Fibonicci Series

$$\begin{array}{ccccccc} & & & & \downarrow & & \\ 1 & 1 & 2 & 3 & 5 & 8 & 13 \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

$$\text{Fib}(n) = \underbrace{\text{Fib}(n-1)}_{\text{last}} + \underbrace{\text{Fib}(n-2)}_{\text{second last}}$$

PMI - Extended Form

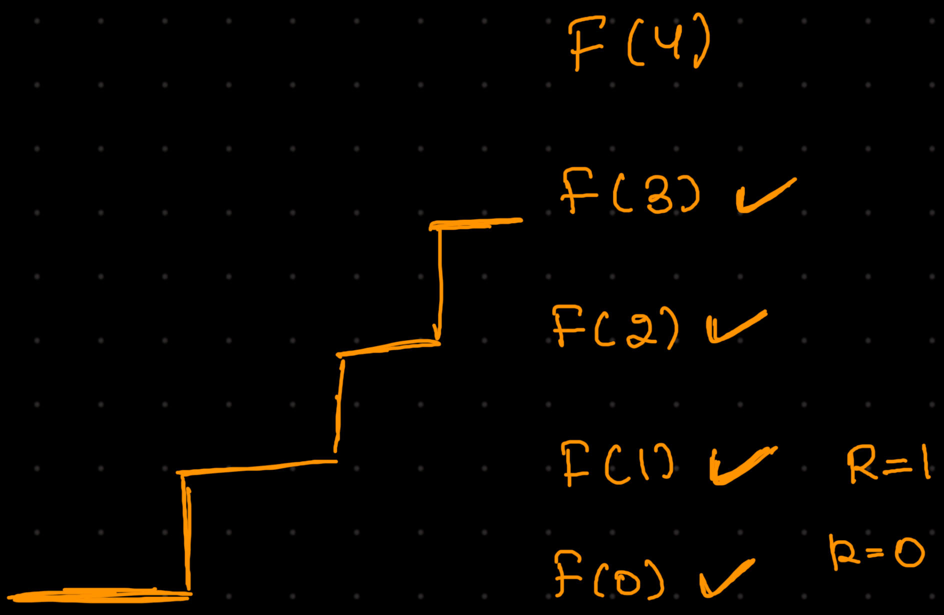
1. Proof for base case F(0)   F(1)

2. Assume for F(i) equal to true where your i can be from

$0 \le i \le R$ F(R)

