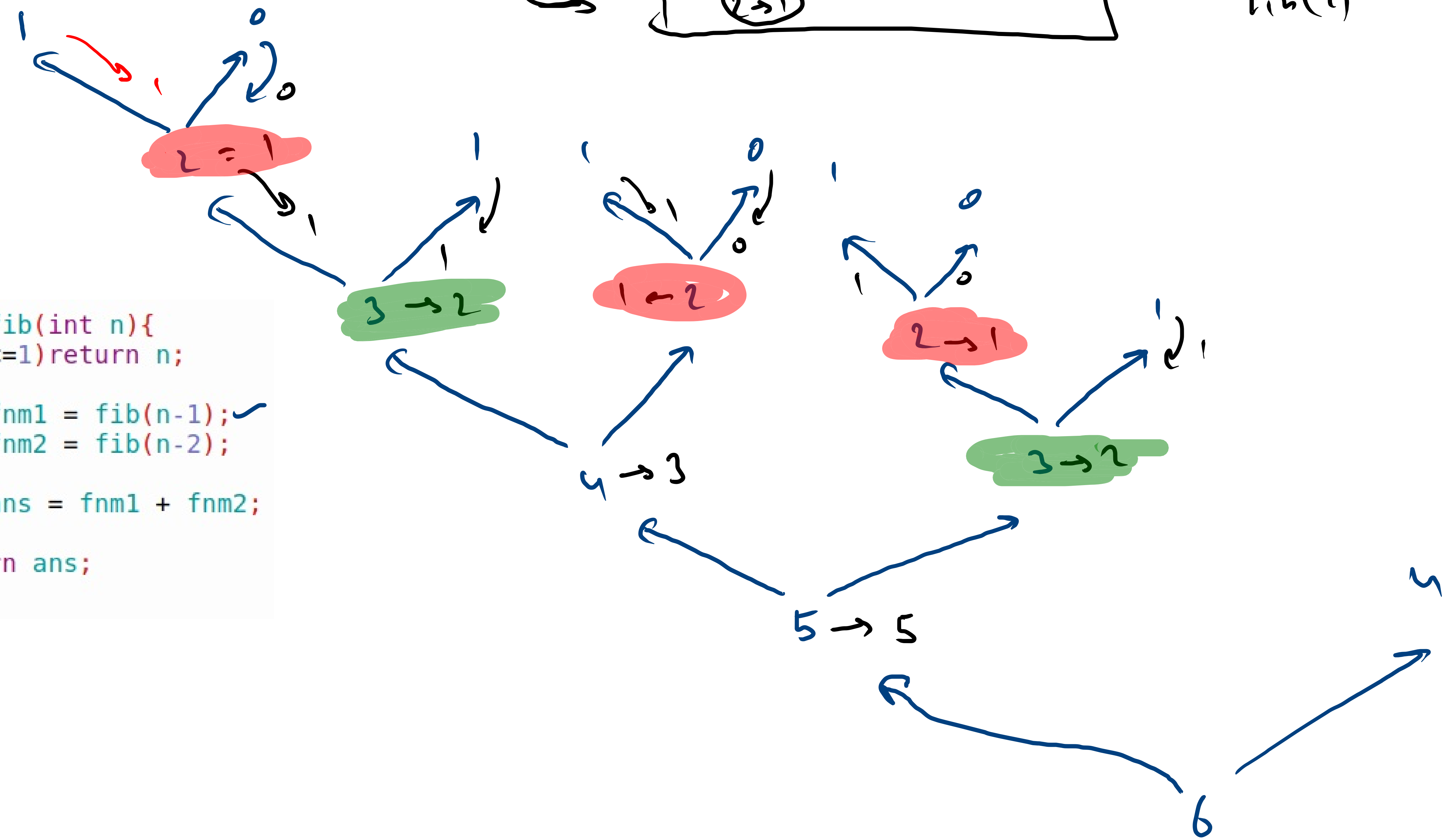
Recursion

```java
static int fib(int n){
    if(n<=1)return n;

    int fnm1 = fib(n-1);
    int fnm2 = fib(n-2);

    int ans = fnm1 + fnm2;

    return ans;
}
```



