Recursion - 2

↳ More problems

↳ T.C/S.C analysis

Q.1 Given a number. Write a recursive program to calculate the sum of digits of that no.

Assumption: sumDigit(N) ⇒ Returns the sum of digits of N.

Main logic: N : 4 2 6 8 9

N%10 → 9

N/10 → 4268

N%10 → last digit

N/10 → Remove last digit

return `(N%10) + sumDigit(N/10);`

Base Condition: if (N<10) return N;

Q.2 Implement pow function :

Given a, n

return a^n [without any library function]

$a = 3, \quad n = 4 \Rightarrow a^n \Rightarrow 3^4 = 81$

$$a^n = a * a^{n-1}$$

Base condition :

$a^0 = 1$

N = 0

ans = 1

$3^0 = 1$

$17^0 = 1$

```
int pow(a, n){
  if(n==0) return 1;

  return  a * pow(a,n-1);
}
```

$a^5 \Rightarrow a * a^4$

$\hookrightarrow a * a^3$

$\hookrightarrow a * a^2$

$\hookrightarrow a * a^1$

$\hookrightarrow a * a$

$a^n \Rightarrow N$ multiplications $\Rightarrow$ T.C $\Rightarrow O(N)$

$a^{10} = a * a^9$

$a^{10} = a^5 * a^5$

$a^{14} = a^7 * a^7$

$a^{15} = a^7 * a^7 * a$

$a^{20} = a^{10} * a^{10}$

$a^{21} = a^{10} * a^{10} * a$

$$a^N = \begin{cases} a^{N/2} * a^{N/2} & \text{if } N\%2==0 \\ a^{N/2} * a^{N/2} * a & \text{if } N\%2==1 \end{cases}$$

$a^{64} \Rightarrow a \ * \ a$

$a^{16} * a^{16}$

$a^{8} * a^{8}$

$a^{4} * a^{4}$

$a^{2} * a^{2}$

$a^{1} * a^{1}$

$a * a^{0} * a^{0}$

$a^N \Rightarrow (\log_2 N)$ no. of multiplications.

$\Rightarrow$ **T.C. = $O(\log N)$**

Google.

```
int pow(a, n, d){
  if(n==0) return 1;

  int halfpow = pow(a, n/2, d)
  int halfans = (halfpow%d * halfpow%d) %d

  if(N%2==0){
      return halfans;
```

Typecast as (long)

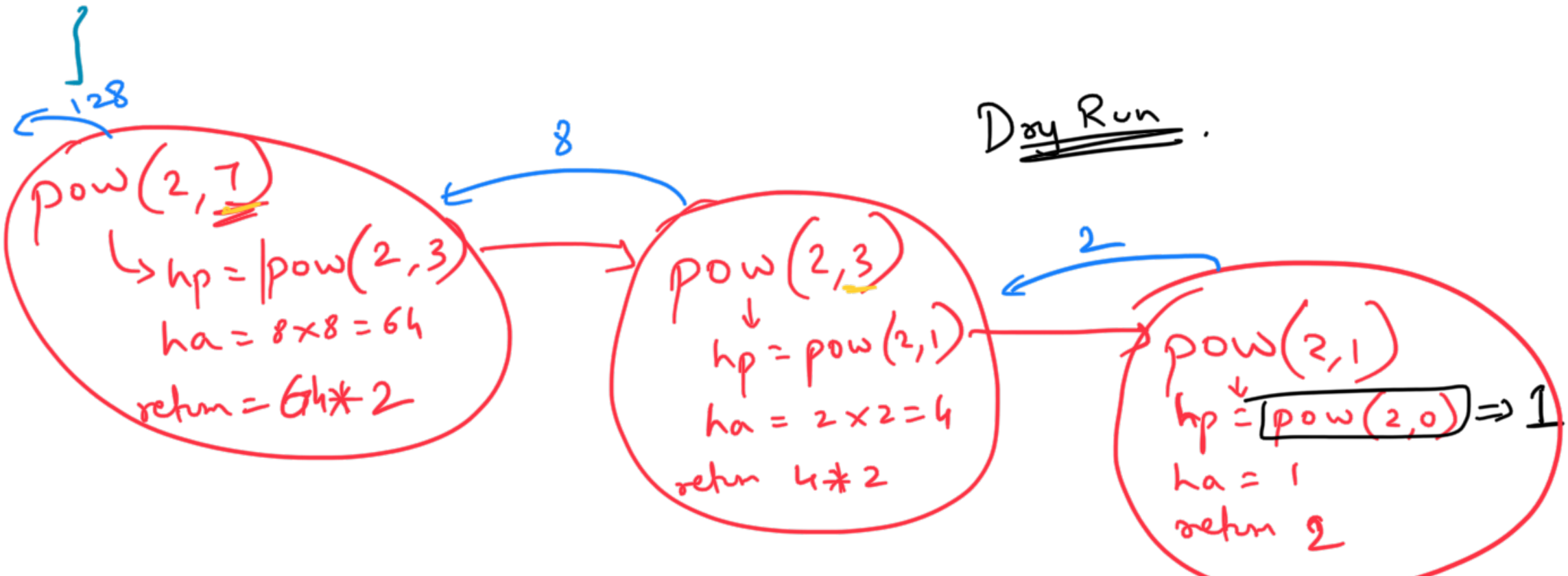
Constraint
$a \le 10^9$
$n \le 10^6$
ans%d

