Subject _____ Sat Sun Mon Tue Wed Thu Fri

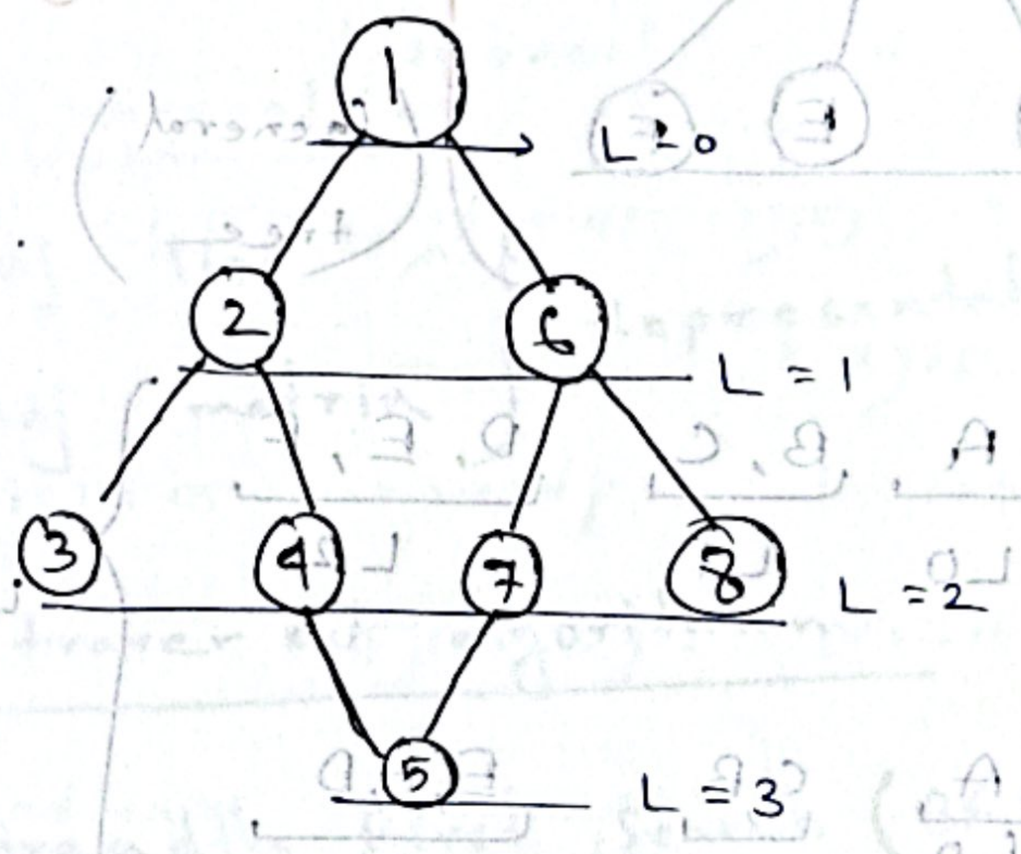
Date : / /

G-5

Breadth - First Search (BFS) | Traversal Technique in Graph :-

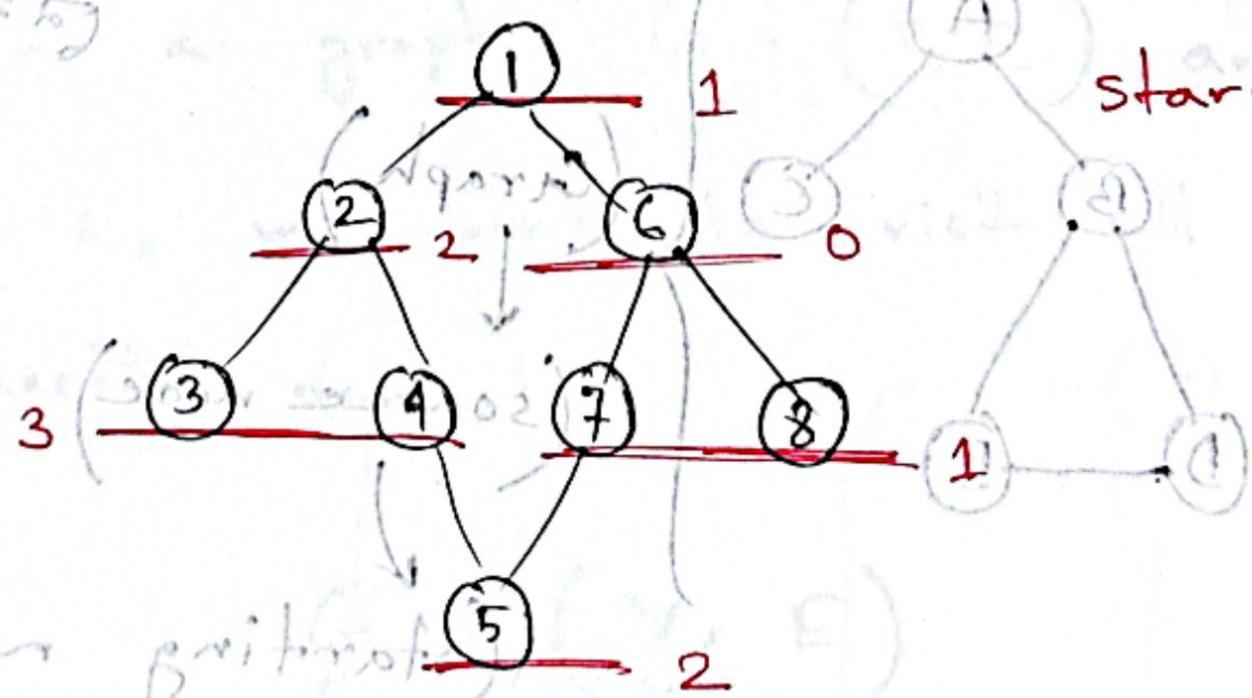
Breadth → (Level wise).