2 → Memo
→ Tabul

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

4

0 1 2 3 4

0 1 1 2 3

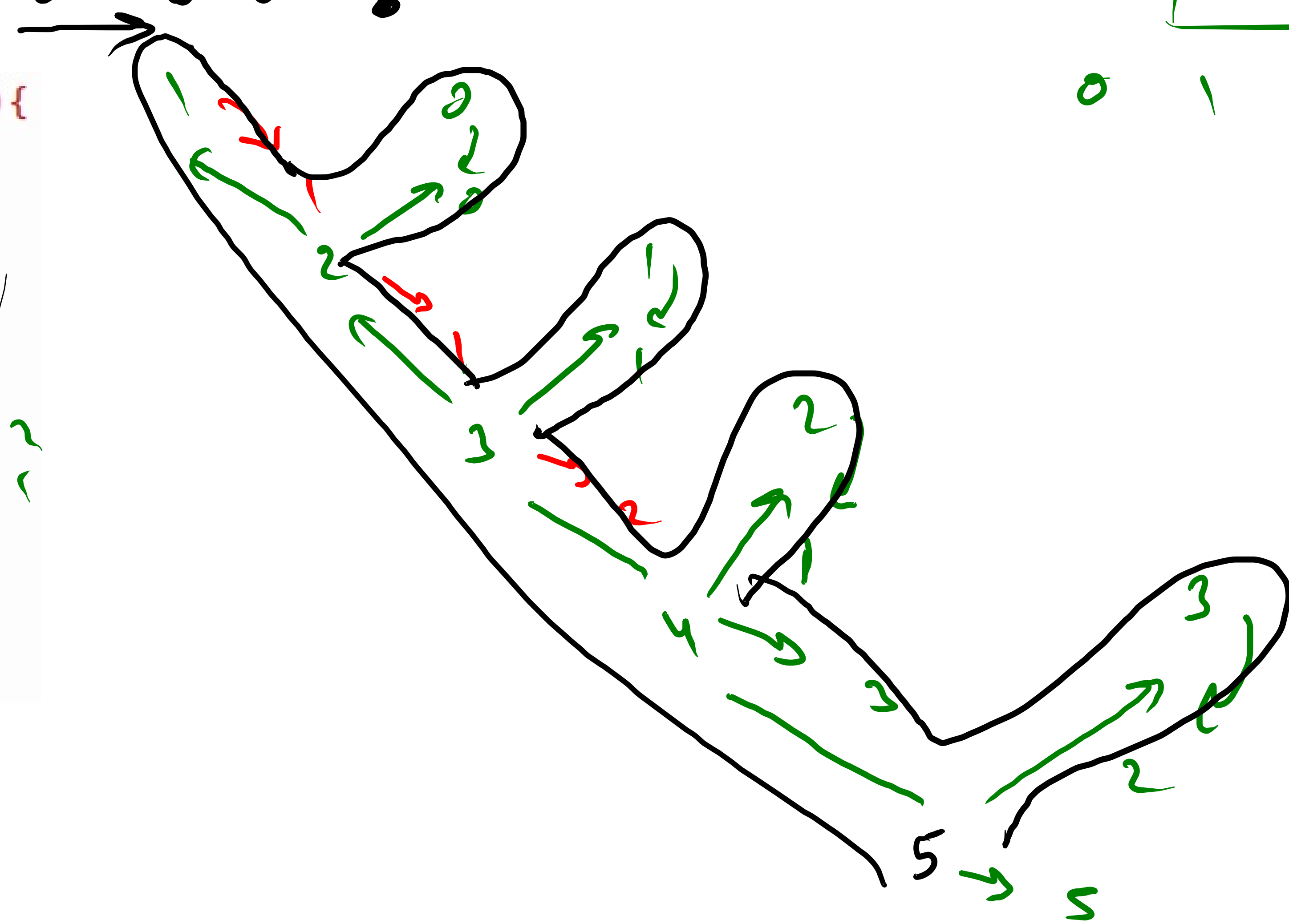
O(n)

```java
static int fibM(int n, int qb[]){
    if(n<=1)return n;

    if(qb[n] > 0){
        return qb[n];
    }

    System.out.println(n);
    int fnm1 = fibM(n-1, qb);
    int fnm2 = fibM(n-2, qb);

    int ans = fnm1 + fnm2;
    qb[n] = ans;
    return ans;
```



