Recursion

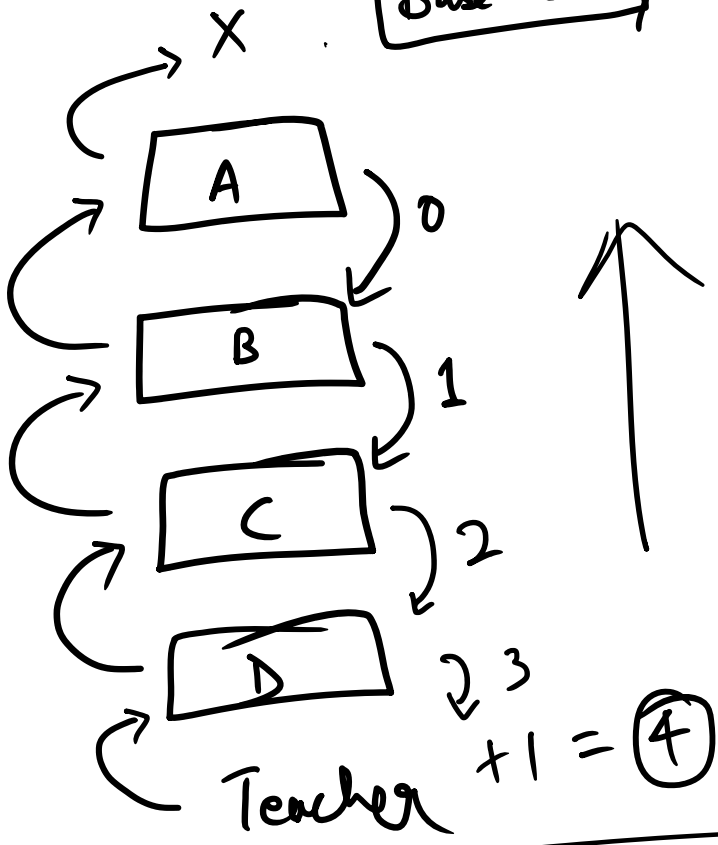
smaller instance → same problem

xyz: pqr

pqr: abc

abc:

Base case / Terminating / Stopping



"how many students are there in front of you?"

Function Execution

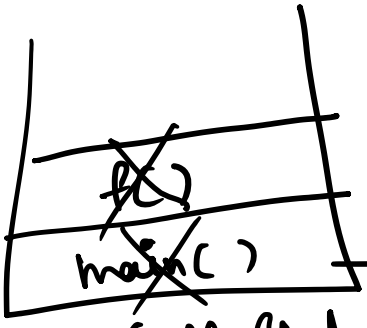
O/p: Start
f

Function

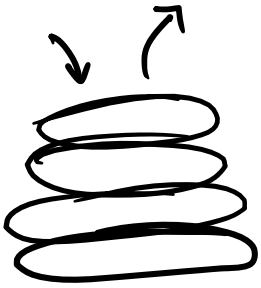
```
main ( )
{
   print (Start)
   f( )
→ print( End)
}
```

```
f( )
{
→ print(f)
}
```

```
f
End
```



Call Stack → AR activation record

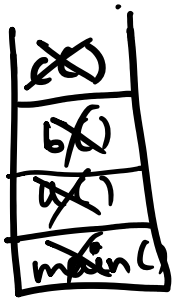


```
main ( )
{
   print (main)
→ a( )
}
```

```
a( )
{
   print(a)
→ b( )
}
```

```
b( )
{
   print(b)
   c( )
}
```

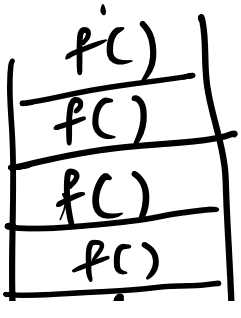
```
c( )
{
→ print(c)
}
```



O/P: main
a
b
c

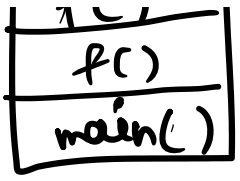
Recursive Function

```
f( )
{
   f( )
```



"Stack overflow"

```
{
  print(f)
  f()
}
```



Call Stack

f f f f

Recursive Tree

f f(4)
↓
f f(3)
↓
f f(2)
↓
f(1)
↓
f(0)

```
f(n)
{
  if(n==0)
    return

  print(f)
  f(n-1)
}
```

Base case

Recursive case

TC → O(n)

SC → O(1)

Auxiliary Stack Space → O(n)

Q. Sum of first n numbers

5 → 5+4+3+2+1

f(5) → 5+f(4)
↓

f(n)

