Interview Recursion

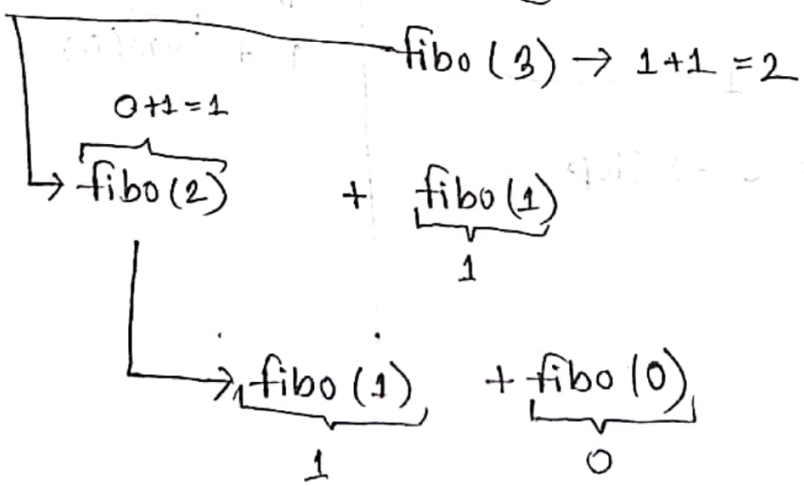
1. Fibonacci

=> 0, 1, 1, 2, 3, 5, 8, 13, 21

```
def fibo(n):
    if n in [0,1]:
        return n
    else:
        return fibo(n-1) + fibo(n-2)

print (fibo (7)) ;  ⟶ 13
```

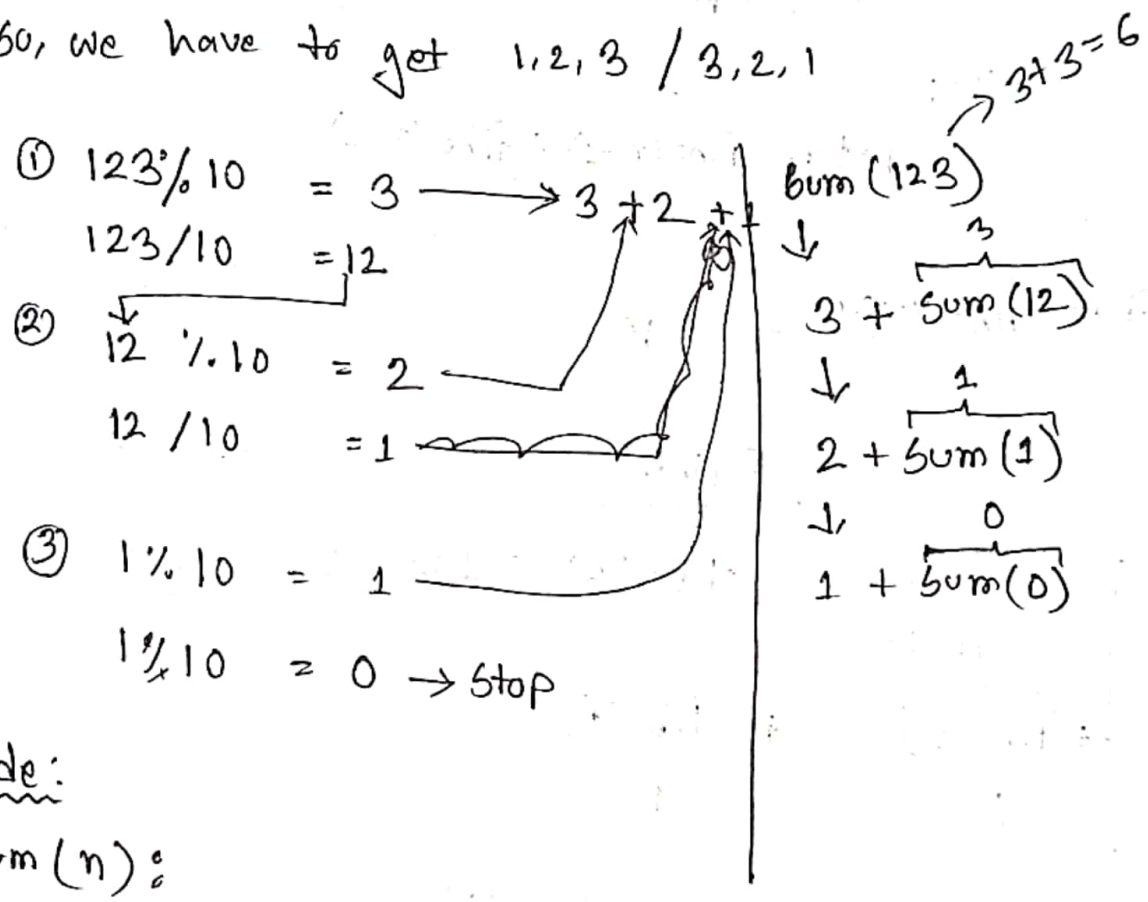
Explain



Sum of the digits of a positive int num using recursion:

