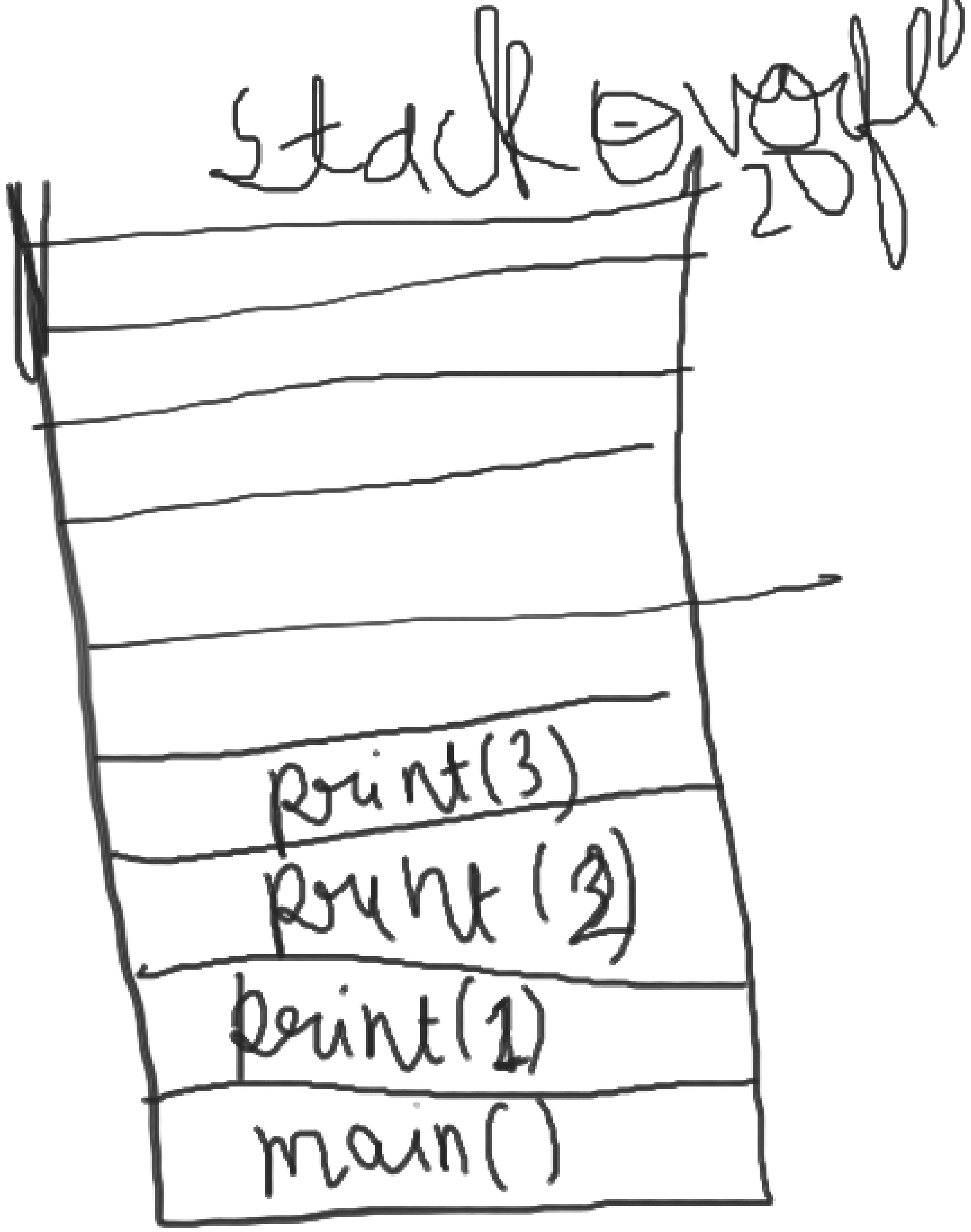
Understanding Recusrion in Depth

Recursion

Main()

Q. print numbers from 1 to 5



```
static void print(int n){
    sout(n);
    print(n+1);
}
```

* 1.while the function has not finished executing the function will remain in the stack

2.when a function finishes executing it is removed from the stack and flow of program is concluded.