{

```
if(n == 0)
    return 0
```

Base case

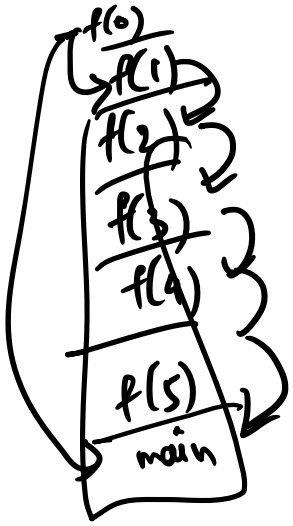
return n + f(n-1)

}

$4 + f(3)$
↓
$3 + f(2)$
↓
$2 + f(1)$
↓
$1 + f(0)$
↓
0

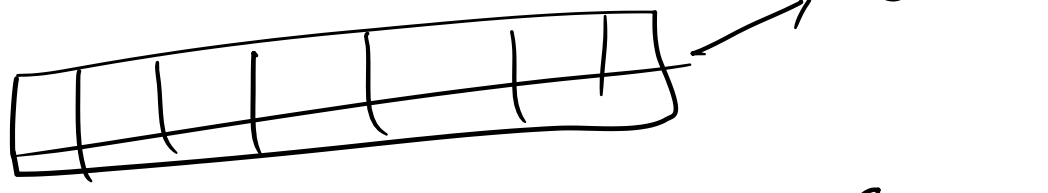
TC → O(n)

SC → O(n)