5
4
3
2

big small

$$T(n) = \underbrace{T(n-1)} + \underbrace{T(n-2)}$$

$$T(n-1) + T(n-1) > \boxed{T(n)} > T(n-2) + T(n-2)$$

$\approx 2^n$

$\approx 2^n$

$$\overset{n-0}{T(n)} = \cancel{2T(n-1)} + 1k \qquad (n+1) \text{ Times}$$

$$\cancel{2T(n-1)} = 4T(n-2) + 2k \qquad 2^n$$

$$4T(n-2) = 8T(n-3) + 4k$$

$$\vdots \qquad 8k$$

$$\vdots \qquad 16k$$

$$n\,\overset{n-n}{T(0)} = O(1)$$

0

1

2

⋮

n

$\boxed{4}$

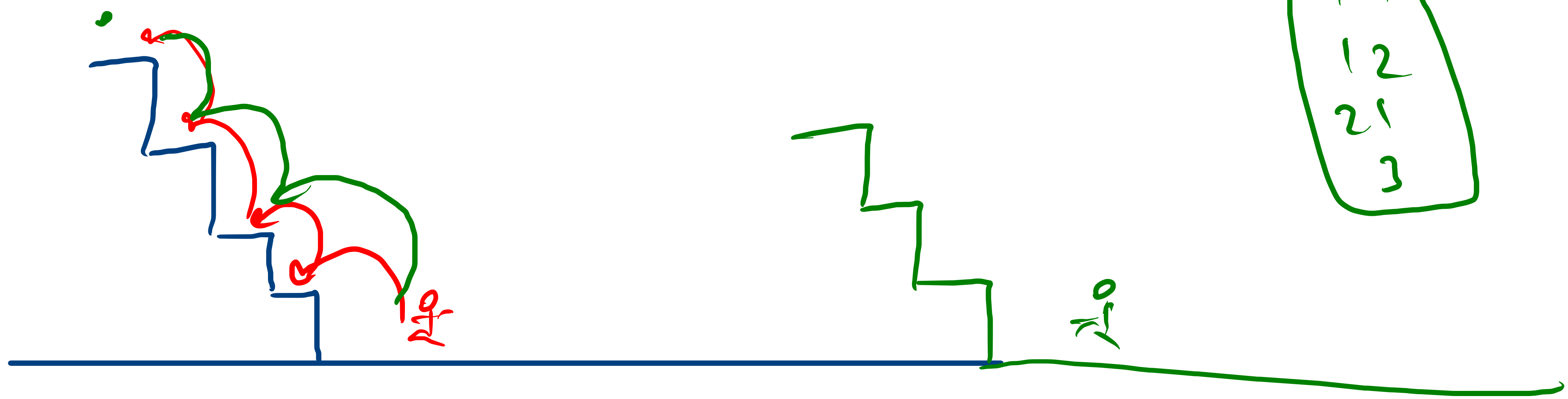
Jump = 1, 2, 3

1 1 1 1

2 1 1

$\boxed{\begin{matrix} 1\,1\,1 \\ 1\,2 \\ 2\,1 \\ 3 \end{matrix}}$

$\boxed{4}$



n

0	2'	2''	1

0	1	2
1	2	~~x~~

~~3~~

2

0	1	2	3
0	1	2	2

2' 2''

2'' 2'

