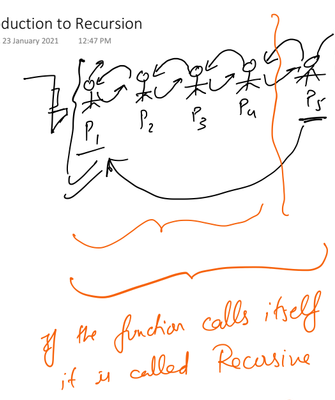
Introduction to Recursion

Saturday, 23 January 2021 12:47 PM

1)


P1
P2
P3
P4
P5

? what is the cost of the movie ticket

f(P5) → f(P4)

If the function calls itself again then it is called Recursive function.

$5! = 5 \times \underbrace{4 \times 3 \times 2 \times 1}$

1! => 1
0! =
-neg => 0

$f(5) = 5 \times f(4)$

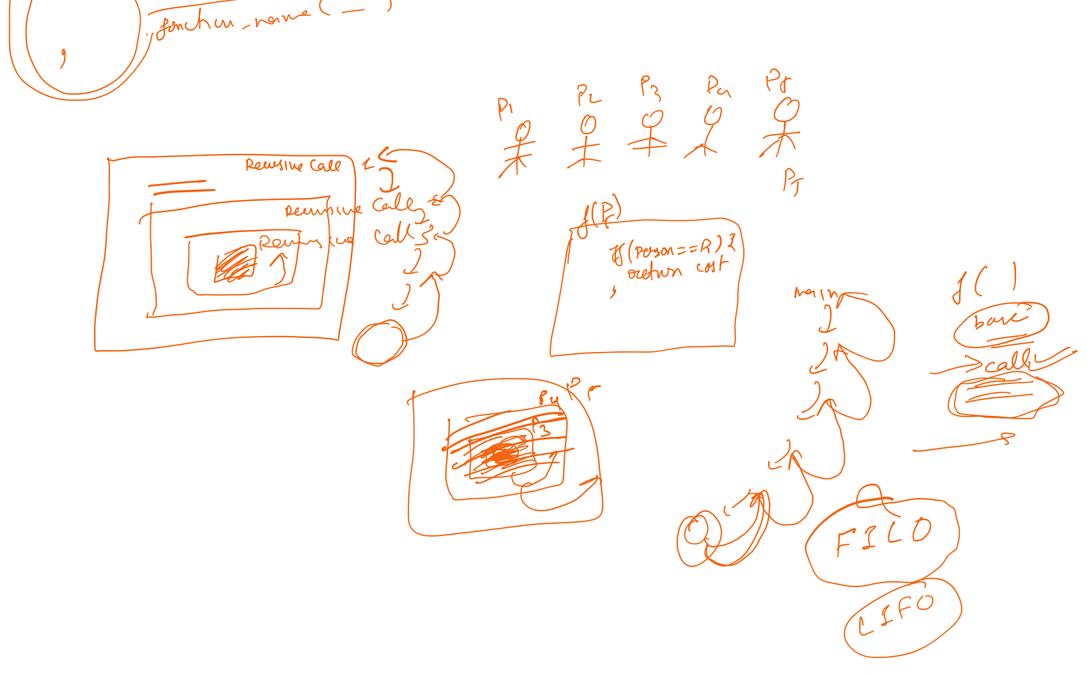
```
f(int n) {
    g( )
}
g( ) {
}
```

1→2→3→11→12→13
x ← 5 ← 4

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

⇒ 1 → 2 → 3
2 ← 1 ← 4
3
(4) → 1 → 2 → 3

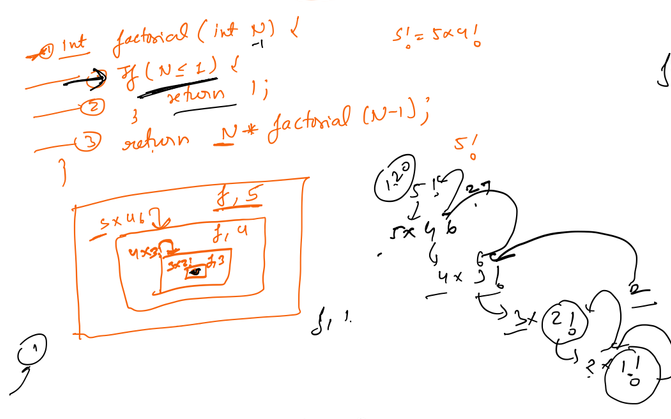
```
return type function_name ( params ) {
    if (condition) {
        return ___
    }
    function_name ( _ )
}
```