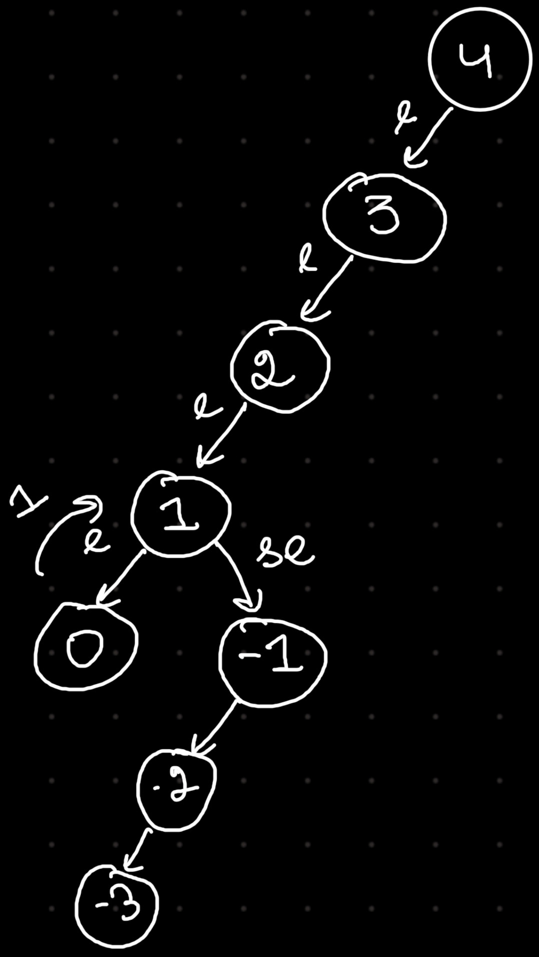
so thot means we can assume for

F(R) true
F(R-1) true } ⇒ all assume true for less than n

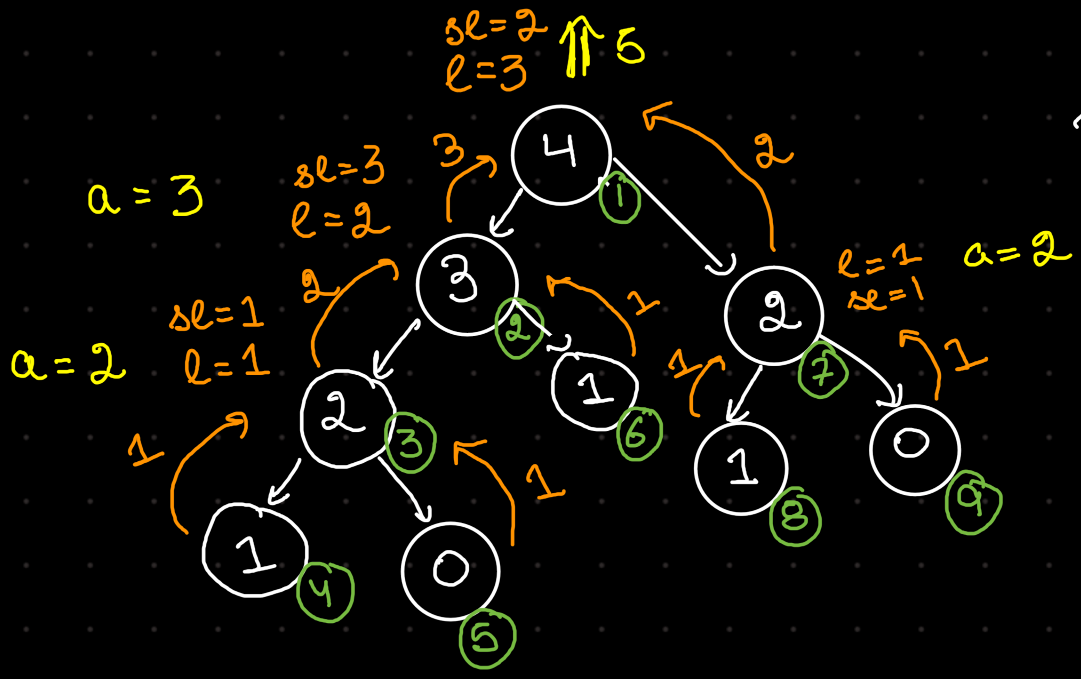


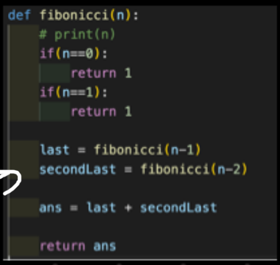
last F(n-1) to be true

second last F(n-2) to be true


4
l
3
l
2
l
1
1
l
se
0
-1
-2
-3

Fibonicci Series : Recursion Tree



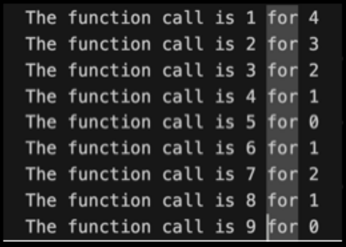


```
def fibonicci(n):
    # print(n)
    if(n==0):
        return 1
    if(n==1):
        return 1

    last = fibonicci(n-1)
    secondLast = fibonicci(n-2)

    ans = last + secondLast

    return ans
```

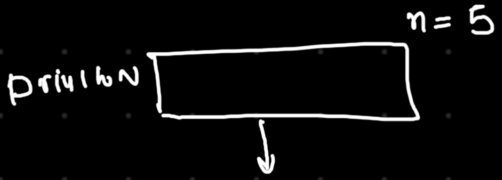


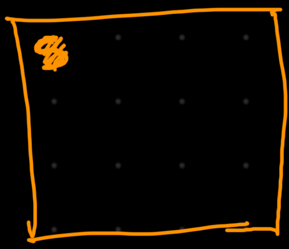
```
The function call is 1 for 4
The function call is 2 for 3
The function call is 3 for 2
The function call is 4 for 1
The function call is 5 for 0
The function call is 6 for 1
The function call is 7 for 2
The function call is 8 for 1
The function call is 9 for 0
```

Q: Write a program for a given number n.

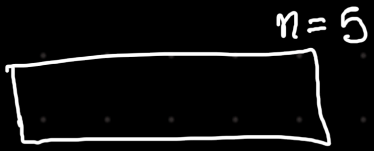
1. Print 1 to N.
2. Print N to 1

Print 1 to N :-





Print N to 1



1. Base Case
2. Recursive call ↺
3. Our work

Head

Tail

Assignment

1. Sum of Digit of a number

2. Power of a number (<u>base</u>, exp)
3 2

Head vs Tail Recursion

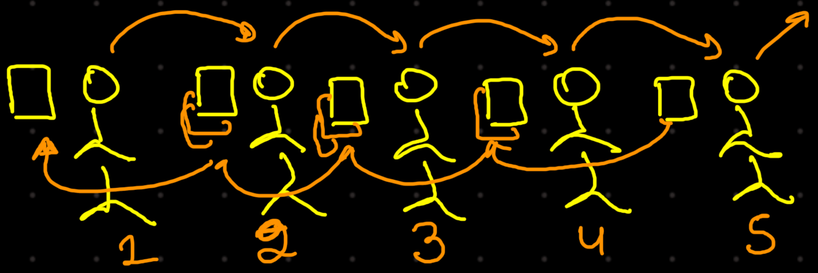
Head

we make recursive call at the beginning of our fn implementation.

Tail

when we make recursive call at the end of our implementation

head



tail