$d \le 10^9$

$a^N \% d$

1

```
}
else return (halfans * a) % d
```



$d = 10^9 + 1$

$\underline{halfpow} = 10^8 \% d$

$halfans = (halfpow \% d * halfpow \% d) \% d$

$= (\underline{10^8 \% (10^9+1)} * \underline{10^8 \% (10^9+1)}) \% (10^9+1)$

$= \left(10^8 \right) * \left(10^8 \right)$

long

T.C $\Rightarrow O(\log N)$

Time Complexity

$sum(N) = N + sum(N-1)$

$N-1 + sum(N-2)$

$N-2 + sum(N-3)$

If $T(N)$ gives you T.C. of $sum(N)$

$T(N-1)$ will give you T.C. of $sum(N-1)$

$sum(N) = N + sum(N-1)$

$T(N) = 1 + T(N-1)$

$N-1 + sum(N-2)$

$1 + T(N-2)$

$$T(N) = 1 + (1 + \underbrace{T(N-2)}_{\rightarrow\ 1 + T(N-3)})$$

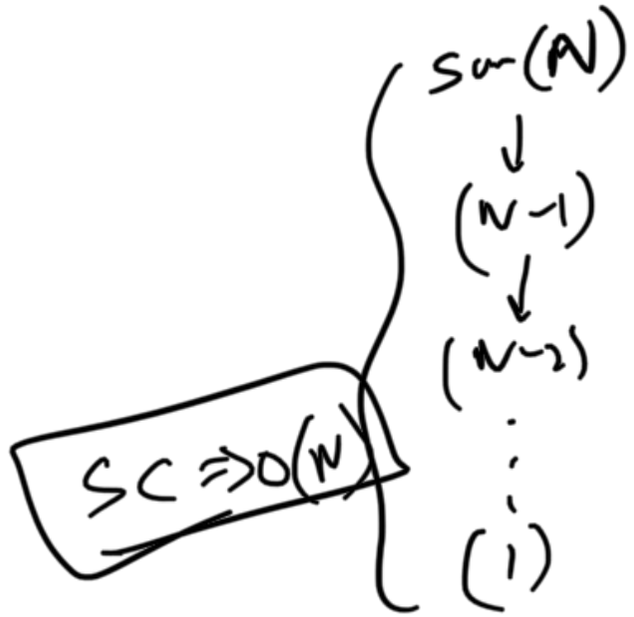
$$T(N) = 2 + T(N-2)$$

$$T(N) = 2 + (1 + T(N-3))$$

$$= 3 + T(N-3)$$

$$T(N) = 4 + T(N-4)$$

$$\vdots$$



After K steps,

$$T(N) = K + T(N-K)$$

Last step if $K = N$

$$T(N) = N + T(N-N)$$

$$= N + T(0)$$

$= N + 1$

$$T(N) = O(N)$$

Q

$$T(N) = 2\,T(N-1) + 1 \qquad \& \quad T(0) = 1$$

$$T(N-1) = 2\,T(N-2) + 1$$

$$T(N) = 2\left(2T(N-2) + 1\right) + 1$$

$$= 4\,T(N-2) + 3 \Rightarrow 2^2\,T(N-2) + (2^2 - 1)$$

$$T(N-2) = 2\,T(N-3) + 1$$

$$T(N) = 4\left(2\,T(N-3) + 1\right) + 3$$

$$= 8\,T(N-3) + 7 \Rightarrow 2^3\,T(N-3) + (2^3 - 1)$$

$$fib(N) = fib(N-1) + fib(N-2)$$

$$T(N-1) + 1 + T(N-1)$$

$$T(N) = 8\left(2T(N-4)+1\right)+7$$

$$= 16\,T(N-4)+15 \Rightarrow 2^4\,T(N-4)+\left(2^4-1\right)$$

Substitution method using Recurrence relation

After K steps,

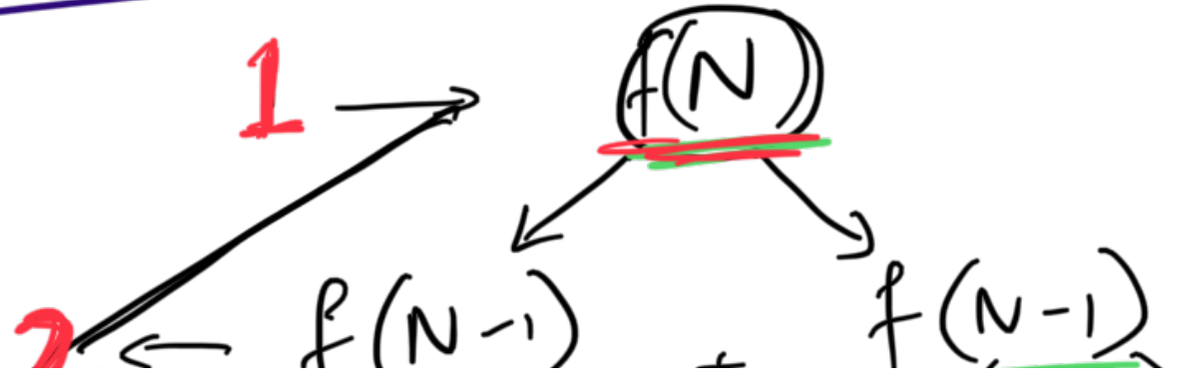
$$T(N) = 2^K\,T(N-K)+\left(2^K-1\right)$$

