Greedy Algos contd... (Activity Selection Problem) (n, m)

* Given a list of activities with their start & finish times, calculate the maximum number of activities that can be performed by a person, based on the condition that he/she can only perform one single activity at a particular time without any overlapping.

(s, f) → (Comparator) ↓ (1, 4) 4 or 5 or 1s = 0

Activity	Start	Finish	Sort (Finish)	Output
A1	5	7 ④	A3 (1, 4)	A3 (1, 4)
A2	8	9 ⑤	A6 (3, 5)	
A3	1	4 ①	A5 (0, 6)	A1 (5, 7)
A4	5	9 ⑥	A1 (5, 7)	
A5	0	6 ③	A2 (8, 9)	A2 (8, 9)
A6	3	5 ②	A4 (5, 9)	

* Minimum Number of Platforms: → (output = 3)

arrival[ ] = { 900, 940, 950, 1100, 1500, 1800 };
departure[ ] = { 910, 1200, 1120, 1130, 1900, 2000 };
dep. Sort[ ] = { 910, 1120, 1130, 1200, 1900, 2000 };
0 1 2 3 4 5

Times in AM to PM format.

a (900 940 950) i
d (1200 910 1120) j x

900 940 950
910 1120 120

platforms: S | U — 1, 2, 3, 2, -1, -1, -1
maxplatforms: S | U — 1, 2, 3, 2 → ③

* 2 Unique elements: (XOR) (ECE) rsb ∧ arr[i] G1 = 1, 3, 1 ; G2 = 2, 4, 2

arr = { 1, 3, 2, 4, 1, 2 } = xorAll = 7

2's: rsb = xorAll & −xorAll

= 7 & −7

= 0111 & 1001

rsb = (0001) = 1

0111 / 1000 / +1 0001 / 1001 / 0111 / 1

0001 / ∧① 0001 / ∧③ 0011 / ∧① 0001 → ① ; 0001, 0011, 0001 → 0011 ③ ; 0010, 0100, 0010 → 0100 ④

n1 = 0
n2 = 0 1

for (i = 0, i < n, i++) {
  if (arr[i] & rsb == ①) ⇒ 1 — (1∧3∧1) ③
    n1 ∧= arr[i]
  else
    n2 ∧= arr[i] — 2∧4∧2 = ④
}

rsb = 0001
= 1

0011 / 0100 = = 0 == 1 last

0111 / 0111 / X 1 = 0

1, 2, 3, 4, 1, 2
0001 0001 → 0 → 0
0010 0001 → 3 → 1
0011 0001 → 2
0100 0001 → 5
0001 0001 → 0 → 0
0010 0001 → 3 → 1

* Chocolate Distribution Problem or (Minimum Absolute Difference)

* (Sliding Window Problem)

[O/p = 2] arr = { 7, 3, 2, 4, 9, 12, 56 } (m = 3) students

Sort (n log n) 2, 3, 4, 7, 9, 12, 56

i = 0; i + m − 1
0 + 3 − 1 = 2
1 + 3 − 1 = 3

$a_j - a_i$
= 4 − 2 = ②
7 − 3 = 4
9 − 4 = 5
12 − 7 = 5
56 − 9 = too large

min value — m_n, m_x — [2, 3] 2 → 3

(Trie) Data Structure (Suffix Tree)

* Contacts in Phone
* Word Dictionary (obsolete)
* Auto complete feature in Browsers

insert( )
search( )

