GRAPH Interview Problems

SANKETID

↳ Agenda → Diverse problem solving experience of graphs

→ Problems relevant for SDE 1-2

Pre-requisite → Basic knowledge of graph

→ Basic knowledge of Recursion

→ few DS, queue, ll etc

Easy-Med

Q→ You have been given a grid of charachters. Also you have a word given

Check if the word exists in the grid, in the grid we can construct the word by taking any adjacent non diagonal cell. The same letter from the grid shouldn't be used twice.

$(m \times n) \le 2 \times 10^2$

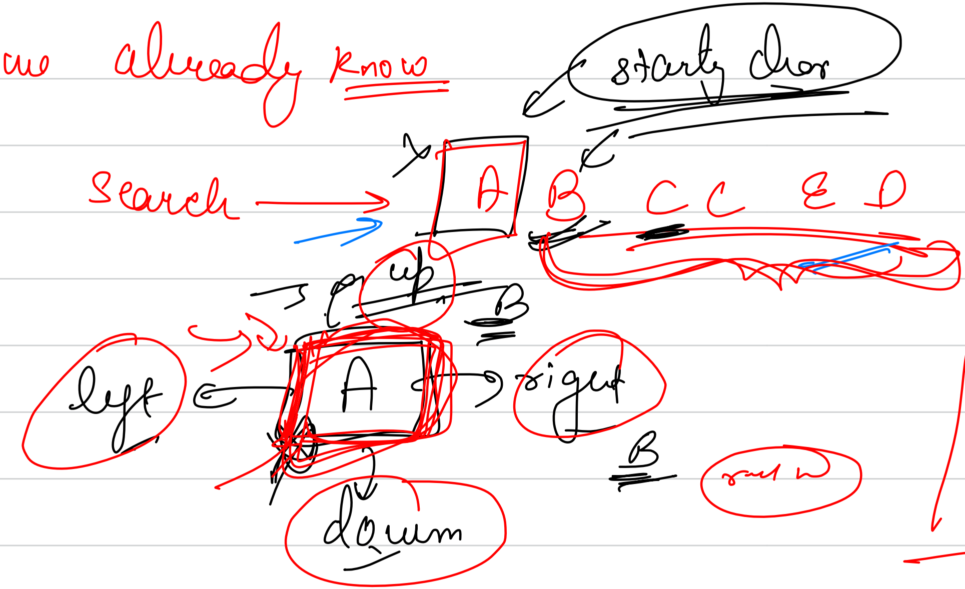
$word \le 10^3$

A	B	C	E
S	F	C	S
A	D	E	E