For last step, $N-K=0 \Rightarrow \boxed{K=N}$

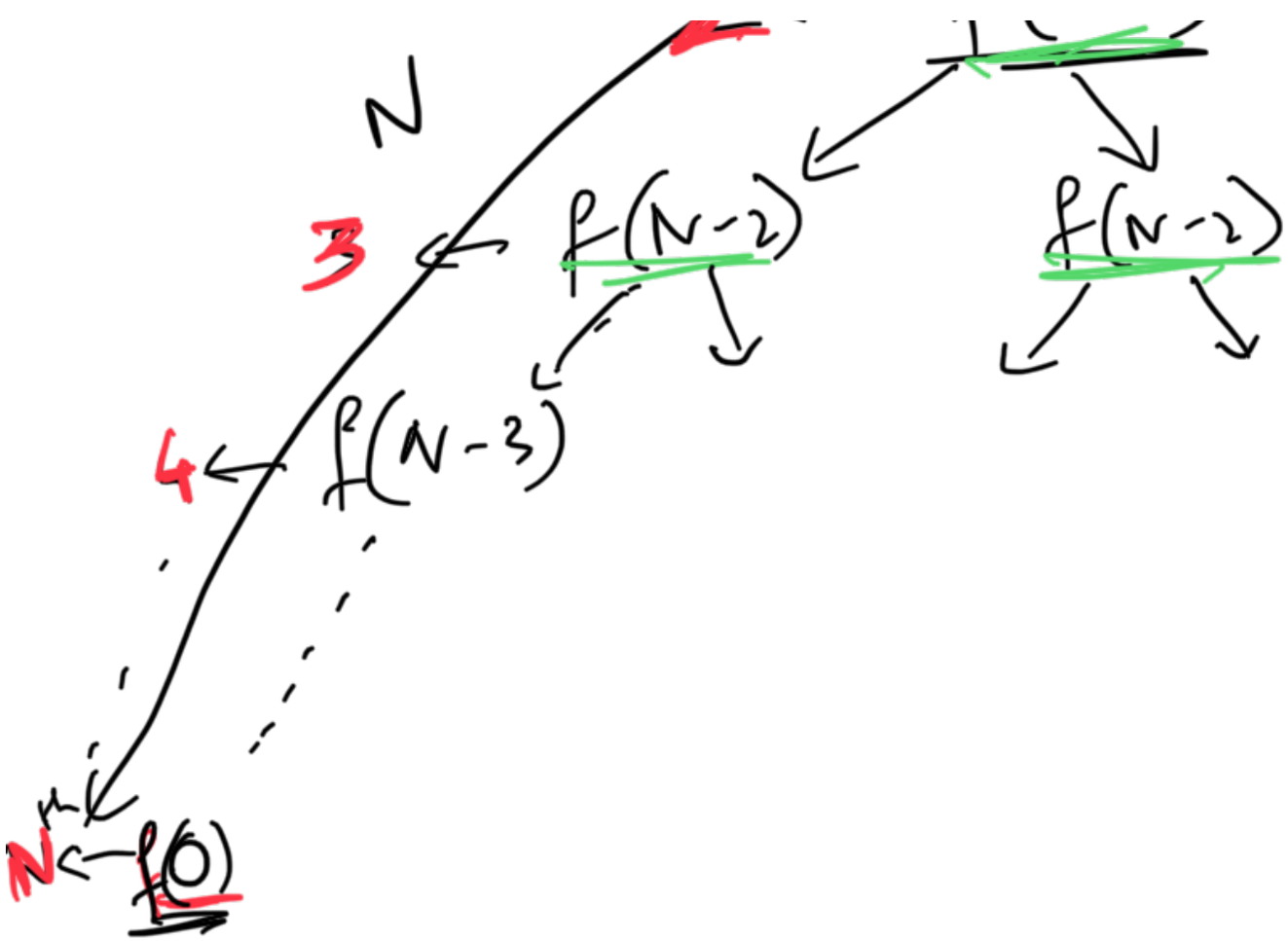
$$T(N) = 2^N\,T(0)+\left(2^N-1\right)$$

$$= 2^N+2^N-1 \Rightarrow 2\left(2^N\right)-1$$

$$\boxed{T(N) = O(2^N)}$$



Recursion Tree

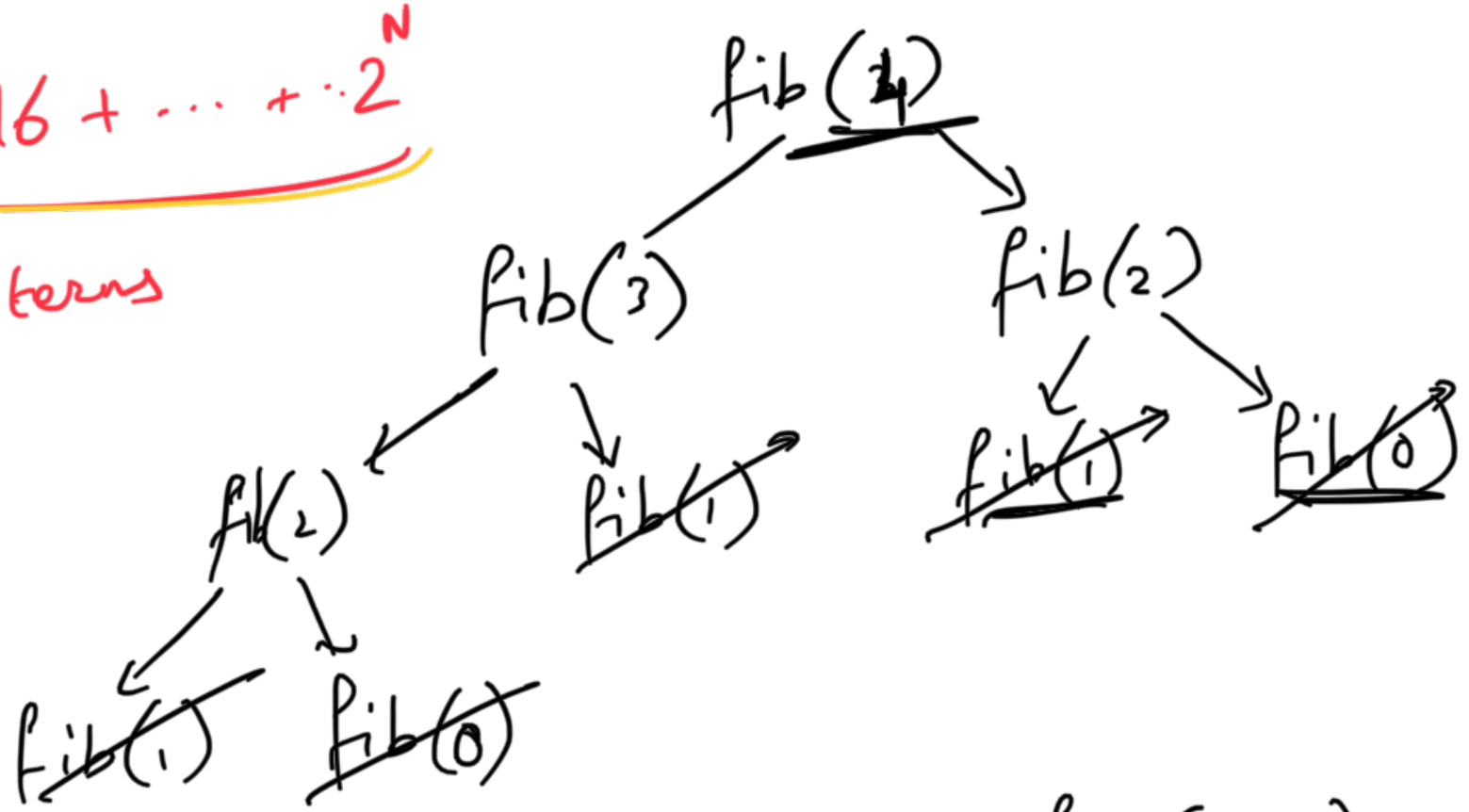


$$f(N) = f(N-1) + f(N-2)$$

$$T(N) = T(N-1) + T(N$$

$$1 + 2 + 4 + 8 + 16 + \cdots + 2^{N}$$

G.P with N terms



fib(1000)

Break → 10:25

for loop,

1, 1, 2, 3, 5, 8 ...

a b c

$a = 1$

$b = 1$

$c = a + b$

$a = b$

$b = c$

Recursion

[ ]

log N

N^K

Q: $T(N) = T(N/2) + 1$

$= T(N/4) + 2$

$= T(N/8) + 3$

$\vdots$

$T(N) = T(N/2^K) + K$

Pow

Binary Search.

