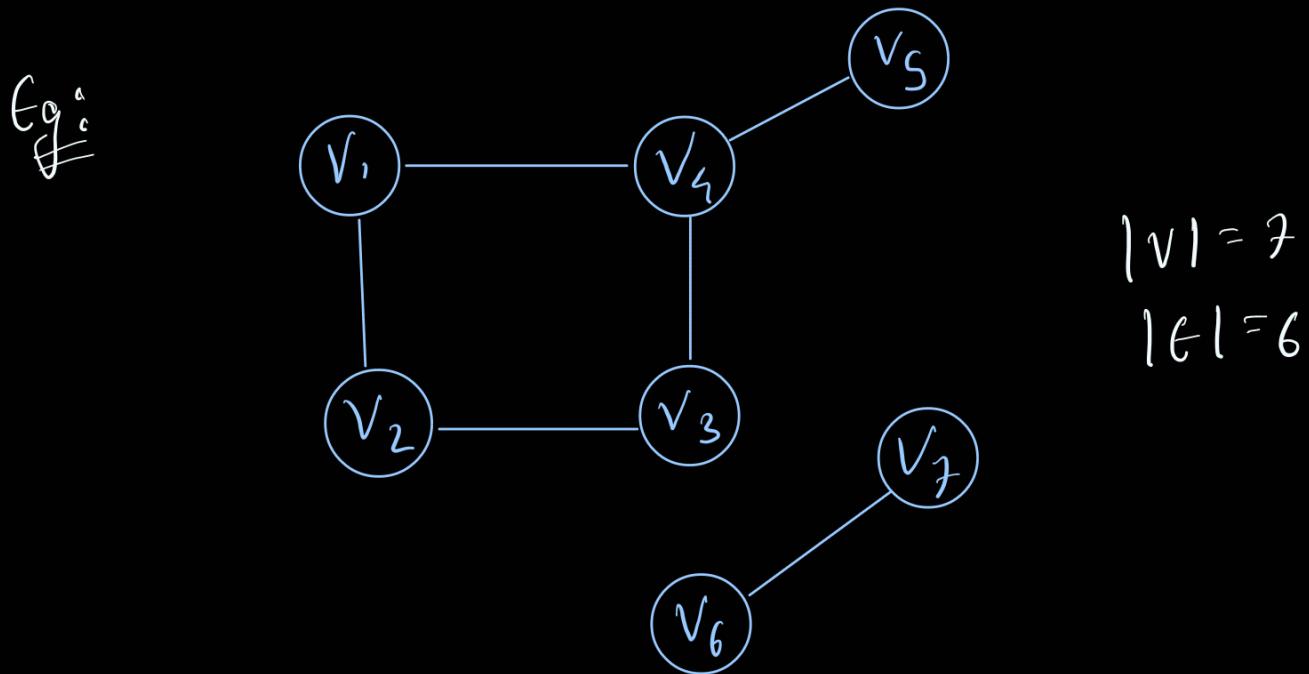




What is a Graph?

Formal:

→ A graph $G_1 = (V, E)$ consists of a vertex set V and edge set E , such that $E \subseteq V \times V$



$$V = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$$

$$E = \{(v_1, v_2), (v_1, v_4), (v_2, v_3), (v_3, v_4), (v_4, v_5), (v_6, v_7)\}$$

Adjacency Matrices:

Denotes if there is an edge b/w two vertices or not.

If edge b/w i and j nodes $\Rightarrow a_{ij} = 1$
else $\Rightarrow a_{ij} = 0$

- In this case, the matrix will be 7×7 .

	1	2	3	4	5	6	7
1	0	1	0	1	0	0	0
2	1	0	1	0	0	0	0
3	0	1	0	1	0	0	0
4	1	0	1	0	1	0	0
5	0	0	0	1	0	0	0
6	0	0	0	0	0	1	0
7	0	0	0	0	0	1	0

Sym. about
main
diagonal

→ What is the maximum number of edges that a graph $G_1 = (V, E)$ on n vertices ($|V| = n$) can have?

$$|E| = {}^n C_2 = \boxed{\frac{n(n-1)}{2}}$$

(Edge b/w every
two nodes)

→ How many different graphs are there on the vertex set

$$V = \{1, 2, \dots, n\}$$

→ (${}^n C_2$ ways of this)

Pick any two vertices from the set,

there are 2 possible cases, either 1 or 0.