P1 P2 P3 P4 P5
P6
Recursive Call
Recursive
Call
Call
f(P)
if (Person==P) {
return cost
f( )
Base
call
FIFO
LIFO

```
int factorial (int N) {
    if (N<=1) {
        return 1;
    }
    return N * factorial (N-1);
}
```

5! = 5×4!

factorial (0)

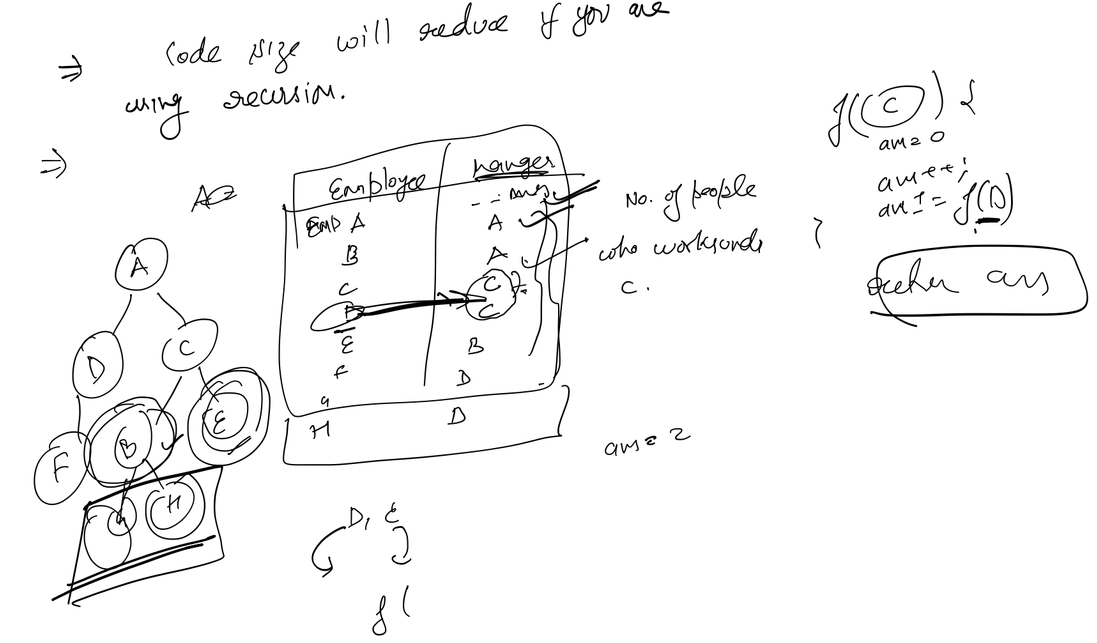

5×4!
f, 5
4×3!
f, 4
5!
5×4
4×3
3×2
2
1
1

Iteration v/s Recursion

1)
```
for ( i= ; ; ) {
}
```

⇒ code size will reduce if you are using recursion.