Search → ABCCED

True

What we already know



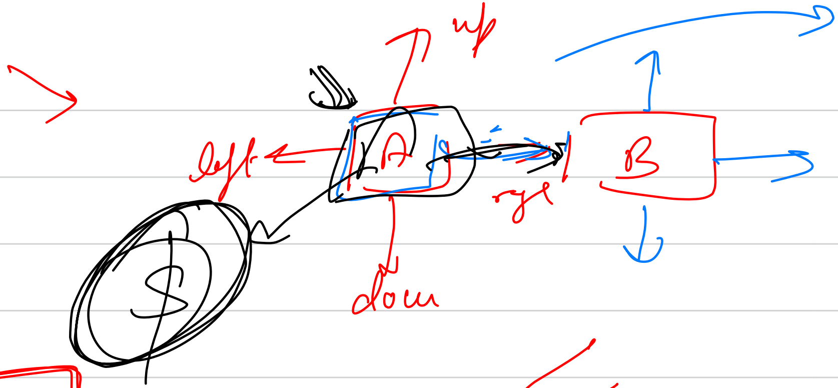
2 card

explore every direction

DFS

(Recursive code): 2

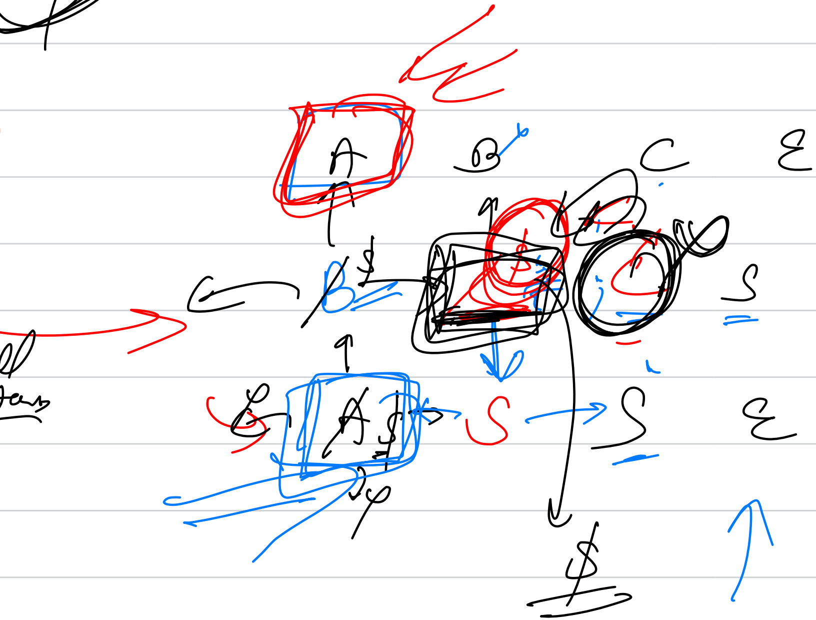
3) base case
(recursive intuitive)
self work

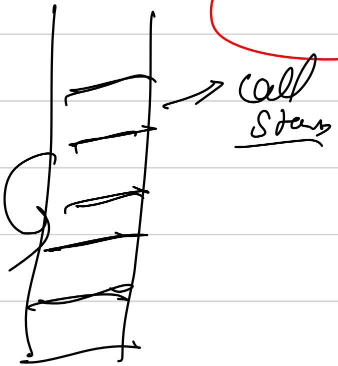


infinite recursion

visited array m^2

DFS





A B A

Back tracking

~~step~~

→ brute

→ we are not able to recurse further

no need of looping

(Space Complexity) → O(len(word))

call stack

return

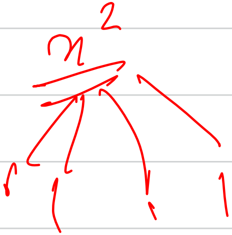
sum firstchar

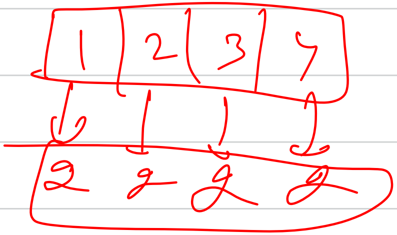
(Time Complexity)