↓

a b c d

1

2

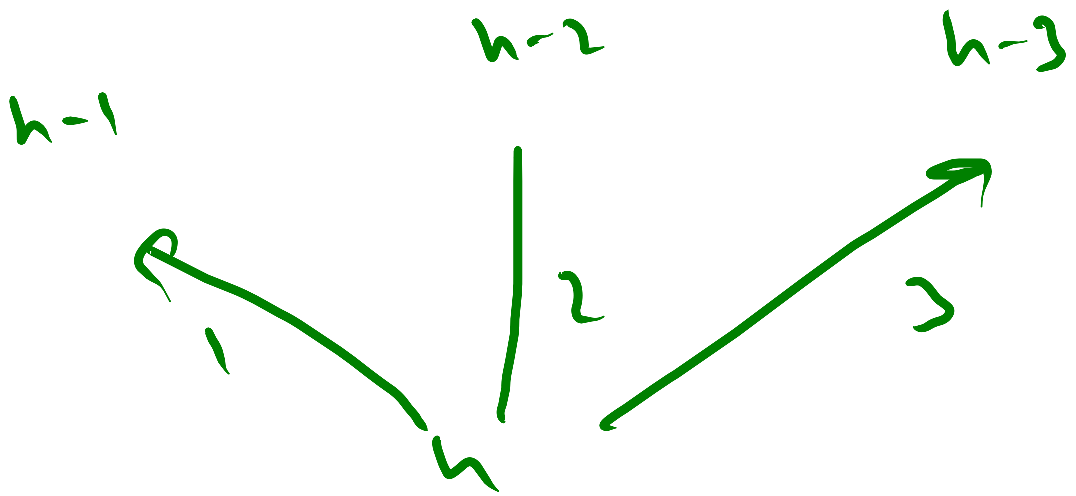
0 1 2 3 4

a b c d e

↑

5

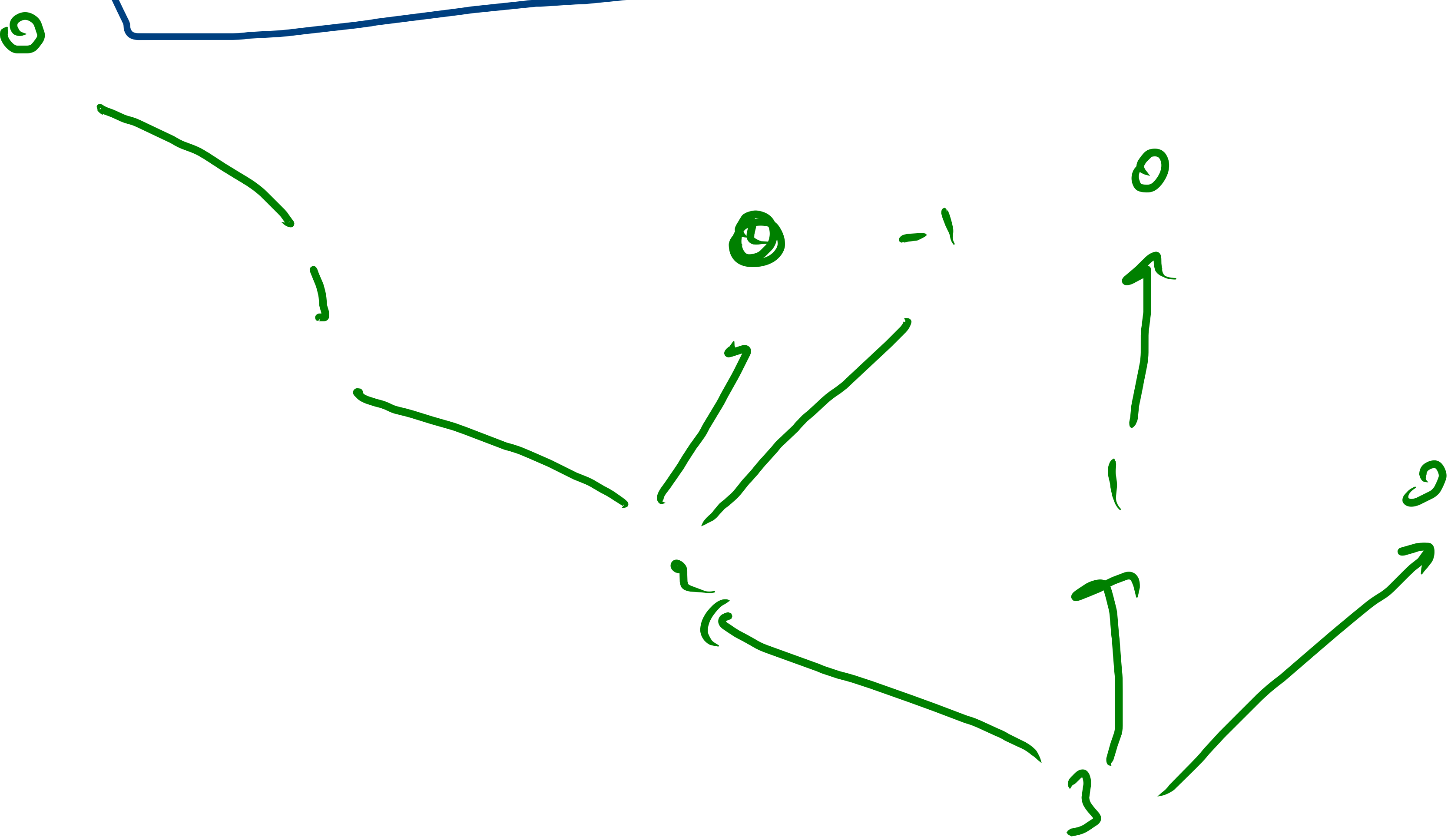
dp $\boxed{n = 0}$



$f(n) = f(n-1) + f(n-2) + f(n-3)$

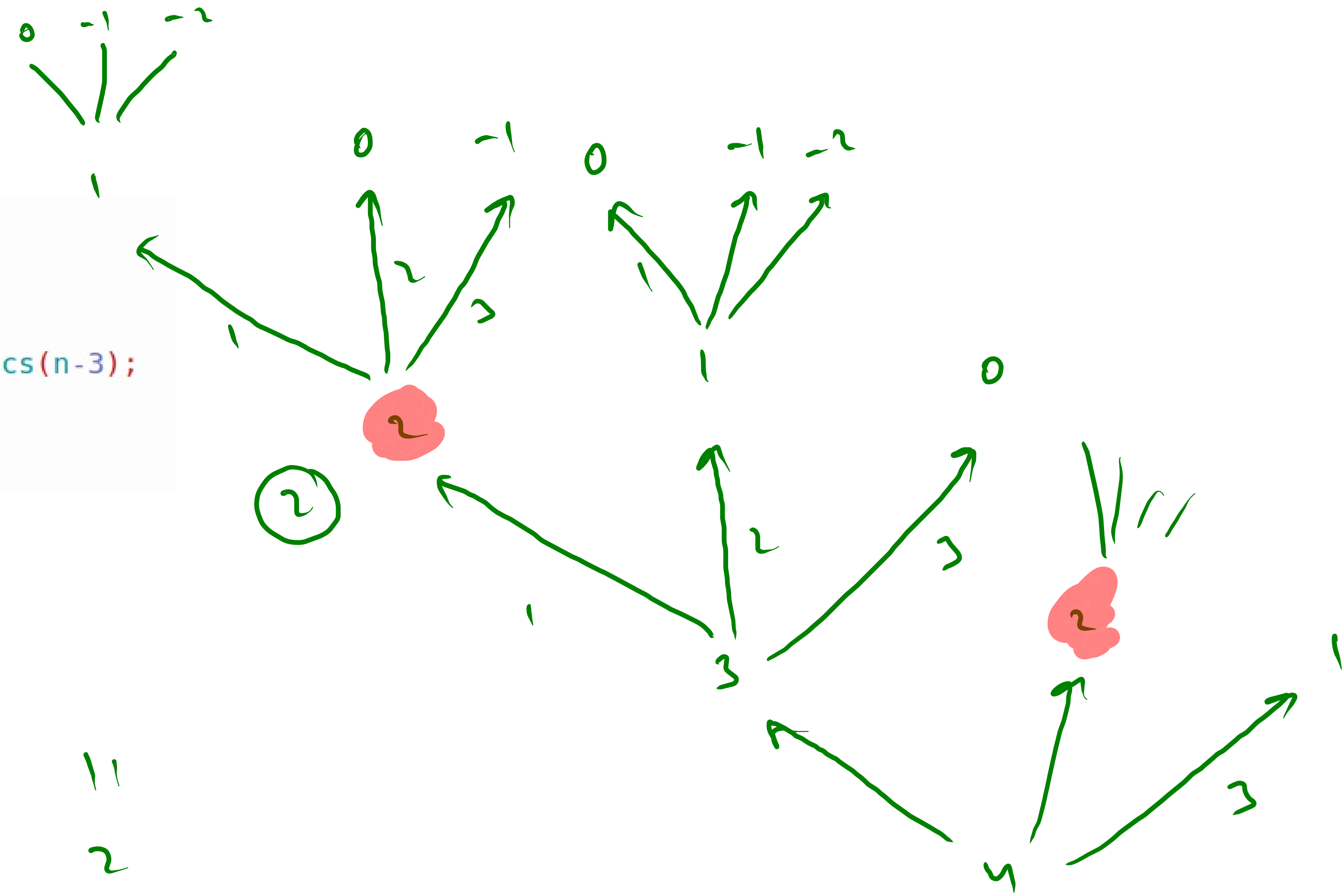
n == 0 return 1