* arya, true ✓
* aryan, true ✓
* arya man, true ✓
* aryabhatta, true ✓

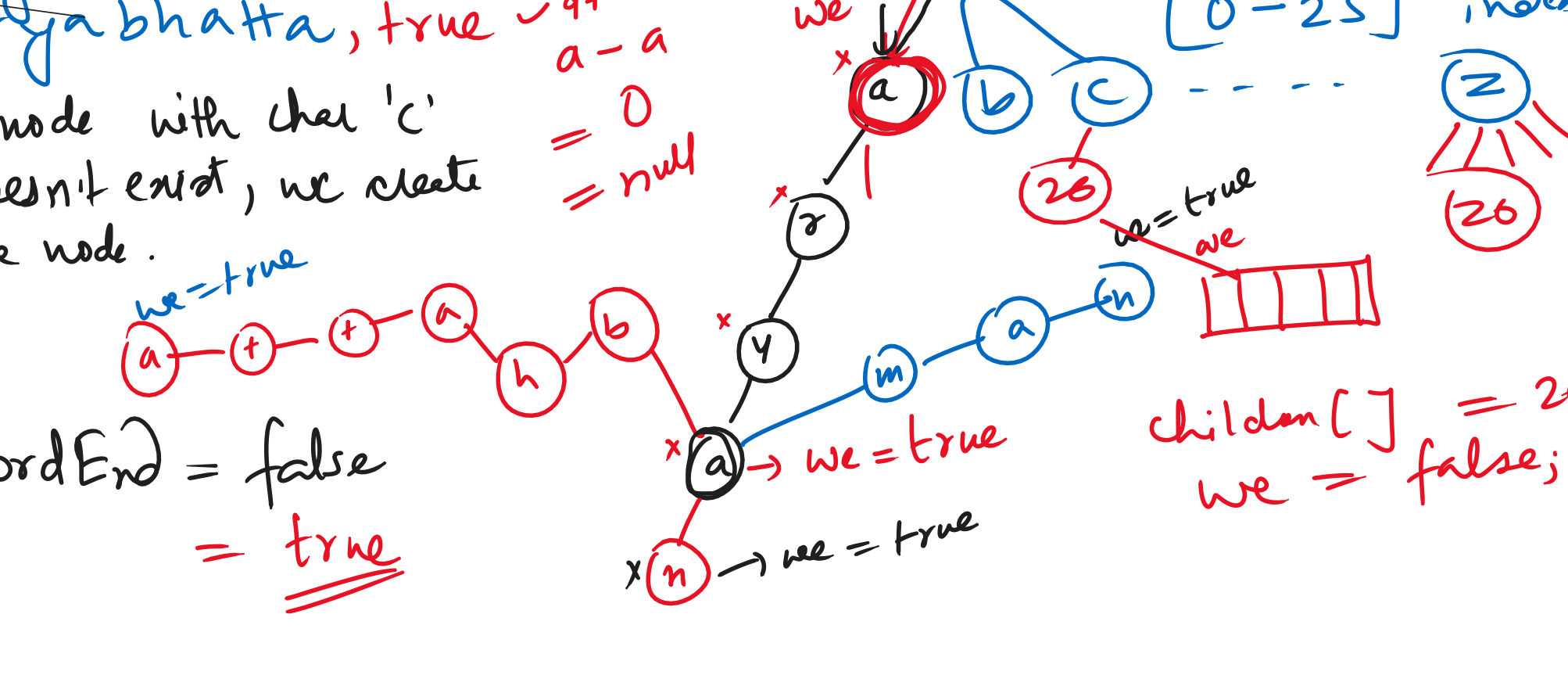
HashMap <K, V> (String, Boolean)

(Space | Memory)

* arya, true ✓
* aryan, true ✓
* arya man, true ✓
* aryabhatta, true ✓ 97 97 a − a = 0 = null

* If node with char 'c' doesn't exist, we create the node.

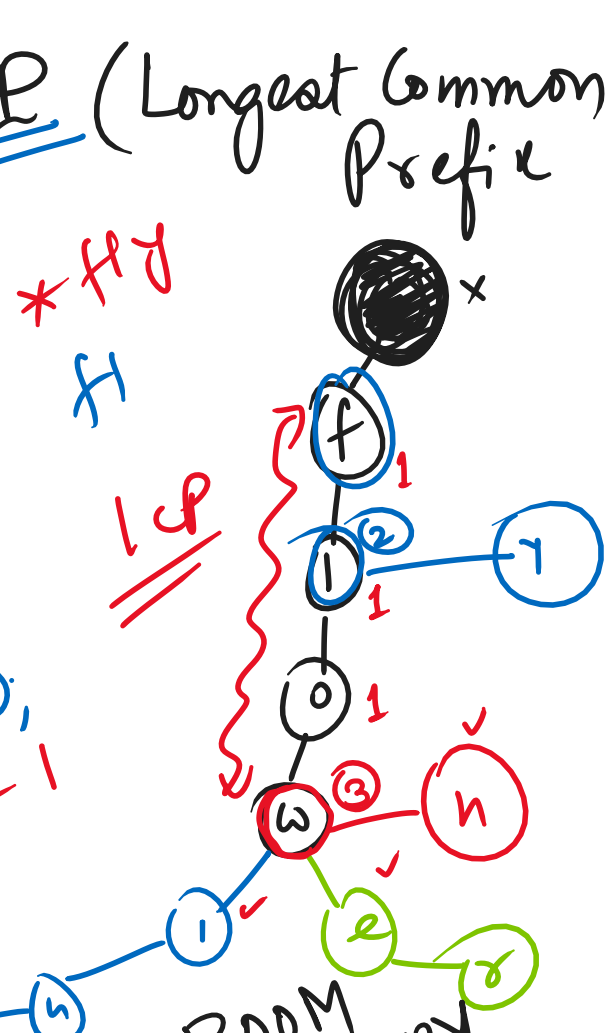
wordEnd = false = true



* LCP (Longest Common Prefix)