VISUALIZING Recursion

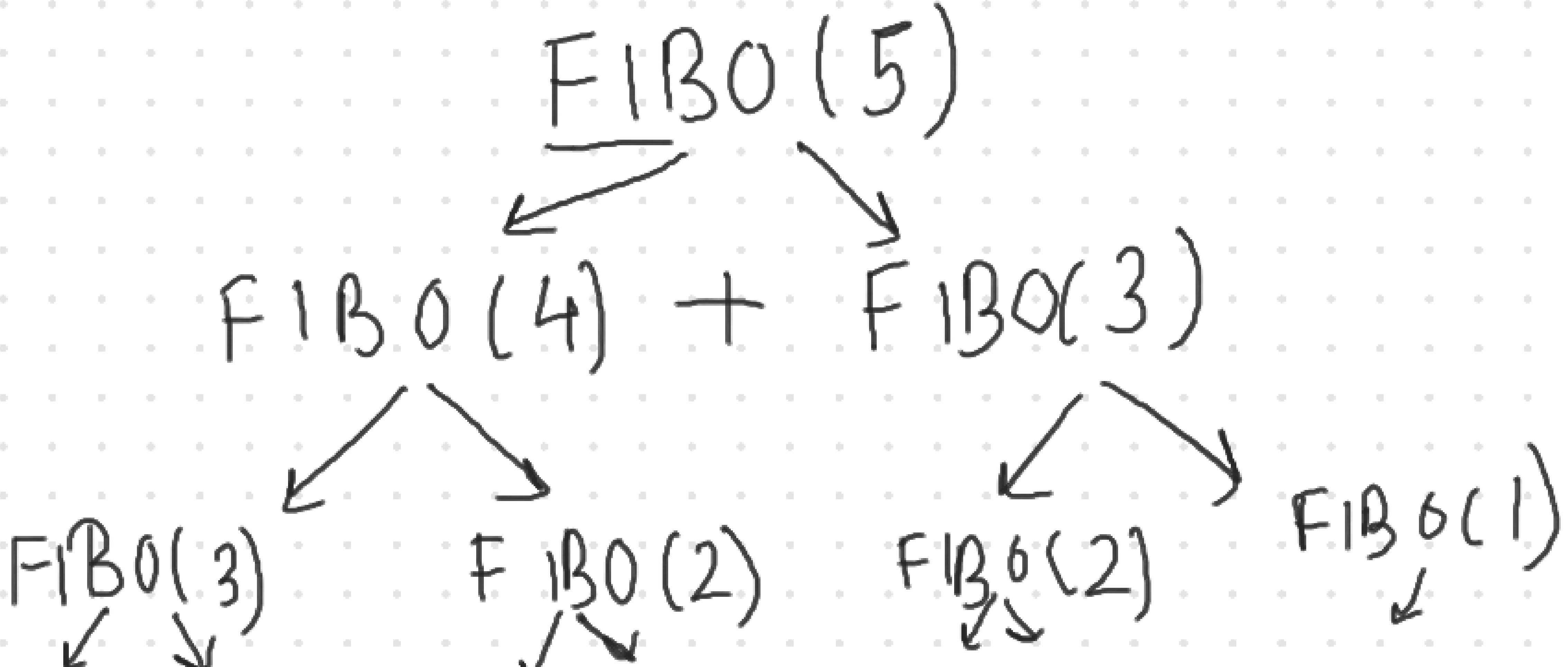
Q. Find the nth fibonacci number

Fibo. No.'s: 0, 1, 1, 2, 3, 5, 8. . . . .