$$(2) {}^n C_2 = \boxed{(2)^{\frac{n(n-1)}{2}}}$$

$$\text{Proof: } {}^n C_0 + {}^n C_1 + {}^n C_2 + \dots + {}^n C_n = (2)^n$$

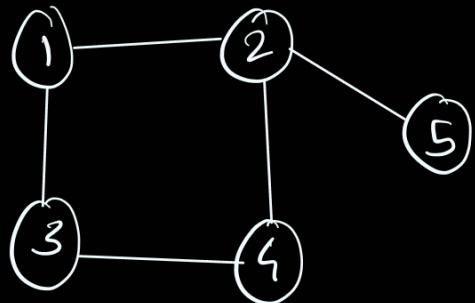
\uparrow \uparrow
 zero edges one edge

$$= (2)^{\frac{n(n-1)}{2}}$$

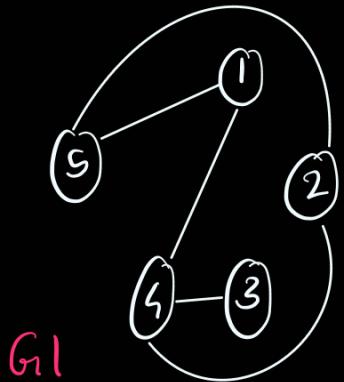
Matrix

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

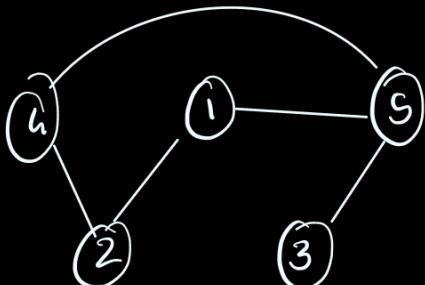
Graph



Eg:

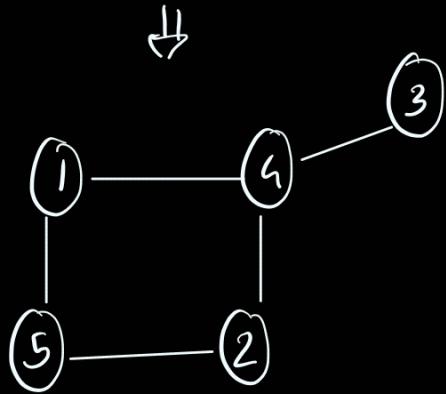


G1

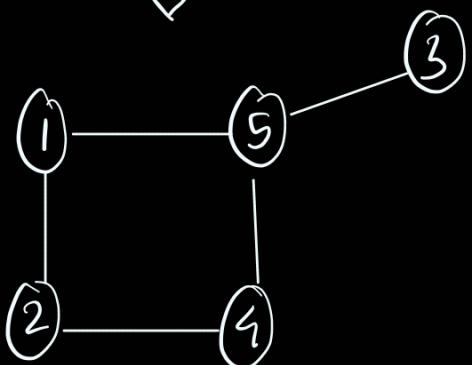


G2

G1 and G2
are isomorphic



↓



Two graphs are isomorphic to each other if the rows or columns of the adjacency matrix of one of them can be exchanged so that these are same as the adjacency matrix of other.

NOTE:

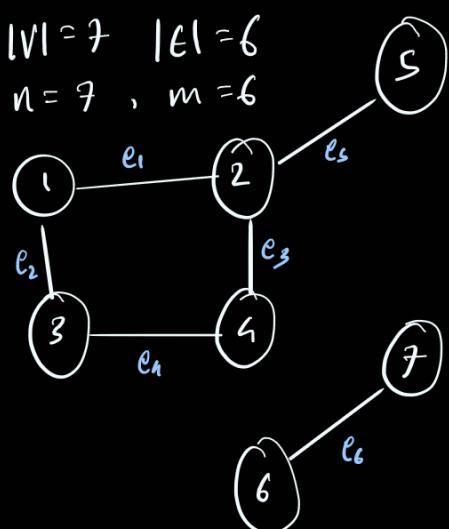
Given two graphs G_1 and G_2 , both with n vertices, check if they are isomorphic or not.

$$TC \text{ (Best Alg)} = \boxed{2^{O(\log n)^3}} \quad (\text{Babai, 2016})$$

※ 3 Ways of Representing Graph :

- (A) Adjacency Matrix [$n \times n$ matrix, 2D array]
- (B) Incidence Matrix [$n \times m$ matrix, 2D array]
- (C) Adjacency List [n -linked lists, Space: $2n$]

