Recursion - Basic Problem Solving

13 June 2022 19:58

Q Fibonacci

0 1 1 2 3 5 8 13 .....

0th 1th 2nd 3rd 4th 5th 6th 7th

n = 3

output -> 2

n = 4

output -> 3

find nth fibonacci no.

nth

Bigger Problem

(n-1)th

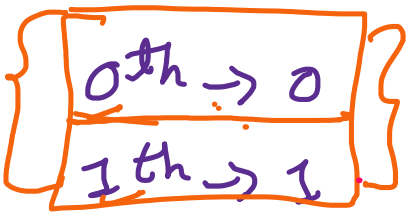
(n-2)th

Smaller Problem

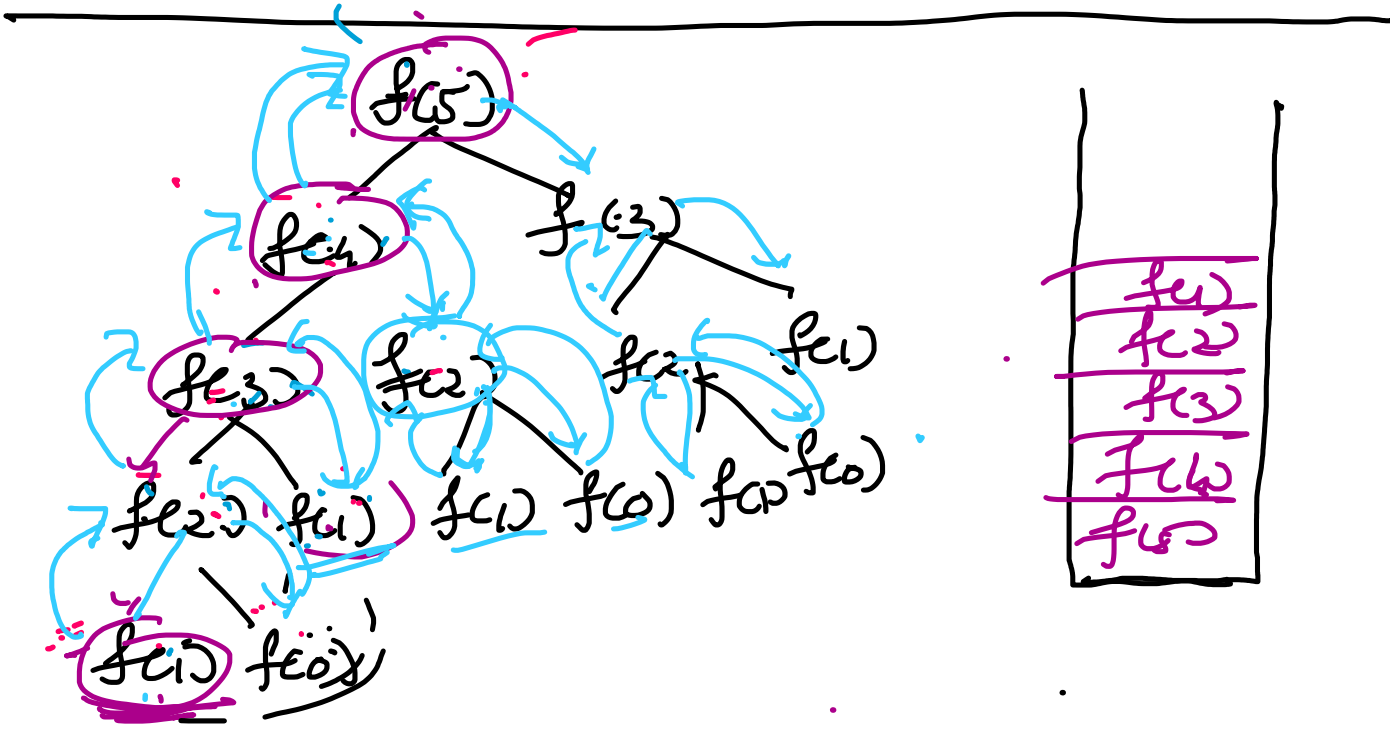
f(n) = f(n-1) + f(n-2)

Recursive Solution

1. Base case
2. Recursive call



Base case



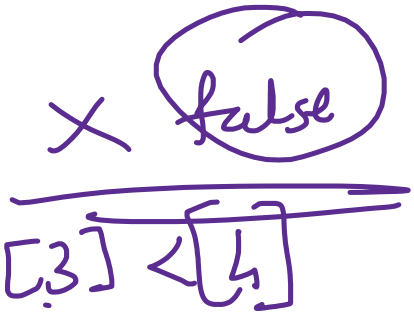
Check if array is sorted ascending?