$$\frac{N}{2^K} = 1 \Rightarrow 2^K = N \Rightarrow K = \log_2 N$$

$$T(N) = T(1) + \log_2 N$$

$$= 1 + \log N$$

$$\boxed{T(N) = O(\log_2 N)}$$

Quiz :

$$T(N) = 2\,T(N/2) + 1 \Rightarrow 2^1\,T\left(\frac{N}{2^1}\right) + \left(2^1 - 1\right)$$

$T(N/2) \rightarrow 2\,T(N/4) + 1$

& $T(1) = 1$.

$$= 2\left(2\,T(N/4) + 1\right) + 1$$

$$T(N) = 4\,T(N/4) + 3 \Longrightarrow 2^2\,T\left(\frac{N}{2^2}\right) + \left(2^2 - 1\right)$$

$T(N/4) \rightarrow 2\,T(N/8) + 1$

$$= 4\left(2\,T(N/8) + 1\right) + 3$$

$$T(N) = 8T(N/8) + 7 \Rightarrow 2^3 T\left(N/2^3\right) + \left(2^3 - 1\right)$$

$$\vdots$$

After K steps,

$$\boxed{T(N) = 2^K T\left(N/2^K\right) + \left(2^K - 1\right)}$$

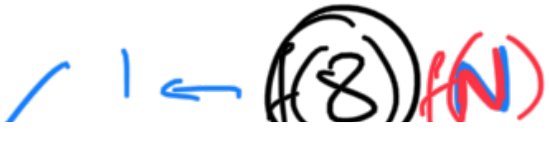
At the last step, $\frac{N}{2^K} = 1 \Rightarrow 2^K = N \Rightarrow \boxed{K = \log_2 N}$

$$T(N) = 2^{\log_2 N} T\left(\frac{N}{2^{\log_2 N}}\right) + \left(2^{\log_2 N} - 1\right)$$

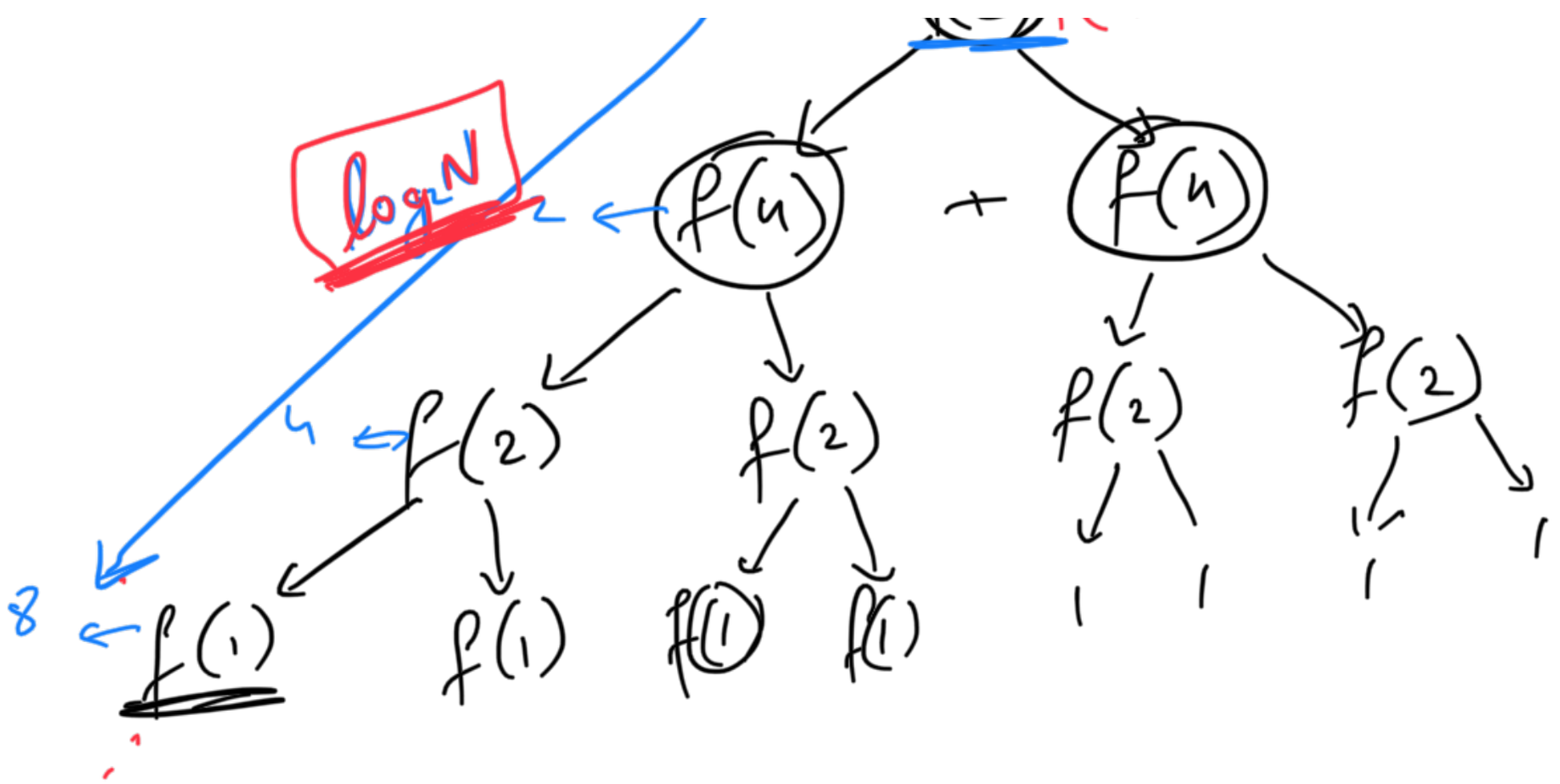
$$= N * T\left(\frac{N}{N}\right) + (N - 1)$$