* flow
* flown
* flowing
* flower
LCP = flow
children[26]
we = false
children count = 0, cc == 1

* fly
fl



(Maven) Build Tool

(MySQL WorkBench) DBMS

code, coder, coders, coders arcade

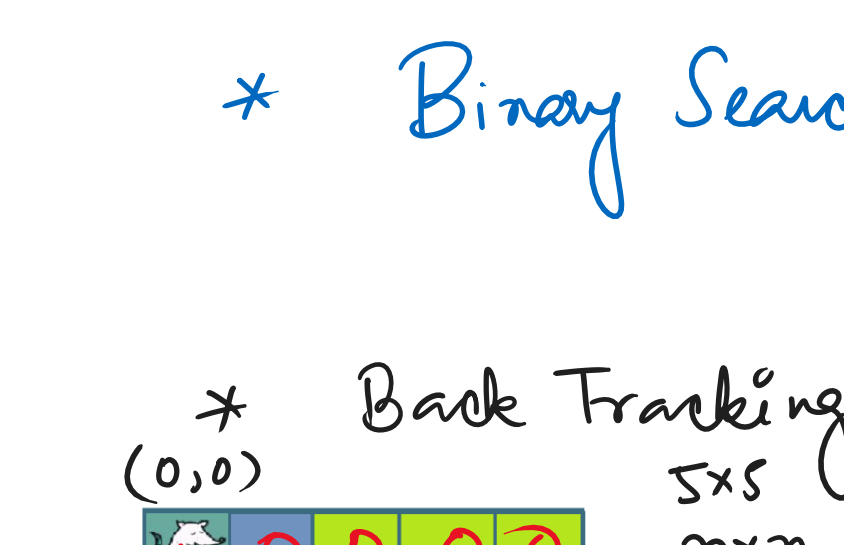
NIGHT TITING + IntelliJ Idea (Community Edition)

LISTEN SILENT ⇒ (JDK − 17) or above

* Binary Search + Recursion → DJ → DP; Tree, Graph

* Back Tracking: → 5x5 matrix, nxn matrix

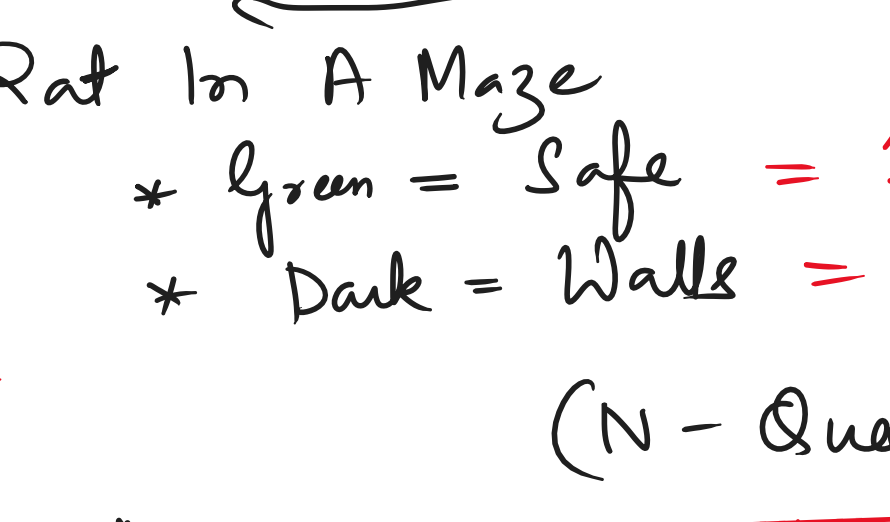
Rat In A Maze
* Green = Safe = 1
* Dark = Walls = 0



start (x, y) {
→ forward (x+1, y)
→ downward (x, y+1)
cell = 0
}

nxn all zeroes resultant array.