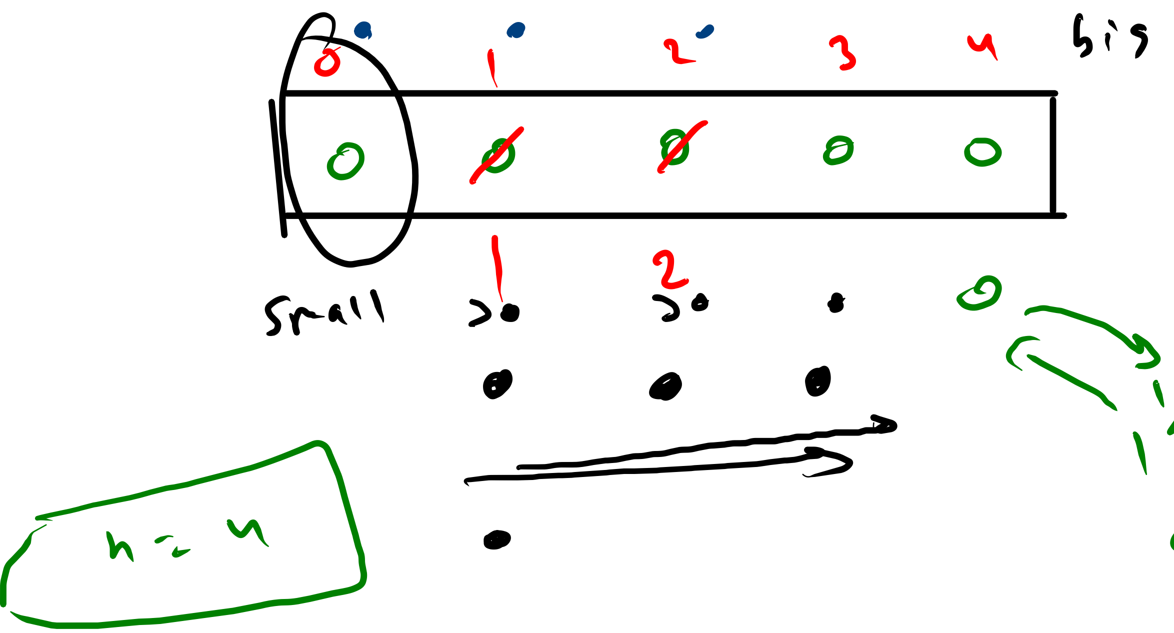
n < 0 return 0

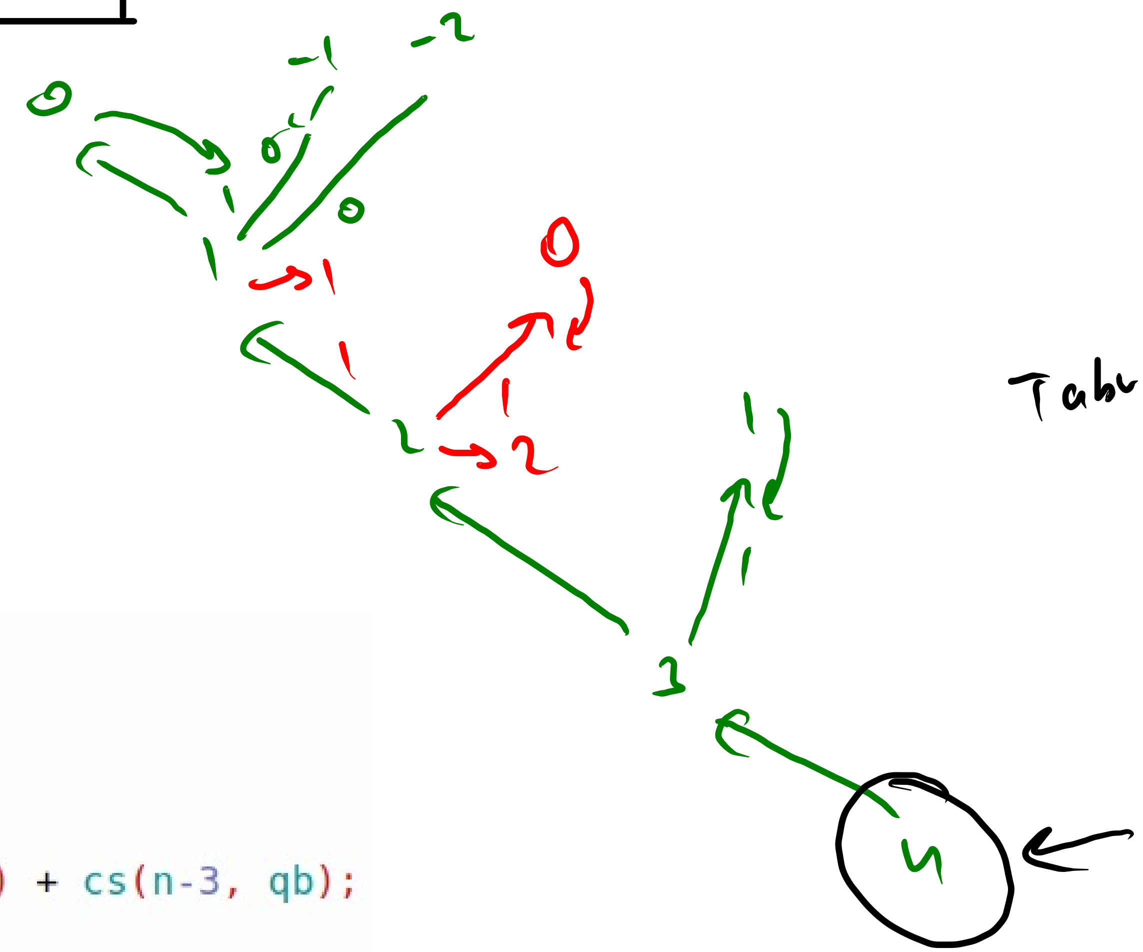


```java
static int cs(int n){
    if(n==0)return 1;
    if(n<0)return 0;

    int ans = cs(n-1) + cs(n-2) + cs(n-3);
    return ans;
}
```







recursion