Let the graph be:



(A) Adjacency Matrix:

	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0
2	1	0	0	1	1	0	0
3	1	0	0	1	0	0	0
4	0	1	1	0	0	0	0
5	0	1	0	0	0	0	0
6	0	0	0	0	0	0	1
7	0	0	0	0	0	1	0

If $(v, v) \in E$

then,

$$\text{Adj}[v][v] = 1$$

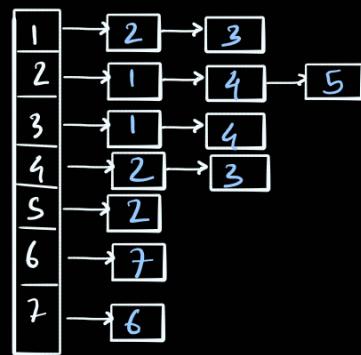
$$\text{Space} = n \times n = 7 \times 7$$

(B) Incidence matrix:

Space:
 $n \times m$
 $= 7 \times 6$

$$\begin{matrix} & e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 2 & 1 & 0 & 1 & 0 & 1 & 0 \\ 3 & 0 & 1 & 0 & 1 & 0 & 0 \\ 4 & 0 & 0 & 1 & 1 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 1 & 0 \\ 6 & 0 & 0 & 0 & 0 & 0 & 1 \\ 7 & 0 & 0 & 0 & 0 & 0 & 1 \end{matrix}$$

(C) Adjacency List:



Disadvantage:

If $n = 20,000$, then 19.998

elements in each column will be zero.

$$m \leq \frac{n(n-1)}{2}$$

Adjacency List

Adjacency Matrix

Space
Complexity

$$O(2m)$$

$$O(n^2)$$

$(v, u) \in E?$

$$O(n)$$

$$O(1)$$

All neighbours
of a vertex

$$O(n)$$

$$O(n)$$

All edges

$$O(ntm)$$

$$O(n^2)$$

Insert edge

$$O(1)$$

$$O(1)$$

Delete edge

$$O(n)$$

$$O(1)$$

Neighbour:

A vertex v is said to be a neighbour of u , if $(v, u) \in E$

Degree:

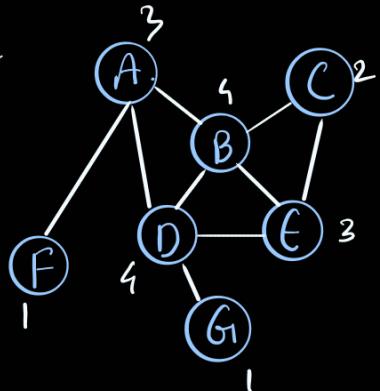
The degree of a vertex is the number of neighbours it has.

$$0 \leq \deg(v) \leq n-1$$

~~Handshaking Lemma:~~

If each person at a party shakes hands with those that the person is friends with, then the total number of handshakes is equal to half the [(cumulative total of) the number of handshakes each person made].

Eg:



$$\begin{aligned} & \deg(A) + \deg(B) + \dots + \deg(F) \\ &= 3 + 4 + 2 + 4 + 3 + 1 + 1 \\ &= 18 \end{aligned}$$

$$\begin{aligned} n(\text{edges}) &= 9 \\ \therefore n(\epsilon) &= \frac{\sum \deg(v)}{2} \end{aligned}$$

Fact: No. of people, who shakes odd number of hands is even.

$$\stackrel{M-1}{=} \text{let } n = |V|$$

$$\stackrel{M-1}{=} \text{Proof: and, } m = |\epsilon|$$

As each edge contribute to the degrees of two vertices,

$$\sum \deg(v_i) = 2m$$

let x = no. of vertices with odd degree.

$$\therefore \sum \deg = (n-x)(\text{even}) + (x)(\text{odd})$$

$$\therefore 2m = \text{even} + x \times \text{odd}$$

$$\therefore \text{even} = \text{even} + x \times \text{odd}$$

$$\therefore \text{even} = x \times \text{odd}$$

$$\therefore x = \text{even}$$

M-2 Empty Graph: 0 is even

There are 3 cases:

Case ①: $\begin{array}{cc} \textcircled{A} & \textcircled{B} \\ \text{odd} & \text{odd} \end{array} \Rightarrow \begin{array}{cc} \textcircled{A} & \textcircled{B} \\ \text{even} & \text{even} \end{array}$