starting node → 1.



1 | 2 6 | 3 4 7 8 | 5

0 | 1 | 2 | 3



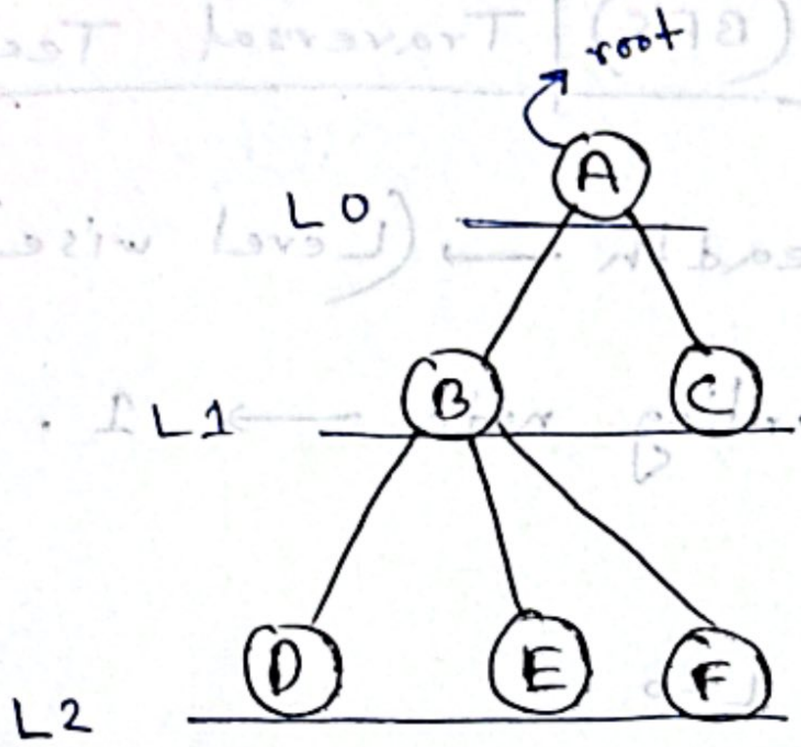
6 | 1 7 8 | 2 5 | 3 4

0 | 1 | 2 | 3

Subject _____ Sat Sun Mon Tue Wed Thu Fri

Date : / /

Breadth First Search



~~Binary tree~~ → In-order, post-order, Pre-order

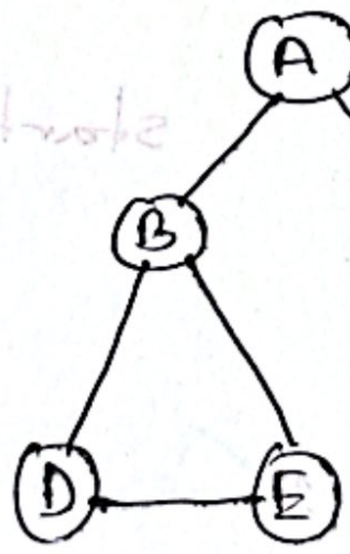
(General tree).

A, B, C, D, E, F
L0 L1 L2

A, C, B, E, F, D
L0 L1 L2

Level wise traversal.

(graph ও root node নেই)



(Graph)
↓
(source node)
↓
(starting node).