	Recursive	Iterative (loop)
TC →	O(n)	O(n) → Draw
SC →	O(n)	O(1) → Iterative

Sudoku →



Fibonacci Series

0, 1, 1, 2, 3, 5, 8, 13, ---

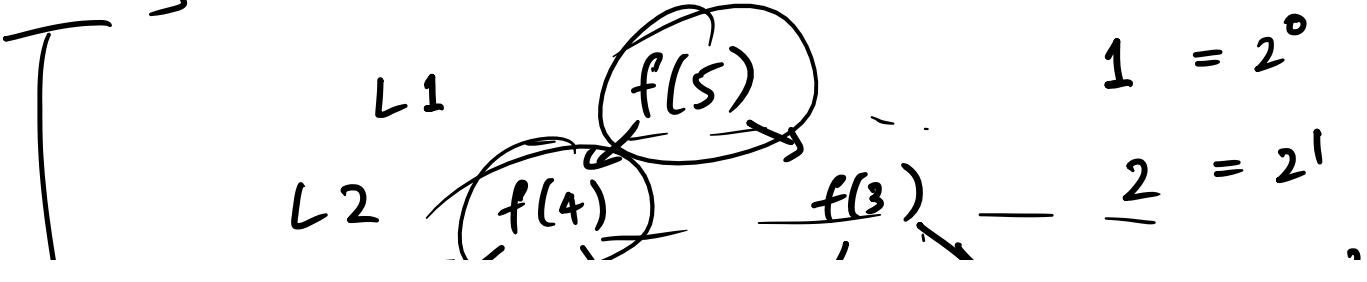
$f(0) = 0$

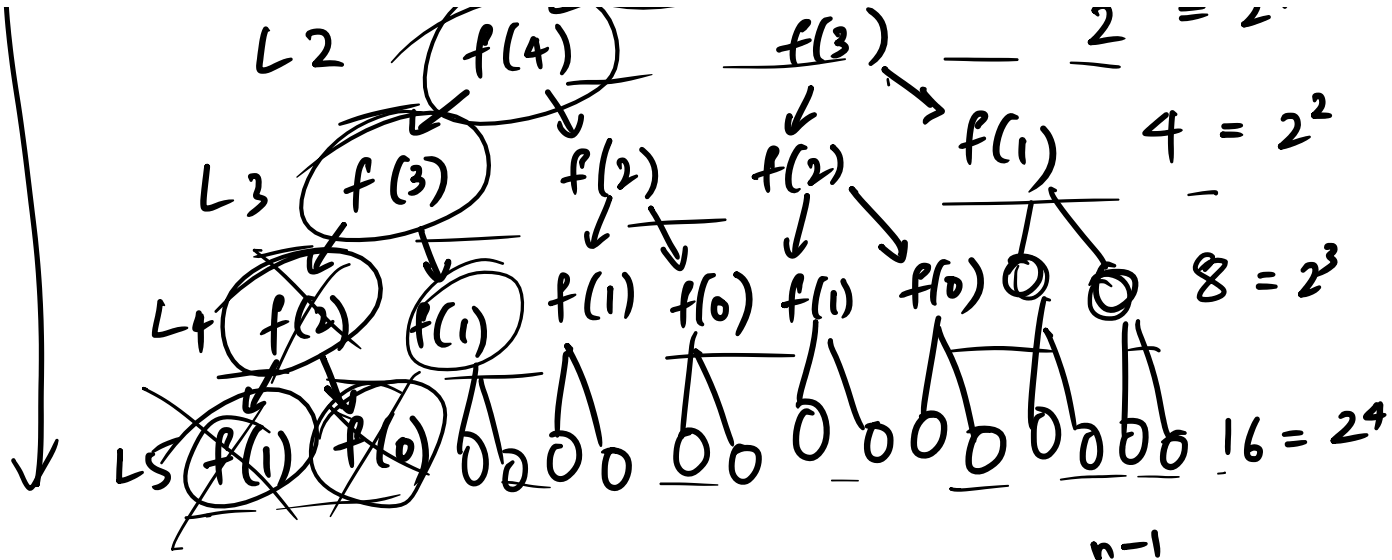
$f(1) = 1$

(I) $f(n) = n \quad | \quad n \leq 1$

(II) $f(n) = f(n-1) + f(n-2) \quad | \quad n \geq 2$

```
f(n)
{
    if (n <= 1)          } Base case
        return n
    return f(n-1) + f(n-2)   } Recursive case
}
```





$2^0 + 2^1 + 2^2 + \ldots + 2^{n-1}$

GP

$\frac{a(r^n - 1)}{r - 1}$

$\frac{1(2^n - 1)}{2 - 1} = 2^n - 1$

$\approx O(2^n)$

$n! = n \times n-1 \times n-2 \times n-3 \times \ldots 1$

$5! = 5 \times 4 \times 3 \times 2 \times 1$

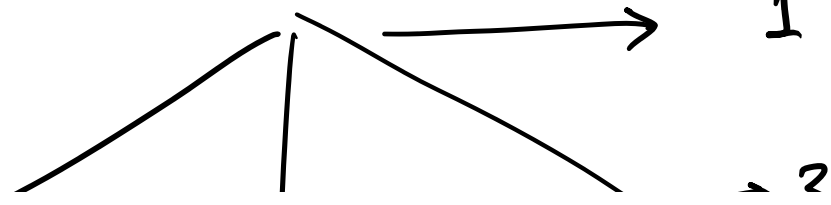
$0! = 1$

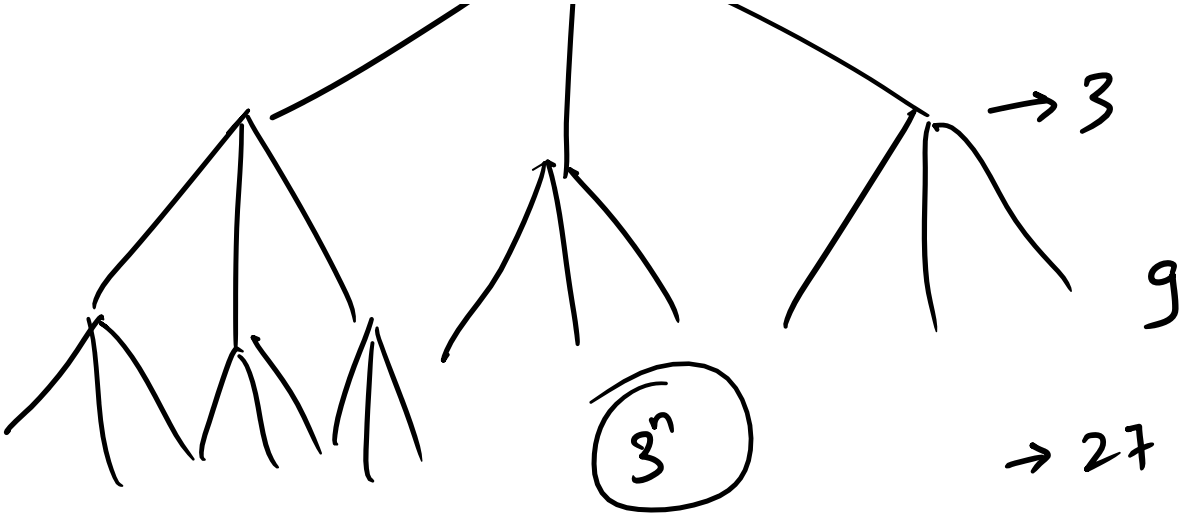
```
if(n == 0)
    return 1
```

n (+) f(n-1

(×)

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




→ 3
9
3^n
→ 27