I/P [5,4,3,2,1]
O/P false

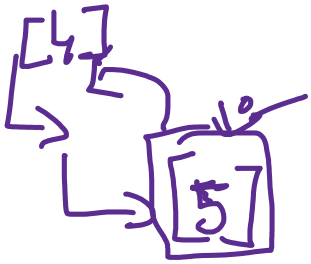
I/P [1,2,3,4,5]
O/P true

[1,2,3,4,5] } Bigger Problem
0 5 elements

Dipanshu (1)
↳ Priyanshu2
[2,3,4,5]
Ekta



[3,4,5]



f(arr, i) → Bigger Problem

f(arr, i+1) → Smaller Problem

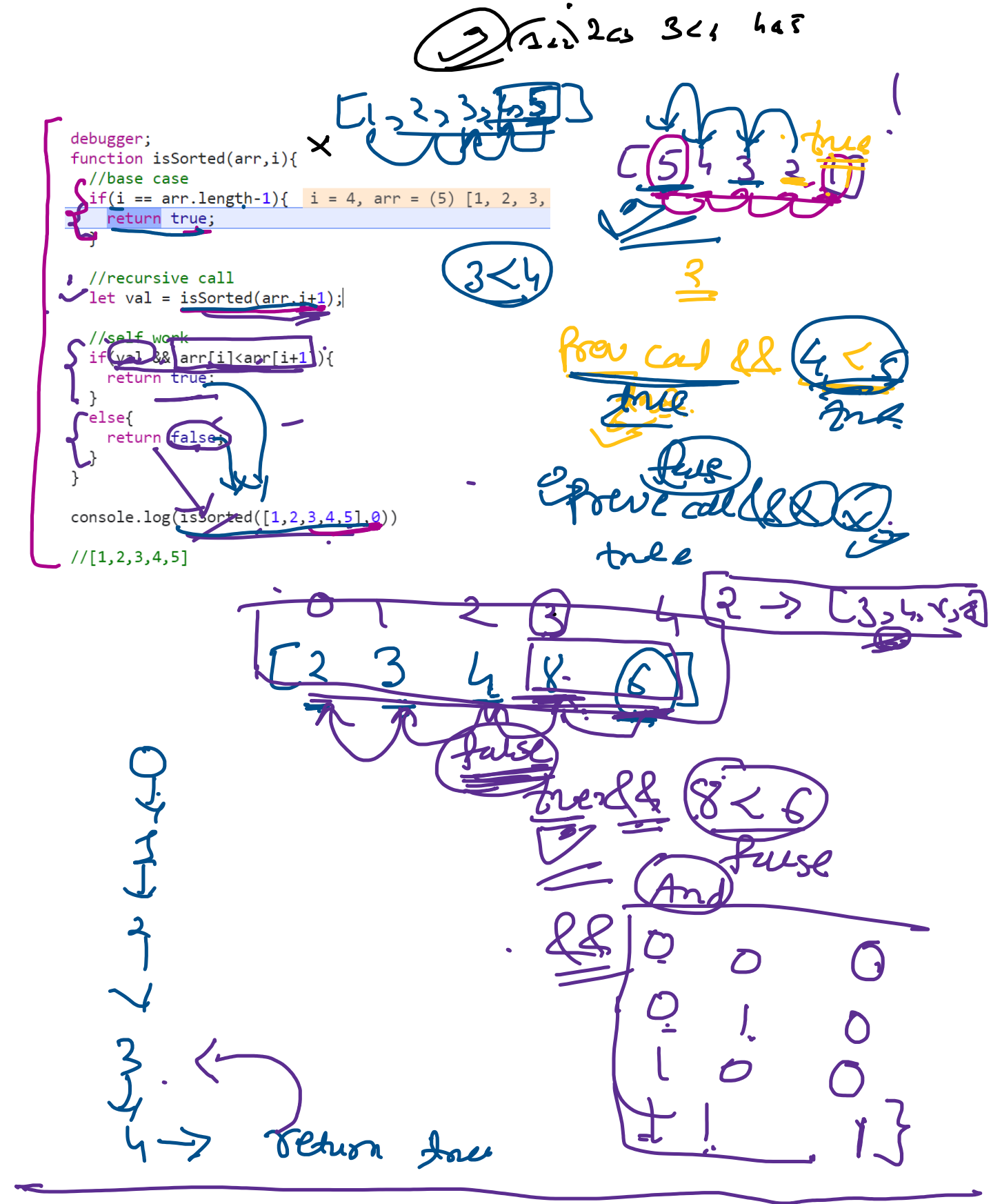
yes

no

no

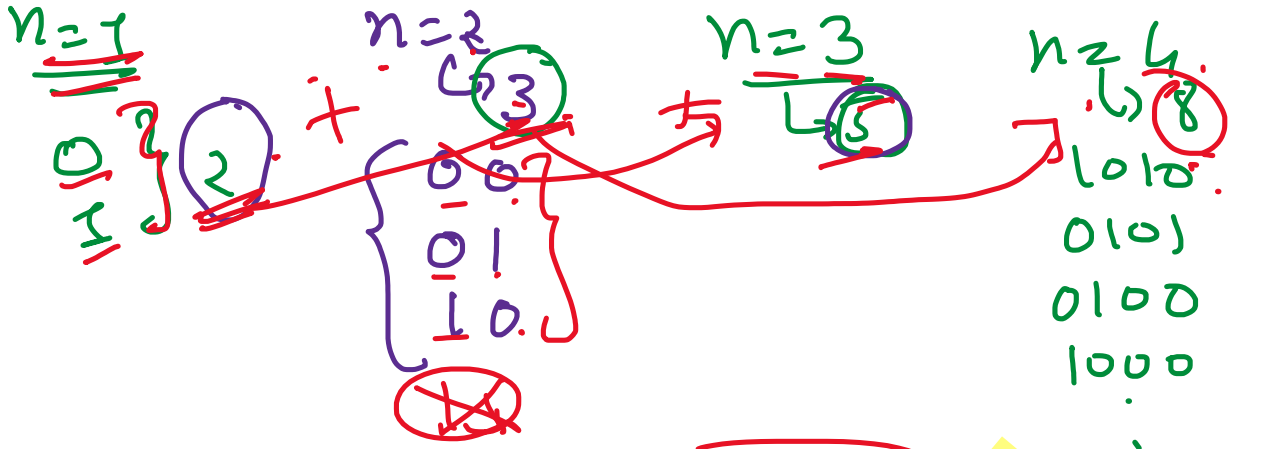
if(arr[i] < arr[i+1])

[-1]



e.g. I/P n=3

O/P: 5 [101, 100, 010, 001, 000]



Q Subsequence

arr = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10].

Subsequence: may not be contiguous but maintain relative order.

Elements in the subsequence appear in the same order as in original array. Only difference is that they might not be contiguous.

[4, 5, 6, 7] → Subarray / Subsequence ✓

[4, 6, 7] → Subarray / Subsequence

[1, 9, 10] → X

[5, 6, 7] → ✓

[1,3,2,3] → X Y

[1,4,7,3] → X X

Q Given an array, find all subsequences.

I/P [1,2,3,..] ⇒

O/P

[1]
[2]
[3]
[1,2]
[1,3]
[1,2,3].
[2,3]
[...]

For any array of length n
the total no. of subsequences ⇒ 2^n

[2,3] → [2] [3] [2,3] ... 4

$2^2 \Rightarrow 4$

Ekta
[1,2,3]
Subsequence
[Bigger Prob]
[1,2,3]

Deepanshu
[2,3]
[2,3]
[2] [3]

Base case
Aradhya
[3]
[[3]