⇒


A
B
C
D
E
F
G
H
Employee
Manager
No. of people who works under C.
ans=2

```
f(C) {
    ans=0
    ans++;
    ans+= f(D)
    return ans
}
```

```
calculate_employee ( manager ) {
    if (.mang (emp) == manager) {
        ans++;
        ans+= calculate (Emp)
    }
    return ans
}
calculate_employ ( C )
```

a, b → a^b

$a^b = \underbrace{a * a * a \cdots}_{b \text{ times}}$

$a * (\underbrace{a*a \cdots}_{b-1 \text{ times}})$

pow(a, b) = a * pow (a, b-1)

b==0 → return 1;

```
int power (a, b) {
    if (b==0)
        return 1;
    return a* power(a, b-1);
}
```

2^4 → 2×4

Print (power (2, 4)) 16 → main

return 2* power (2, 3)

return 2* power (2, 2)

return 2* power(2, 1)

return 2* power(2, 0)

return 1

Q→ Given a number calculate it's number of digits. (-, 0, +)

⇒ f(-11123) → 5
f(23) → 2
f(0) → 1

f(-11123) → 1+ f(-1112)
1+ f(-111)
1+ f(-11)
1+ f(-1)
1

-9, -8, ... 0, 1, 2 ... 9

-9 ... 9

```
int f (int N) {
    if (-9<= N <= 9)
        return 1;
    return 1 + f(N/10);
}
```

N%10

f(2341) → 1 + f(234)
4 + f(23)
3 + f(2)
2

f(N)
4 × f(4)
4 × f(3)

```
f(N)
if (N==0)
    return 1
return N*f(N-1)
```

O(Total no. of function calls × Operation in one function)

⇒ O(N×1)

1 f N → N-1 → N-2

Q→ a^b ⇒ f(a, b)
a× f(a, b-1)
a× f(a, b-2)

O(b×1)

a^1000000 ⇒ O(1000000)
a×a× ... 7 times

a^b ⇒ a×a×a ... b times
(a×a ...) (a×a×a ...) × a (if b is odd)
b/2 times b/2 times
f(a, b/2) × f(a, b/2) × a (if b is odd)

b&1

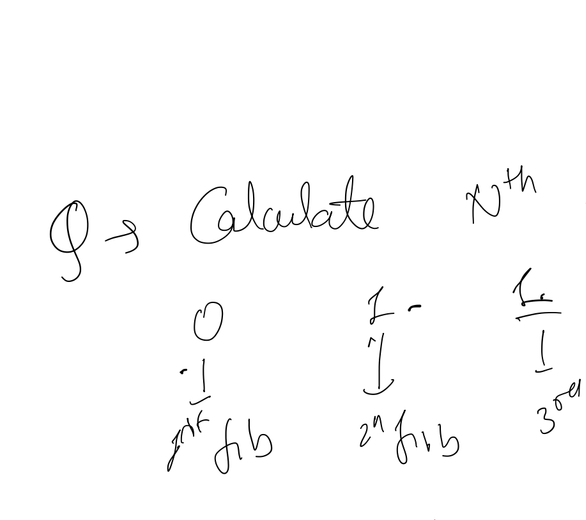
a^7 → a×a×a ... 7 times
(a× ... 3 times) (a× 3 times) × a
b/2 b/2