যেকোনো টাই source node হতে পারে।

Subject _____ Sat Sun Mon Tue Wed Thu Fri

Date : / /

Directed Graph এ যদি indegree zero হয় তাহলে সেই node টি কে আমরা source Node।

undirected Graph G যেকোনো valid node কে source node হতে পারে।

Graph Traversal :-

- Adj. list
- Adj matrix

} Representation

* Two traversal algorithm:-

① Breadth First Search (BFS)

② Depth First Search (DFS)

* Given a graph $G = (V, E)$ and a source vertex s, we want to visit all the vertices (v) exactly once।

$$G = (V, E)$$

Graph vertices Edges

Subject _____ Sat Sun Mon Tue Wed Thu Fri

Date : / /

BFS ⟶ Queue use করি।

DFS ⟶ Stack use করি

BFS

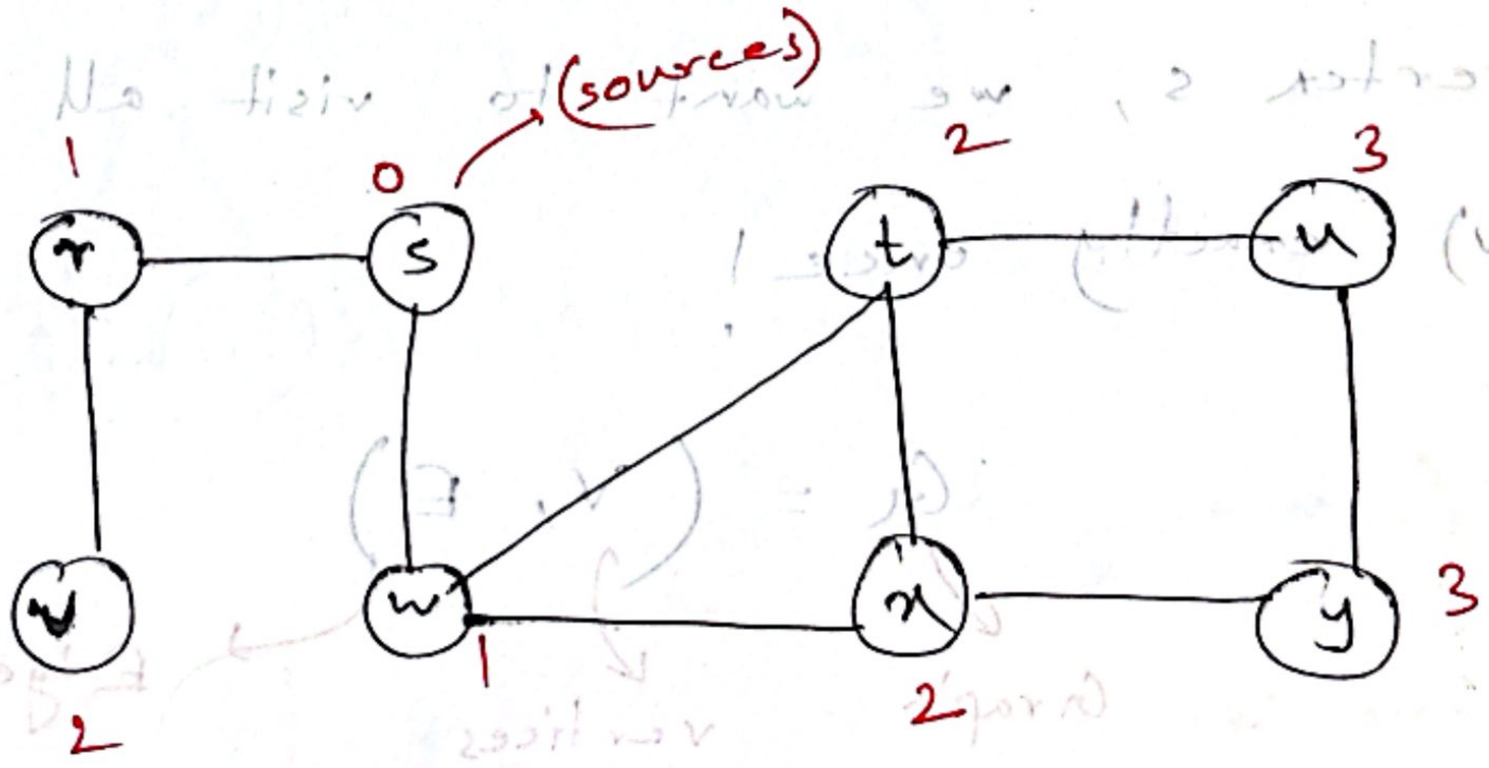
① Level wise traversal

② source থেকে শুরু হয়। source যে কোনো হতে পারে।

③ Traversal sequence কখনোই same না, আলাদাও হতে পারে।

④ source change হলে number of level যাবতে যা বদলাতে পারে।

⑤ Queue use করি।



$\underbrace{s}_{L0}, \underbrace{r, w}_{1}, \underbrace{v, x, t}_{2}, \underbrace{y, u}_{3}$

Subject _____ Sat Sun Mon Tue Wed Thu Fri

Date : / /

BFS – shortest path and shortest distance.