Case ②: $\begin{array}{cc} \textcircled{A} & \textcircled{B} \\ \text{even} & \text{even} \end{array} \Rightarrow$

Case ③: $\begin{array}{cc} \textcircled{A} & \textcircled{B} \\ \text{odd} & \text{even} \end{array} \Rightarrow$

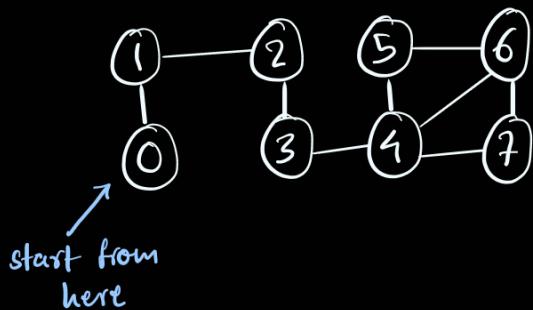
Breadth - First - Search (BFS):

- Start with a vertex A.
- Then explore the neighbours of A.
- Repeat this for each of the neighbours of A.
And so on...
- Keep exploring until there is nothing left to explore.

Pseudocode:

```
BFS(A)
{
    push (null, A) to queue Q.
    while (Q is nonempty)
    {
        p = pop(Q)
        if (visited(p) == FALSE)
        {
            visited(p) ← TRUE
            parent(p) ← null
            for each edge (p, v)
                push (v, p) to Q
        }
    }
}
```

Eg: Let $|V| = 8$, $|E| = 9$



start from here

0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0

visited:

BFS Traversal : ans

1) queue: (null, 0)
ans: 0

visited:

0	1	2	3	4	5	6	7
1	0	0	0	0	0	0	

2) queue: (0, 1)
ans: 0 1

visited:

0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0

3) queue: (1, 2)
ans: 0 1 2

visited:

0	1	2	3	4	5	6	7
1	1	1	0	0	0	0	0

4) queue: (2, 3)
ans: 0 1 2 3

visited:

0	1	2	3	4	5	6	7
1	1	1	1	0	0	0	0

5) queue: (3, 4)
ans: 0 1 2 3 4

visited:

0	1	2	3	4	5	6	7
1	1	1	1	1	0	0	0

6) queue: (4, 5), (4, 6), (4, 7)
ans: 0 1 2 3 4 5

visited:

0	1	2	3	4	5	6	7
1	1	1	1	1	1	0	0

7) queue: (4, 6), (4, 7)
ans: 0 1 2 3 4 5 6

visited:

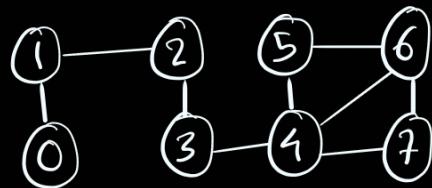
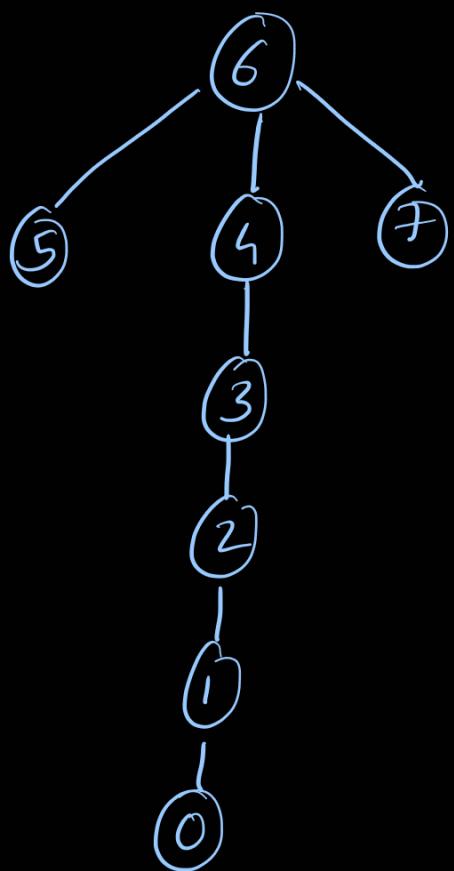
0	1	2	3	4	5	6	7
1	1	1	1	1	1	1	1