0th 1st 2nd 3rd 4th

{2nd Fibo No = 0th + 1st}

5th



Recursion tree

~~FIBO(0)~~ 2

~~FIBO(2)~~

FIBO(2)

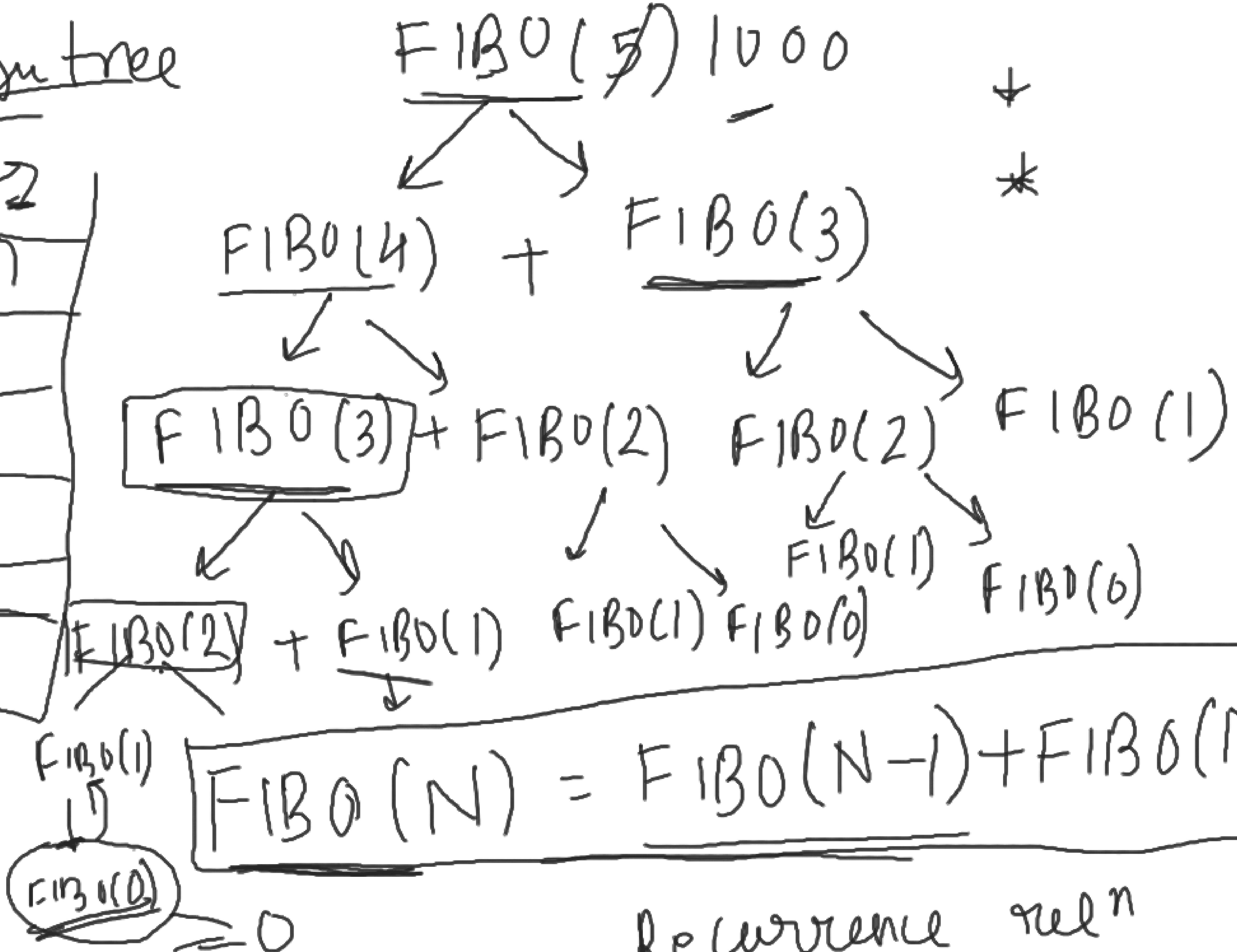
FIBO(3)

FIBO(4)

FIBO(5)

main()

Stack



$$\boxed{FIBO(N) = FIBO(N-1) + FIBO(N-2)}$$

Recurrence relⁿ

Why Recursion?