big → small

Top → bottom

Tabulation → small → bottom

big

Top

big

```
static int cs(int n, int qb[]){
    if(n==0)return 1;
    if(n<0)return 0;

    if(qb[n]>0)return qb[n];
    int ans = cs(n-1, qb) + cs(n-2, qb) + cs(n-3, qb);
    qb[n] = ans;
    return ans;
}
```

mimo → Tab

Direct Tabulation

storage

direction

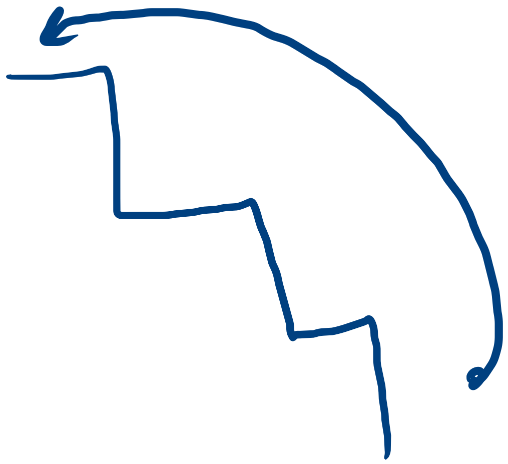
meaning =

travel and solve

1, 2, 3

k

1



1 2 3

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

10 3 3 0 2 1 2 4
2 0 0

1 2 3