⇒ suppose 123 → Ans is = 1+2+3 = 6

So, we have to get 1,2,3 / 3,2,1



Code:

```
Sum(n):
    if n==0:
        return 0
    else:
        return int(n%.10) + Sum(int(n/10))
```

Sum (123) ⟶ 6

Power of a number using recursion:

$\Rightarrow 2^4 = 2 * 2 * 2 * 2$

$\Rightarrow x^n = x * x^{n-1} = x^1 * x^{n-1} = x^{(1+n-1)} = x^n$

$\Rightarrow 2^4 = 2 * 2^3 * 2^2 * 2^1 * 2^0$

$2^1 \downarrow 2 \quad 2^0 \downarrow 1$

Code:

```
def Pow(base, exp):        # 2^4  (base = 2, exp = 4)
    if exp == 0:
        return 1
    if exp == 1:
        return base
    return base * pow(base, exp-1)

print(Pow(2,4)) ⟶ 16
```

pow(2,4) → 2 * 8 = 16

↓

2 * pow(2,3) → 2 × 4 = 8

↓

2 * pow(2,2) → 2 × 1 = 2

↓

2 * pow(2,1)

2

pow(2,0) :

= 1

48 % 18 = 12
18 % 12 = 6
12 % 6 = 0
6 % 0 = 0

6

Find GCD of two numbers using Recursion.

=> gcd (8, 12) = 4

8 = 2 * 2 * 2

12 = 2 * 2 * 3

Euclidean algorithm:

48 % 18 = 12
18 % 12 = 6
12 % 6 = 0
6 % 0 =

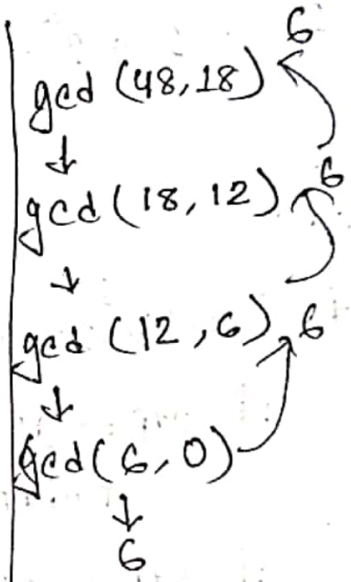
gcd (48, 18)

Step 1: 48 / 18 = 2 reminder 12

Step 2: 18 / 12 = 1 reminder 6

Step 3: 12 / 6 = 2 reminder 0

Step 4: 6 / 0 = ! → Ans = 6



algo:

gcd (a, 0) = a

gcd (a, b) = gcd (b, a mod b)

Code:

```
def gcd (a, b):
    if b == 0:
        return a
    else:
        return gcd (b, a % b)

print (gcd (48, 18))
```

2 | 1 | 0
1

Convert decimal number to binary:

steps: 13 to binary:

Div	Quo	Rem
13/2	6	1
6/2	3	0
3/2	1	1
1/2	0	1
0/2 → return 1		

binary: ~~1011~~ 1101

No, how we will get 1 to 1101 ?

1 0
1 = 101 * 10 + 0 = 1010
0 = 10 * 10 + 1 = 101
1 = 1 * 10 + 0 = 10

algo: $f(n) = n \bmod 2 + 10 * f(n/2)$

Code:

```
def dec2bin(n):
    if n==0:
        return 1
    else:
        return n%2 + 10 * dec2bin(int(n/2))

print(dec2bin(10)) ⟶ 1010
```

10 to binary:

Div	Quotient	Rem
10/2	5	0
5/2	2	1
2/2	1	0
1/2	0	1
0/2	!	→ 0

Binary: 1010

dec2bin(10): → 1010
↓
0 + 10 * dec2bin(5)
↓ 101
1 + 10 * dec2bin(2)
↓ 10
0 + 10 * dec2bin(1)
↓ 1
1 + 10 * dec2bin(0)
.
1