1. It helps us to breakdown complex problems into smaller problems.
2. We can solve recursion problem using Iteration.
3. Space Complexity : Not Constant (∵ Repetetive function calls)

How to approach a problem?

1. Identify if you can break down the problem into simpler problems.
2. Write the recurrence relation
3. Make the recursion tree
4. About the tree:
4.1: See flow of function , stack implementation .
4.2: Follow left tree calls, and then right tree calls
4.3: Use debugger to understand the working of the program.

WORST CASE: 5, 4, 3, 2, 1

j

i = 0 :

4, 5, 3, 2, 1

4, 3, 5, 2, 1

4, 3, 2, 5, 1

4, 3, 2, 1, 5

N - 1

i = 1

4, 3, 2, 1, 5

3, 4, 2, 1, 5

3, 2, 4, 1, 5

3, 2, 1 | 4, 5

N - 2

i = 2 N-3

i = 3 N-4

i = 4 N-5

BEST CASE

~~1, 2, 3, 4, 5~~

i = 0

1, 2, 3, 4, 5

1, 2, 3, 4, 5

1, 2, 3, 4, 5

1, 2, 3, 4 | 5

$$T(N) = O(N-1 + N-2 + N-3 + N-4 + N-\underline{5})$$

$$= O(4N - (1+2+3+ \cdots N))$$

$$= O\left(4N - \left(\frac{N(N+1)}{2}\right)\right)$$

$$= O\left(4N - \left(\frac{N^2+N}{2}\right)\right)$$

$$= O\left(\frac{8\cancel{N} - N^2 - \cancel{N}}{\cancel{2}}\right)$$

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

Worst case

Ignored in T.C equations

o) Constants

o) Less Dominant terms

Q) arr = $\{\underbrace{\overset{0}{5}, \overset{1}{3}, \overset{2}{4}, \overset{3}{1}, \overset{4}{2}}\}$

We want to solve it using one single for loop.

VV Imp # Array elements are in range $\boxed{1 - N.}$

⇒ Cyclic Sort

Index: 0 1 2 3 4

After sorting = {1, 2, 3, 4, 5}

VV Imp $\boxed{\text{Index} = \text{Value} - 1}$

$\begin{array}{lccccc} & 0 & 1 & \boxed{2} & 3 & 4 \\ 1 & \{3, & 5, & 2, & 1, & 4\} \end{array}$ — N

swap

$\begin{array}{lccccc} & 0 & 1 & 2 & 3 & 4 \\ 2 & 2, & 5, & 3, & 1, & 4 \end{array}$

swap

$\begin{array}{lccccc} & 0 & 1 & 2 & 3 & 4 \\ 3 & 5, & 2, & 3, & 1, & 4 \end{array}$

swap

$\begin{array}{lccccc} & 0 & 1 & 2 & 3 & 4 \\ 4 & 4, & 2, & 3, & 1, & 5 \end{array}$

swap

$\begin{array}{lccccc} & 0 & 1 & 2 & 3 & 4 \\ \Rightarrow & \{\underline{1}, & \underline{2}, & \underline{3}, & \underline{4}, & \underline{5}\} \end{array}$

N-1

+

N

Index = Value - 1

Worst Case

$TC = O(N - 1 + N)$

$= O(2N - 1)$

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

Linear T.C

Case 1 $\left\{ \begin{array}{ccccc} 0 & 1 & \boxed{2} & 3 & 4 \\ 0, & 1, & \underline{3}, & 4, & \underline{\underline{5}} \end{array} \right\}$

$\boxed{2}$

Index = arr[i]

return index

Case 2 $\{\underline{0, 2, 1, 3}, 4\}$ $\boxed{5}$

$\underline{\text{arr.length}}$

if(arr[i] != arr[correct]

&& $\underline{\text{arr[i] < arr.length}}$)