8) queue: (4, 7)
ans: 0 1 2 3 4 5 6 7

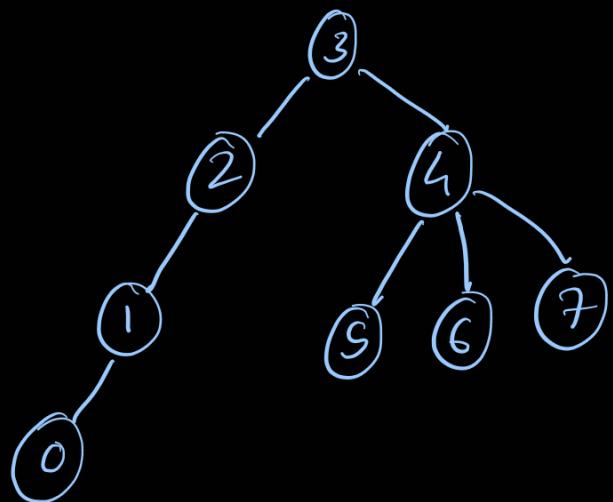
Q empty \rightarrow loop terminates.

Q: What will be the BFS tree for

(i) BREADTH FIRST SEARCH(6) :



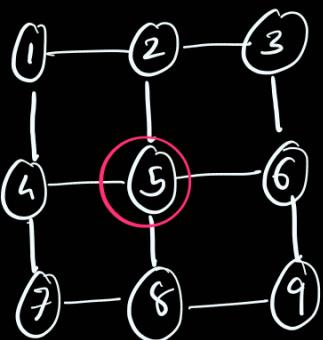
(ii) BREADTH FIRST SEARCH(3) :



Q. Given graph:

$$|V| = 9$$

$$|E| = 12$$



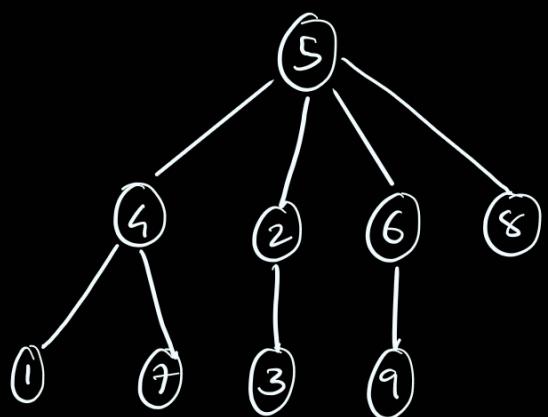
What is a forward edge:

An edge from vertex to its parents

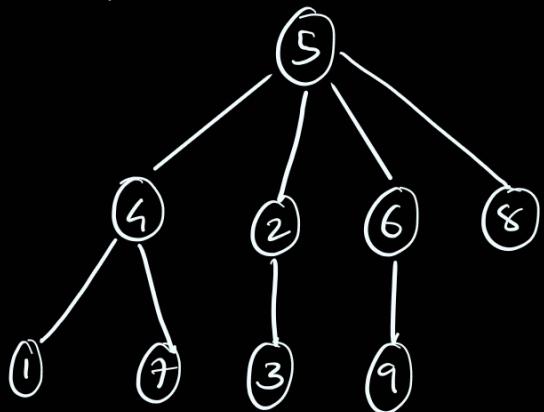
Cross edge:

Edge b/w two vertices at the same level

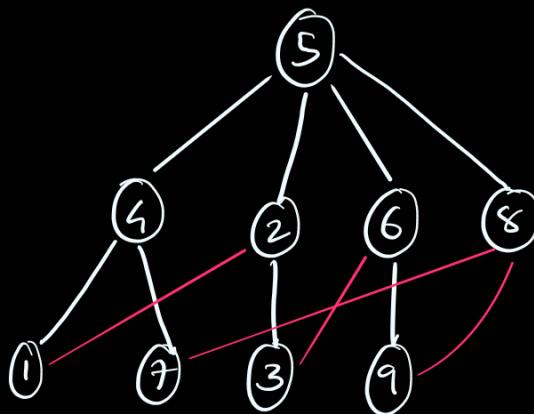
BFS Tree:



forward edges = 8
(f)



Cross edges : 0
(c)



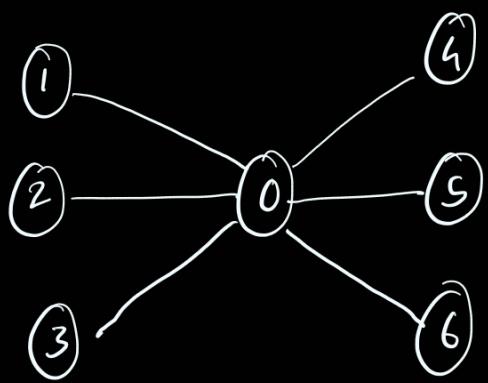
back edge: 4
(b)

$$f + c + b = |E| \quad \checkmark$$

$$8 + 0 + 4 = 12$$

→ A graph which does not have any cross edges is a bipartite graph.
→ No odd length cycle.