(N − Queens)



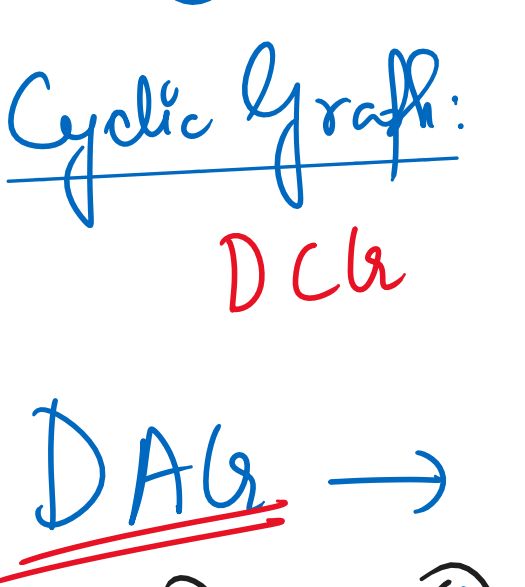
4x4

Non-Linear Data Structures ⇒ The Graph Data Structure

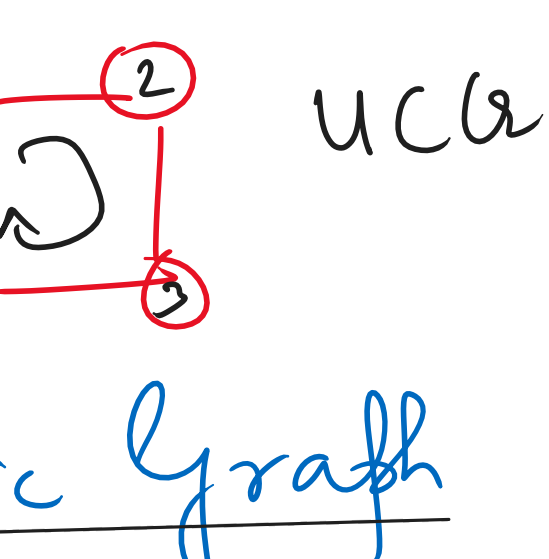
A graph is a non-linear data structure which contains entities called "nodes" connected to each other via "edges".

* Depending on the connectivity of the nodes to one another, there can be various categories of graphs:

① Undirected Graph (Unweighted)

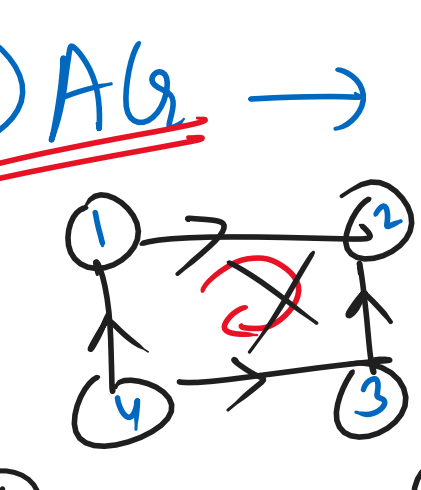


④ Undirected Weighted Graph

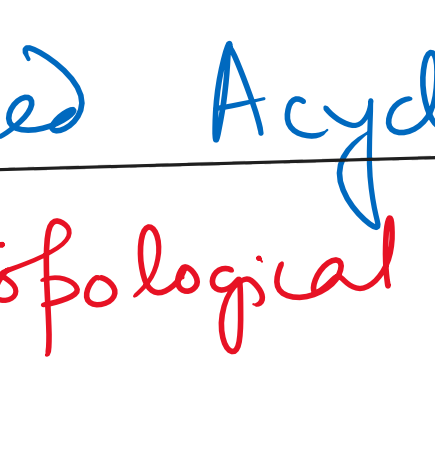


* The edges have some value (weight)

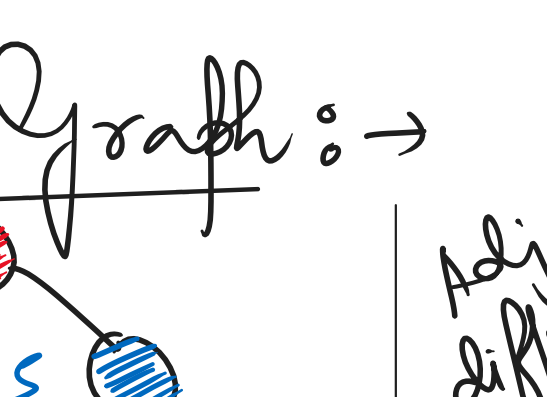
(iii)