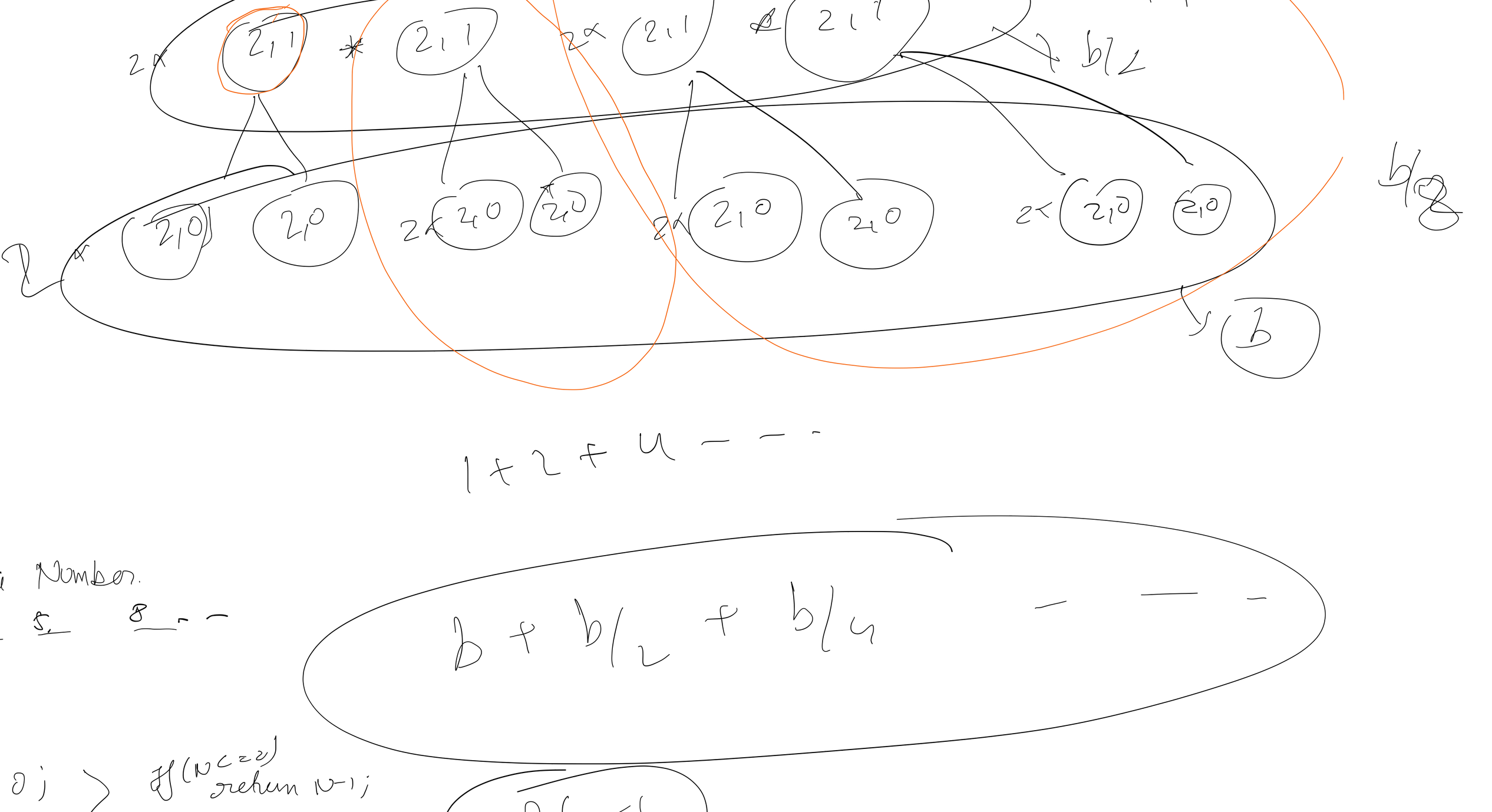
a^6 → a×a ... 6 times
(a× ... 3 times) × (a×a× ... 3 times)

$a^{b/2} \times a^{b/2} \Rightarrow a^{b/2+b/2} = a^b$

```
int f(a, b) {
    if (b==0) return 1;
    int temp= f(a, b/2)
    if (b&1)
        return temp * temp * a;
    else return temp * temp;
}
```

A O(log (a))
B O(log (b))
C O(b)
D O(log (a+b))


2, 6
2, 3
t×t×a
t=
2, 1
b
b/2
b/4
log(b)


2, 6
1
b
b/2
b/4
2, 3
2, 3
2, 1
2, 1
2, 1
2, 1
2, 0
b/4
b/2
b/8
b
log(b)

1 + 2 + 4 ...

b + b/2 + b/4 ...