Eg: Given Tree:

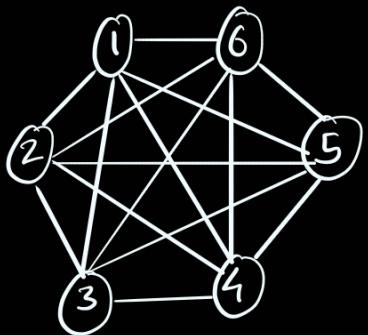


Here, every edge is a forward edge.

A graph which has 0 cross and back edges is a tree.
(All edges are forward edges)

$$|V|=7 \quad |E|=6$$

Eg: Complete Graphs:



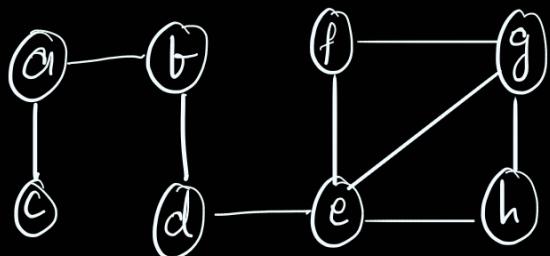
DEPTH FIRST SEARCH

Pseudo code:

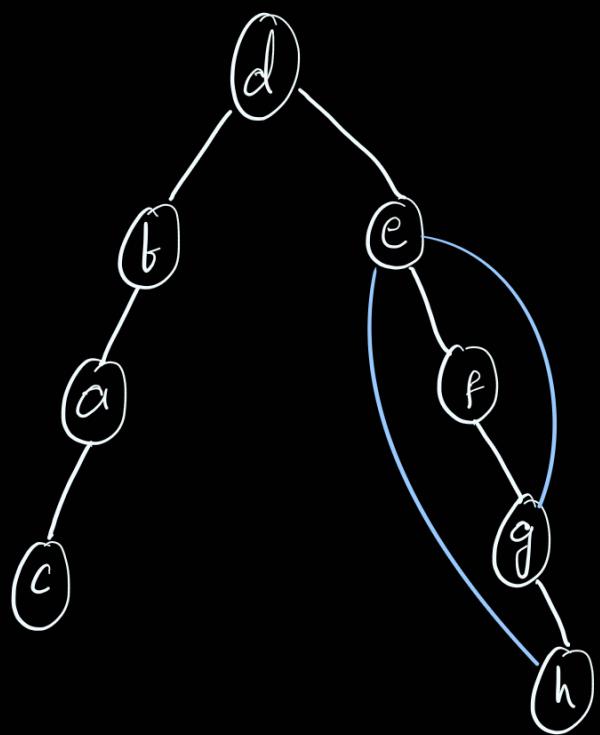
```
DFS(s)
{
    push(null, s) to stack S
    while (S is not empty)
    {
        pop(p, v) from S
        if (visited(v) == FALSE)
        {
            visited(v) ← TRUE
            parent(v) ← p
            for (each (v, w) ∈ E)
            {
                push(v, w) to S
            }
        }
    }
}
```

- Look at a neighbour of s .
- Then, look at a neighbour of neighbour of s .
- Keep traversing in this way and terminate when everything is traversed.

Eg: Consider the following graph:



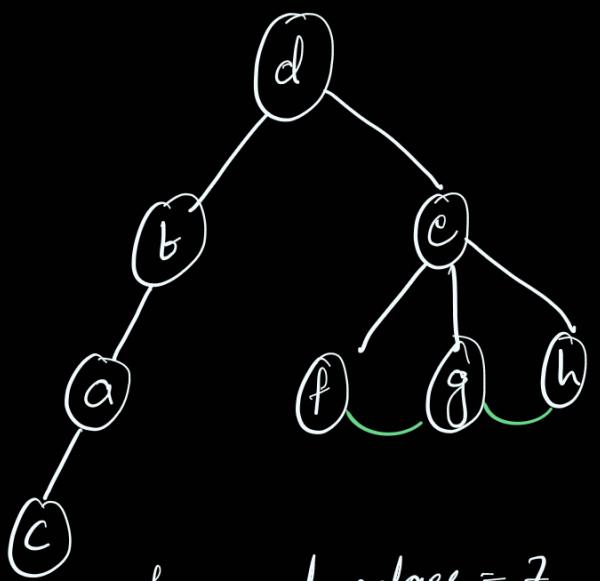
DFS(d)