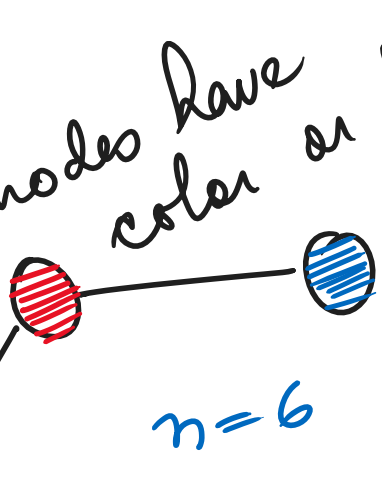
(iv)



(v) Cyclic Graph: DCG

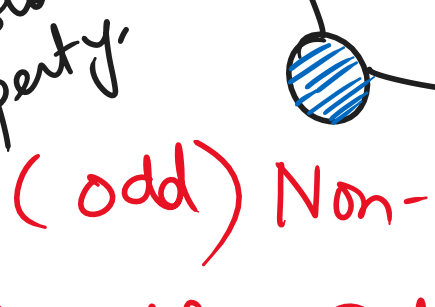


(vi)



UCG

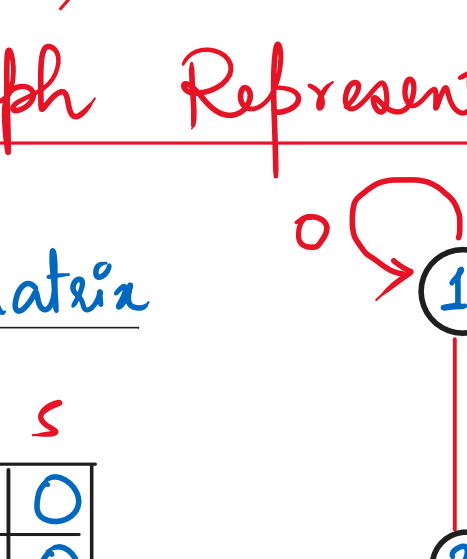
(vii) DAG → Directed Acyclic Graph



(Topological Sort) ***

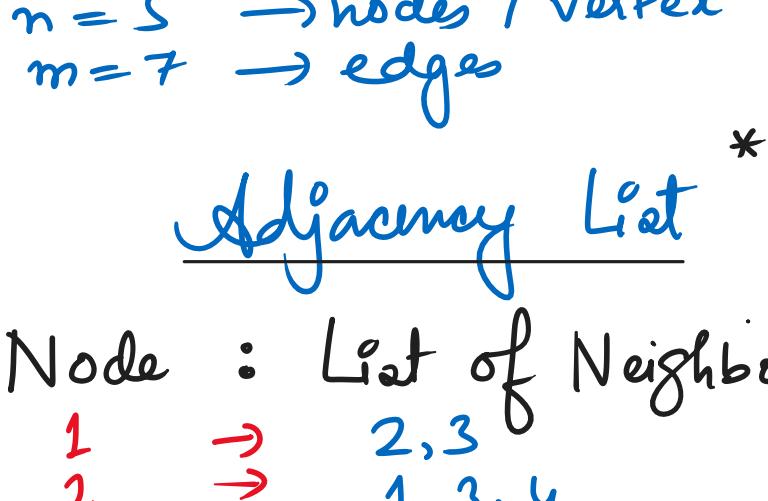
(viii) Bipartite Graph: →

Adjacent nodes have same color or property.