$$= 2N - 1$$

$$\boxed{T(N) = O(N)}$$



Recurrence tree



$$1 + 2 + 4 + 8 + \cdots + 2^{\log N}$$

GP $\Rightarrow r = 2$.

$$\frac{a(r^N - 1)}{r - 1} = \frac{1(2^{\log N} - 1)}{2 - 1}$$

$$= N - 1$$

T.C $\Rightarrow O(N)$

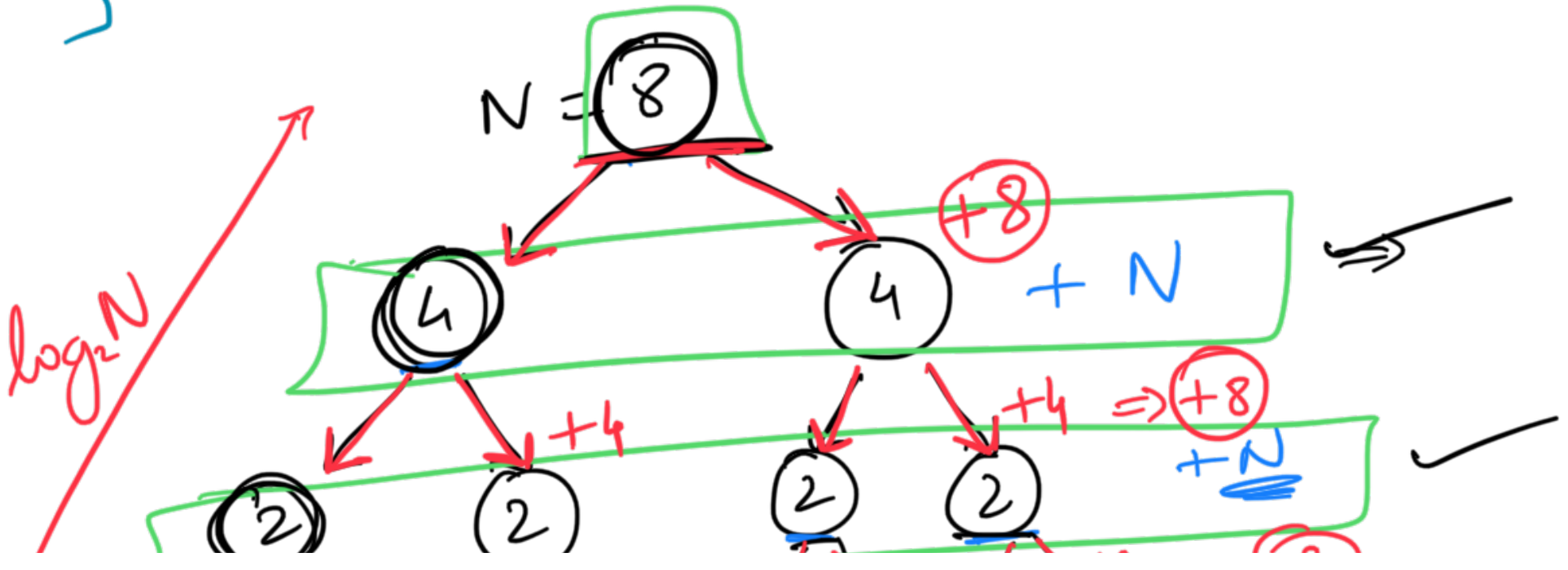
Quiz: $T(N) = 2\,T(N/2) + (N)$

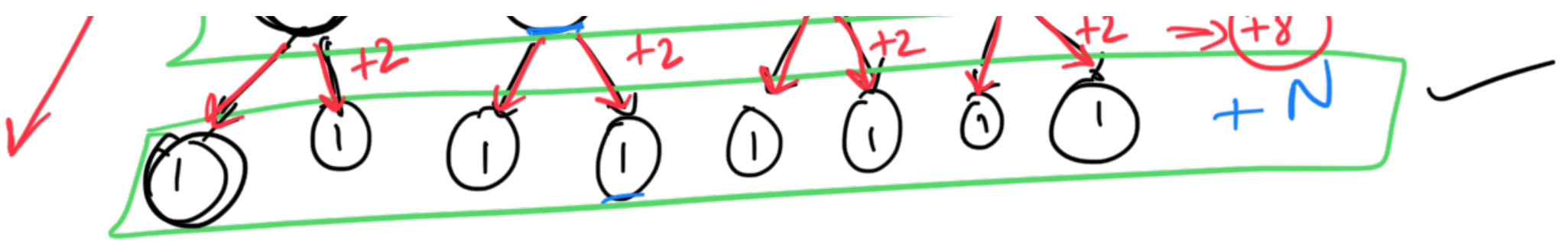
H.W
Do substitution & find the ans.

```
f(N){ → T(N)

    f(N/2) .. → T(N/2)
    f(N/2)    → T(N/2)
    for(i=0→N) → N operations
    ...
}
```