forward = 7

back edges = 2

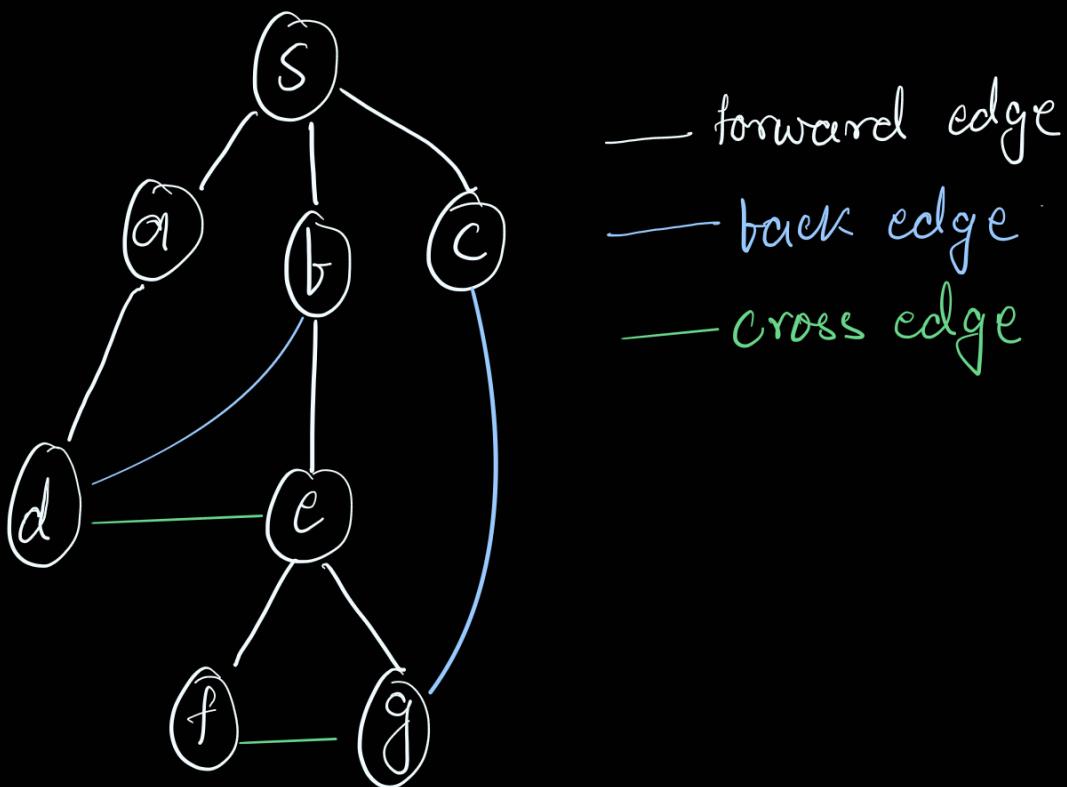
BFS(c)



forward edges = 7

cross edges = 2

Q. Construct the graph for which this is a BSF traversal. You may add more edge if you want.



Facts:

1) Let v be a vertex in level i of a BFS traversal of a graph.

Then, all its neighbours are in one of the levels: $\{i-1, i, i+1\}$
i.e

2) If \exists an edge b/w vertex at level i and level j ,
then $|i-j| \leq 1$.

Hence, the given BFS traversal is wrong.

NOTE: The level of a vertex in its BFS traversal from S is its distance from S . (only true for BFS).

Another Fact:

In $O(|V|+|E|)$, it is possible to compute distances of all vertices from s .

Proof: Do BFS traversal. Let $|V|=n$, $|E|=m$

let $\text{Dist} : \begin{array}{ccccccc} 0 & 1 & 2 & 3 & \cdots & \cdots & n \\ \boxed{0} & \boxed{0} & \boxed{0} & \boxed{0} & & & \boxed{0} \end{array}$

Let $s=1$,

then the array "Dist" will store the distance of i from 1 .