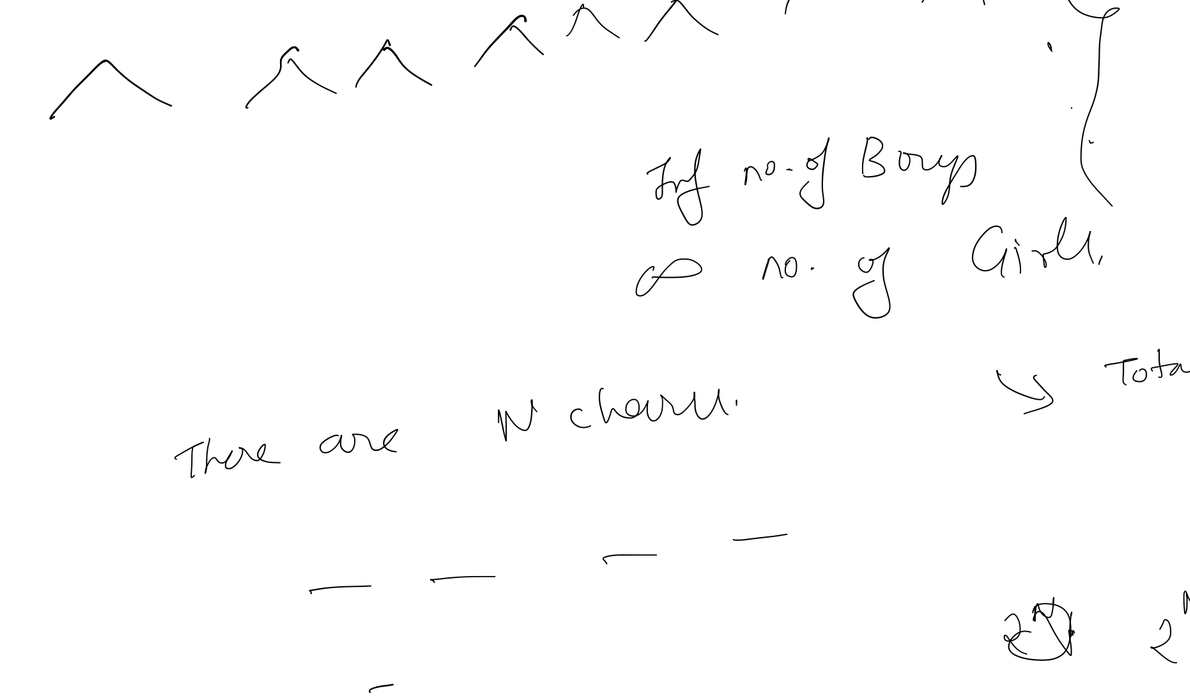
2b - 1

Q→ Calculate Nth Fibonacci Number.

0 1 1 2 3 5 8 ...
1st fib 2nd fib 3rd

```
f(N) {
    if (N==1) return 0;
    if (N==2) return 1;
    return f(N-1) + f(N-2);
}
```

if (N<=2) return N-1;


N
N-1
N-2
N-2
N-3
N-3
N-4

→ N

O(N)

$\underbrace{1+2+4+8+ \cdots + 2^N}_{N \text{ terms}}$ $2^{N+1}-1$

$2^N - 1$

$O(2^N)$

$\frac{a r^N - 1}{r-1}$ $\frac{1 \times 2^N - 1}{2-1}$

$O(2^N)$ $= 2^N - 1$

Inf no. of Boys
∞ no. of Girls

There are N chairs. → Total no. of diff arrangement

_ _ _ _

2^N

No two girls can sit together

$2^{N/2} \rightarrow \frac{2^N}{2}$

(3)

BBB _
BBG _
BGB _
GBB _
BGG _ ✗
GBG _
GGB _ ✗
GGG _ ✗

(5)

N=3
ans=5

G G _ _ _ 2^{N-2} ✗
GB
BB
BG

enigma27

f(N, B) → f(N-1, B) + f(N-1, G)

f(N-2, B)

f(N, B) → f(N-1, ~~B~~) + f(N-1, G) f(N-2, B)

f(N, G) → f(N-1, B)

f(N-1, G)

```
int f( int chair, int last ) {   // 0/1
    if (chair == 0) return 1;
    if (last == 0) {
        return f(chair-1, 0) + f(chair-1, 1)
    }
    else return f(chair-1, 0)
```

→ Fibonacci Number.

f(N, 0)

f(N-1)

f(N-2)

f(N) = f(N-1) + f(N-2)

768-B codeforces → Code for 1

GCD of two Number

1385-D codeforces → a-Good String

Recursive digit Sum → Hackerrank