Merge Sort
quick Sort





At each level

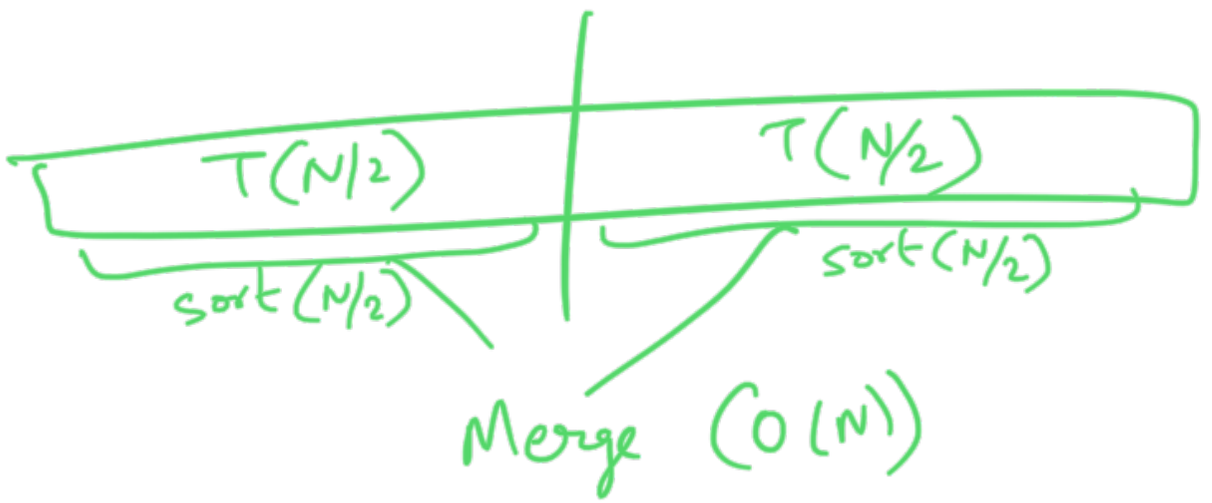
$$\left(1+2+4+8+16+\cdots 2^{\log N}\right) + \underbrace{\left(N+N+N+\cdots\right)}_{\log N\ \text{terms}}$$