$O(n^2)$ l 2

loose bound

Tight bound



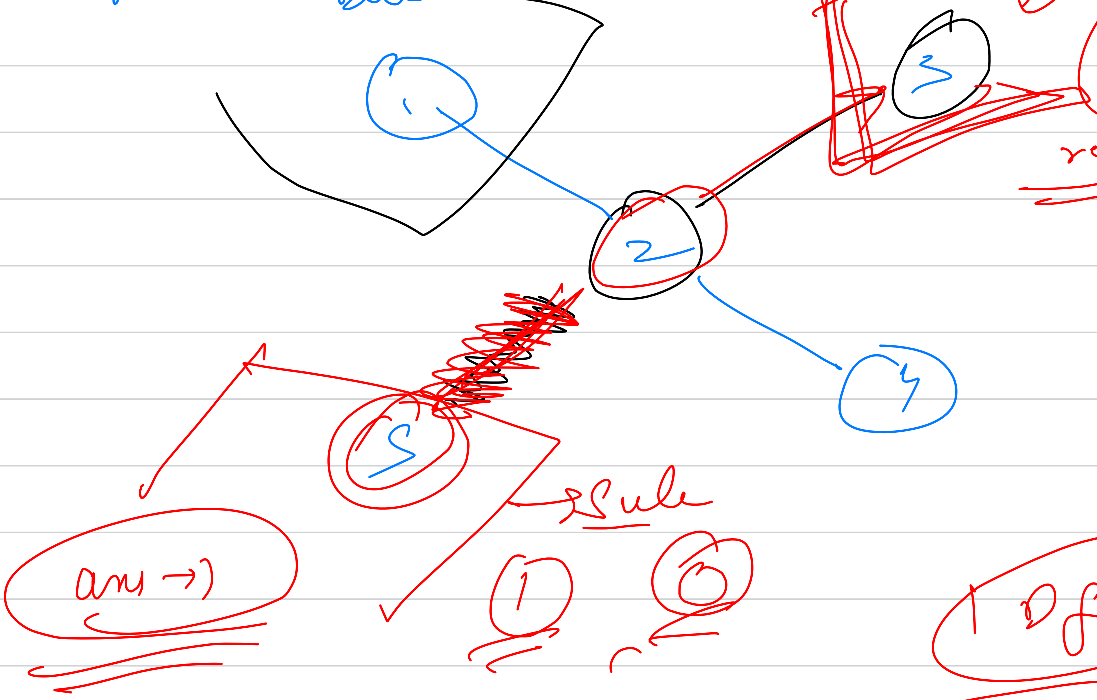


2^4

Hard

Qs You have an undirected tree of n nodes. Some Nodes have color 'A', some have color 'B' Some have color 'C'. It is sure that node for color A & B are present. Find the no. of edges you can remove such that removing one edge divide the tree in 2 components where in one component no node have a color 'A' & in one component no node have a color 'B'.

A → Red B → Blue C → Black



Subtrees

count red

count blue = 0

red

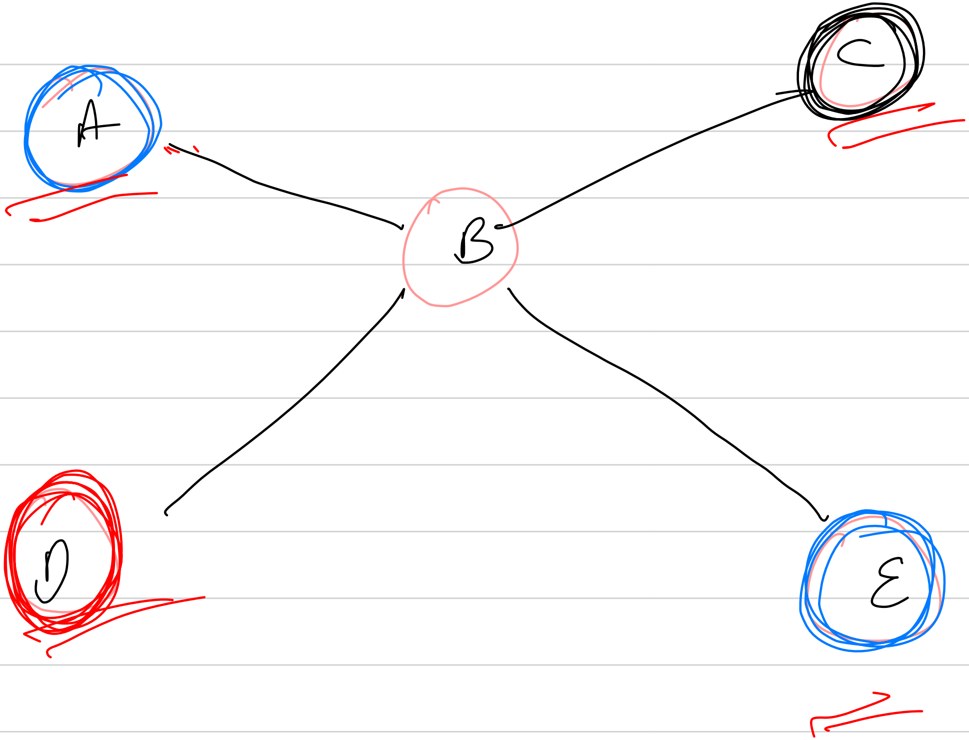
$n \le 3\times10^5$

DFS

DFS → edge

graph?

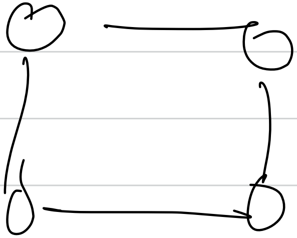
red==1
blue==3



Q → Given a graph, find if the graph contains cycle or not?