→ Initially push(1,0)
[vertex, level]

then traverse all of its !visited neighbours:

for each neighbour (k) of (s, level) :

if $\text{!visited}(k)$

$\text{Dist}[k] = \text{level} + 1$

$\text{push}(k, \text{level} + 1)$

$\text{visited}(k) = 1$

~~X~~ Types of Graph:

Type	Definition
① Empty Graph	A graph with no edges
② Complete Graph	A graph in which all pairs of vertices are edges
③ Path	A chain of connected vertices without repetition
④ Cycle	A loop of connected vertices without repetition
⑤ Tree	Exactly one path b/w every pair of vertices.
⑥ Bipartite Graph	A graph whose vertices can be partitioned into two sets, L & R such that, every edge has one end-point in L and one end-point in R.
⑦ Regular Graph	
⑧ Planar Graph	

NOTE: While doing BFS traversal on a Bipartite graph, any two vertices at the adjacent level are in different sets.

A graph is a Bipartite graph iff it has no cross edge.

Bipartite Graph:

- A graph is bipartite iff its BFS traversal has no cross-edge.
- A graph is bipartite iff it has no odd-length cycle.

Cycles in BFS:

- Odd cycles \Leftrightarrow cross edge
- Even cycle \Leftrightarrow back edge

Trees

- No odd cycles
- No even cycles
- Only forward edges.
- Every tree is a bipartite graph.

Disconnected Graphs:

- A graph is disconnected iff it contains two vertices U and V such that there is no path between U and V .
- The empty graph on $n \geq 2$ vertices is disconnected.

Graph Isomorphism:

- Two graphs are isomorphic if one of them can be relabeled such that their adjacency matrix are identical.
- To show that graphs are NOT isomorphic, show one property of one graph that does not exist in the other graph.

Undirected Graphs

$$\rightarrow G_1 = (V, E)$$

$$\rightarrow E \subseteq V \times V$$

$$\rightarrow (v, v) \notin E$$

$$\rightarrow (u, v) \in E \leftrightarrow (v, u) \in E$$

$$\rightarrow A(i)(j) = A(j)(i)$$

[Adjacency matrix is symmetric]

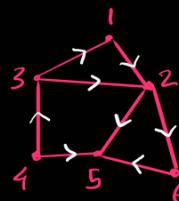
Directed Graphs

$$\rightarrow G_1 = (V, E)$$

$$\rightarrow E \subseteq V \times V$$

$$\rightarrow (v, v) \in E$$

Eg:



	1	2	3	4	5	6
1	0	1	0	0	0	0
2	0	0	0	0	1	1
3	1	1	0	0	0	0
4	0	0	1	0	1	0
5	0	0	0	0	0	0
6	0	0	0	0	1	0

Q Consider the following Graph:



(1) Put arrows on the 8-edges such that the resulting graph is acyclic.



(2) An undirected graph such that such it is impossible to convert it into a "acyclic" directed graph.

↗ any such graph.

Proof: → Given any undirected graph, put a arrow such that it goes from a larger number to a smaller number.

→ As $a > b > c \dots > a$ is not possible, the resultant directed graph will be acyclic.

Shortest Path:

* Single source shortest path (SSSP):

→ Given a directed graph G_1 and a vertex S of G_1 (source), compute shortest path from s to every other vertex of G_1 .

Eg:

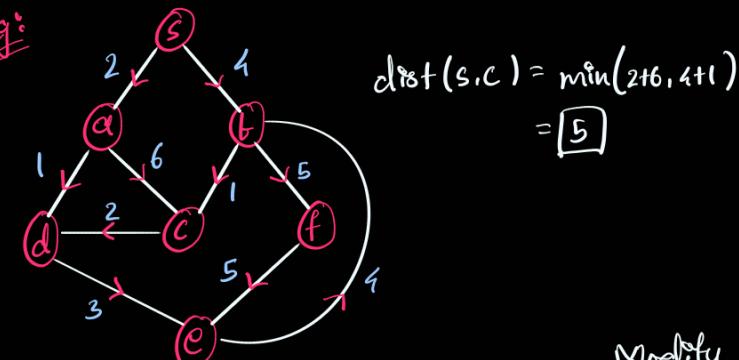
→ Run the BFS from s .

The level number for each vertex denotes the distance of that vertex from the source.

* Weighted edges:



Eg:



$$A \xrightarrow{1} B \equiv A \xrightarrow{1} O \xrightarrow{2} O \xrightarrow{3} O \xrightarrow{4} B$$

→ Modify the graph in the following way, so that we do not need to change the algorithm.