$$\Downarrow$$

$$O(N) \quad + \quad O(N \log N)$$

$$T.C \Rightarrow O(N \log N)$$

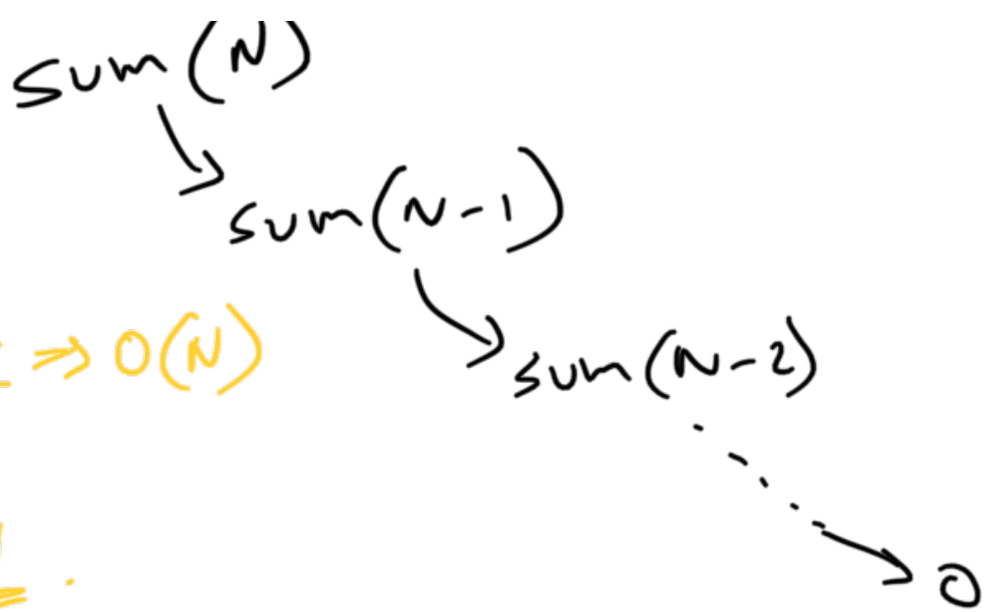
Merge sort



Complexity

$1+2+3\cdots+N$

Space

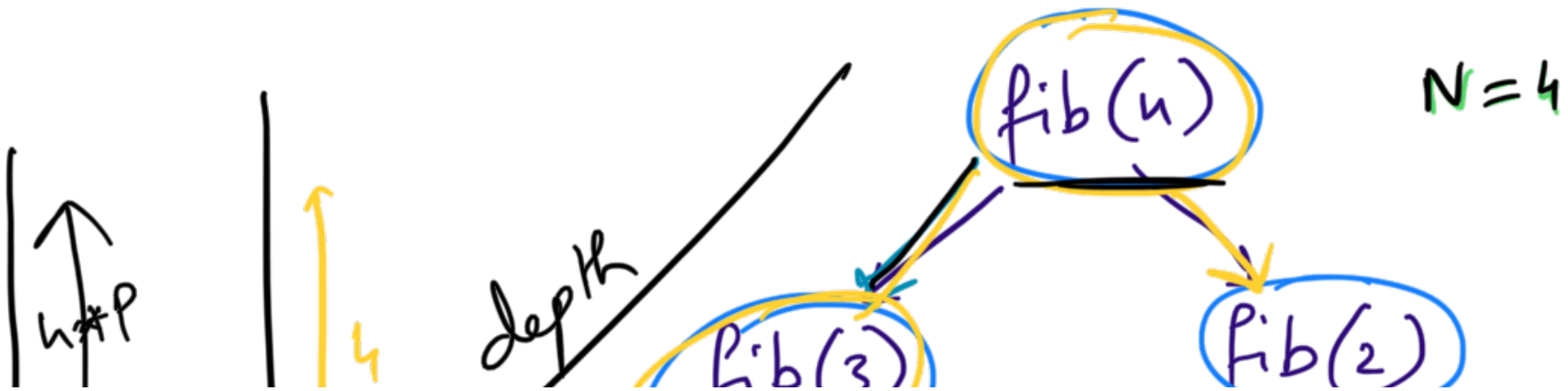


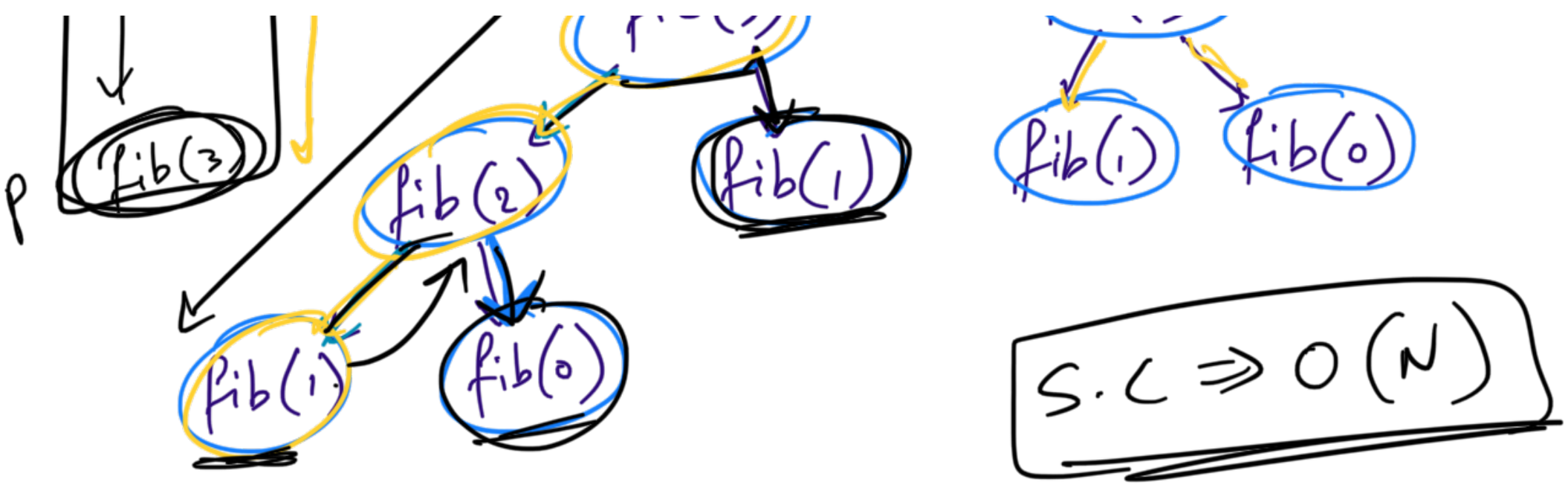
size of stack = N.

T.C ⇒ O(N)

Space complexity ⇒ Max. size of stack at any point of time.

$$fib(N) = fib(N-1) + fib(N-2)$$




p
fib(3)
fib(2)
fib(1)
fib(1)
fib(0)
fib(1)
fib(0)
S.C ⇒ O(N)