$n \le 10^5$

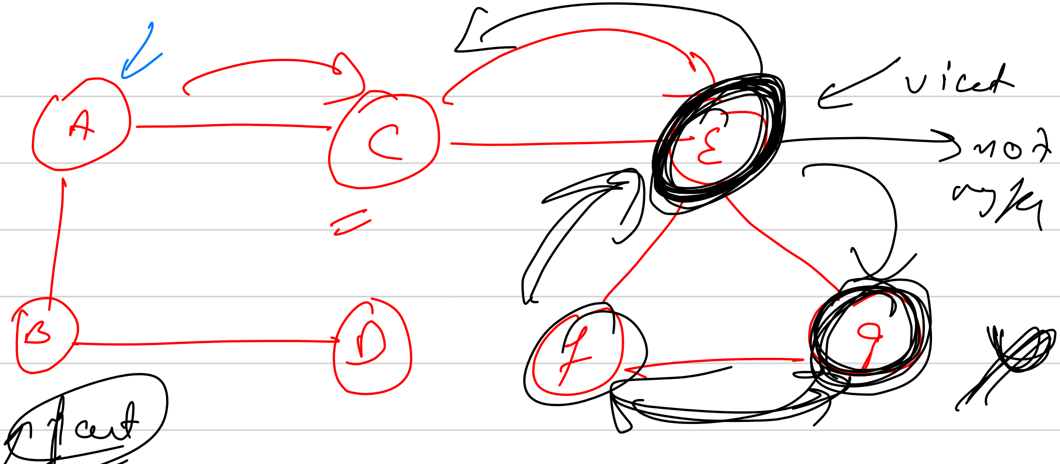
$e \le 10^5$



→ True

~~visited~~

visited



parent

DFS

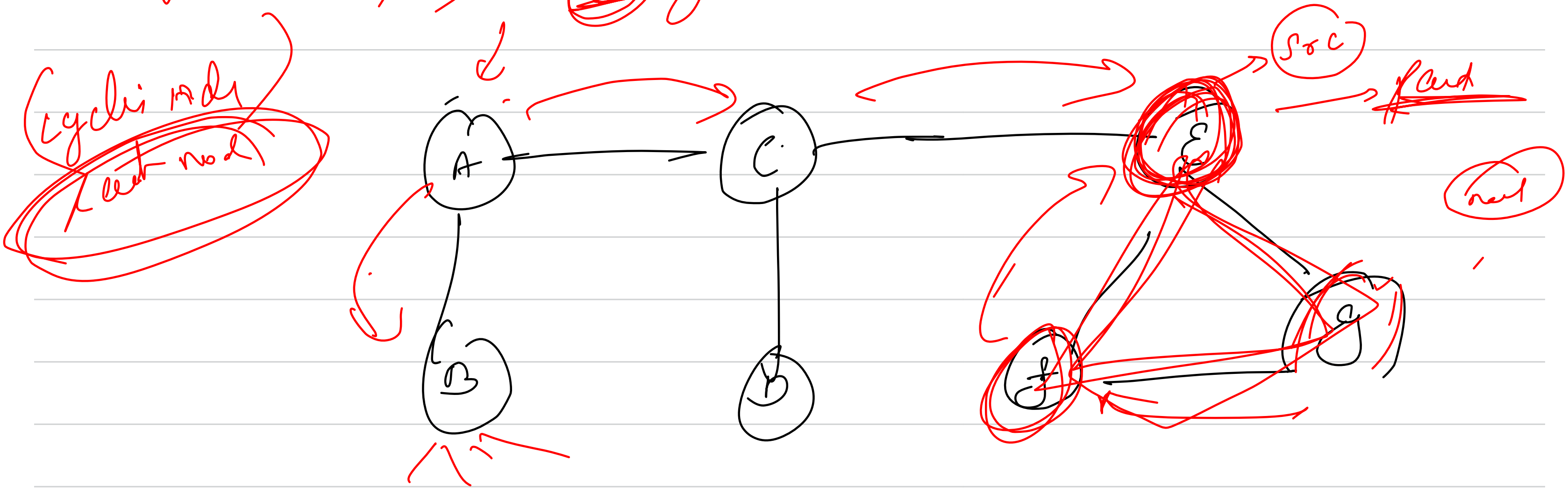
(A, -1) → (C, A) → (E, C) → (g, E)

→ (f, g)

DFS keeping a track of my parent & visited nodes

if you found a visited node which is not the parent of current node, you found a cycle.

visit (A, B, C, E, g, f



(A, -1) → (B, A) → (C, A) → (E, C) → (g, E)

(f, g) →