(odd) Non-bipartite

Adjacent nodes have different color or property



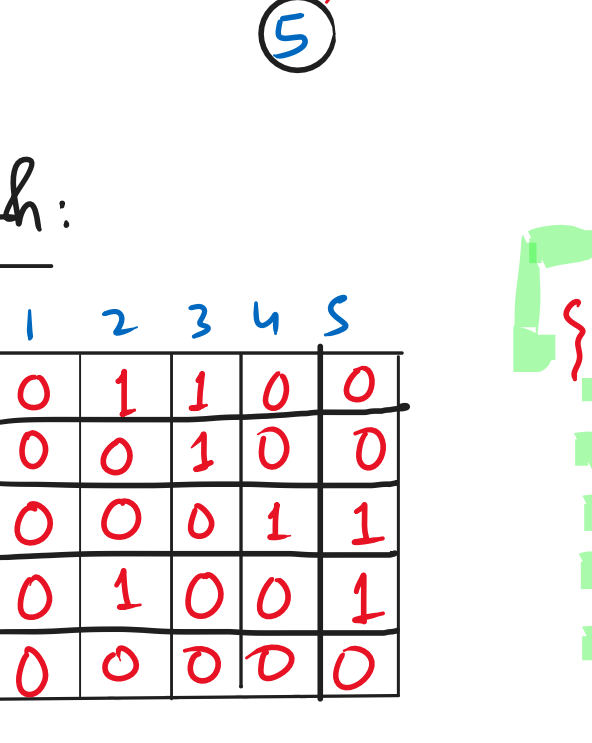
Bipartite (Even)

* Graph Representations: →

n = 5 → nodes / vertex
m = 7 → edges

Adjacency Matrix

	1	2	3	4	5
1	0	1	1	0	0
2	1	0	1	1	0
3	1	1	0	1	1
4	0	1	1	0	1
5	0	0	1	1	0



Adjacency List ***

Node : List of Neighbours
1 → 2, 3
2 → 1, 3, 4
3 → 1, 2, 4, 5
4 → 2, 3, 5
5 → 3, 4

Array List

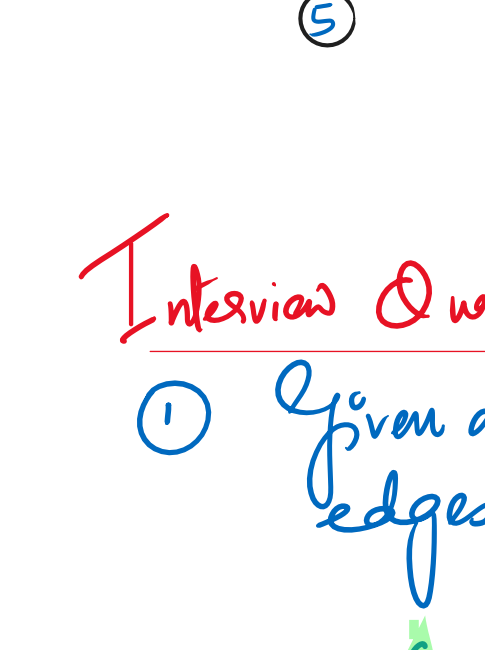
N : LON
{ 1 → 2, 3 }
{ 2 → 3 }
{ 3 → 4, 5 }
{ 4 → 2, 5 }
{ 5 → }

L<L<I>>

AL { [ ] }

AL{ {AL1}, {AL2}, {A3}, {A4}, {A5} }

Directed Graph:



	1	2	3	4	5
1	0	1	1	0	0
2	0	0	1	0	0
3	0	0	0	1	1
4	0	1	0	0	1
5	0	0	0	0	0

Interview Questions: → Generic <T> <?> (Wild Card) IT (CSE | ISE) (template) C++

① Given an undirected graph, find the number of nodes & edges of the graph.

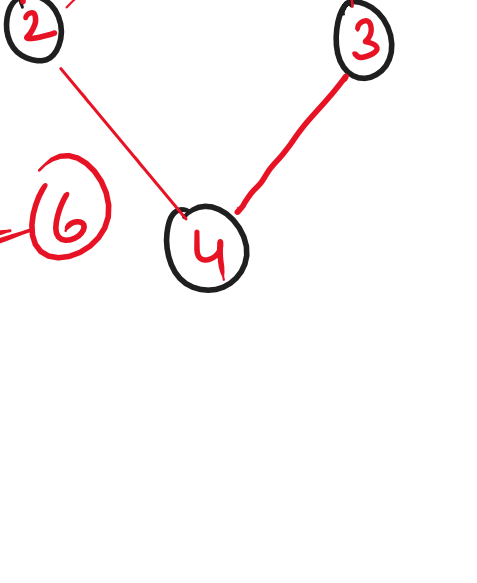
{IBM, HCL, HP, Intel, Bosch, Wipro}

(FDP)

List<List<Int>> adjList

Input: {
{ 0 → (1, 2) } 2
{ 1 → (0, 2, 3) } 3
{ 2 → (0, 1, 4) } 3
{ 3 → (1, 4) } 2
{ 4 → (2, 3) } 2
} 12 / 2 = ⑥

adjList.size()



o/p
6 edges
5 nodes

Big Data ( )