From source node to all other connected node.

BFS
- shortest path
- shortest distance

a unweight Graph ও BFS আমাকে shortest path দিতে পারে।

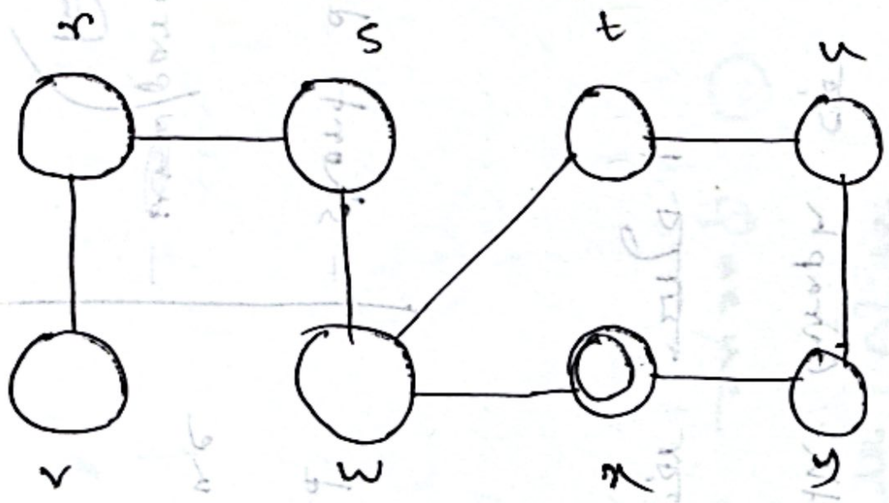
unweight Graph → (path value = 1).

Graph এর সবগুলো Path এর value যদি same হয় তবুও unweidth Graph এর BFS apply করে solve করা যাবে।

BFW works :-

- Unweighted
- same value

- short distance
- (parent দিয়ে)



BFS Pseudocode :-

```
BFS(G, s)
Q = [ ], visited = [ ], parent = [ ], distance[ ]
visited[s] = 1, parents[s] = -1, distance[s] = 0;
ENQueue (Q, s)
while Q is not empty
    u = DeQueue (Q) || DeQueue seq. is the DFS seq.
    for each v ∈ Adj[u];
        if visited [v] = 0
            visited [v] = 1
            distance [v] = distance [u] + 1;
            parent [v] = u
            ENQUEUE (Q, v)
```

Subject _____ Sat Sun Mon Tue Wed Thu Fri

Date : / /

• visited

r	s	t	u	v	w	x	y
~~0~~ 1	~~0~~ 1	~~0~~ 1	~~0~~ 1	~~0~~ 1	~~0~~ 1	~~0~~ 1	~~0~~ 1

• distance :-

r	s	t	u	v	w	x	y
~~0~~ 1	~~0~~ 0	~~0~~	~~0~~ 3	~~0~~ 2	~~0~~ 1	~~0~~ 2	~~0~~ 3

• parent :-

r	s	t	u	v	w	x	y
~~N~~ s	~~N~~ -1	~~N~~ w	~~N~~ t	~~N~~ r	~~N~~ s	~~N~~ w	~~N~~ x

Queue :-

~~s~~
~~r~~
~~w~~
~~v~~
~~x~~
~~t~~
~~y~~
~~x~~

u	Adj [u]	v
s	r, w	~~r~~ ~~w~~
r	s, v	~~s~~ ~~v~~
w	s, x, t	~~s~~ ~~x~~ ~~t~~
v	r	~~r~~
x	w, t, y	~~w~~ ~~t~~ ~~y~~
t	w, x, u	~~w~~ ~~x~~ ~~u~~

Subject _____ Sat Sun Mon Tue Wed Thu Fri

Date : / /

u	adj [u]	v
y	x, u	~~x~~
		~~u~~
u	t, y	~~t~~
		~~y~~

BFS: sequence:-

s, r, w, v, x, t, y, u

what is the shortest distance from s to u?

= 3 (check distance array).

what is the shortest path form s to u?

→ u ← t ← w ← s

(check parent array)

BFS :- shorted path and min distance from source vertices to all other vertices in an unweighted graph.

can always find the solution

In case of directed graph, not all vertices may be explored.

Subject _____ Sat Sun Mon Tue Wed Thu Fri

Date : / /

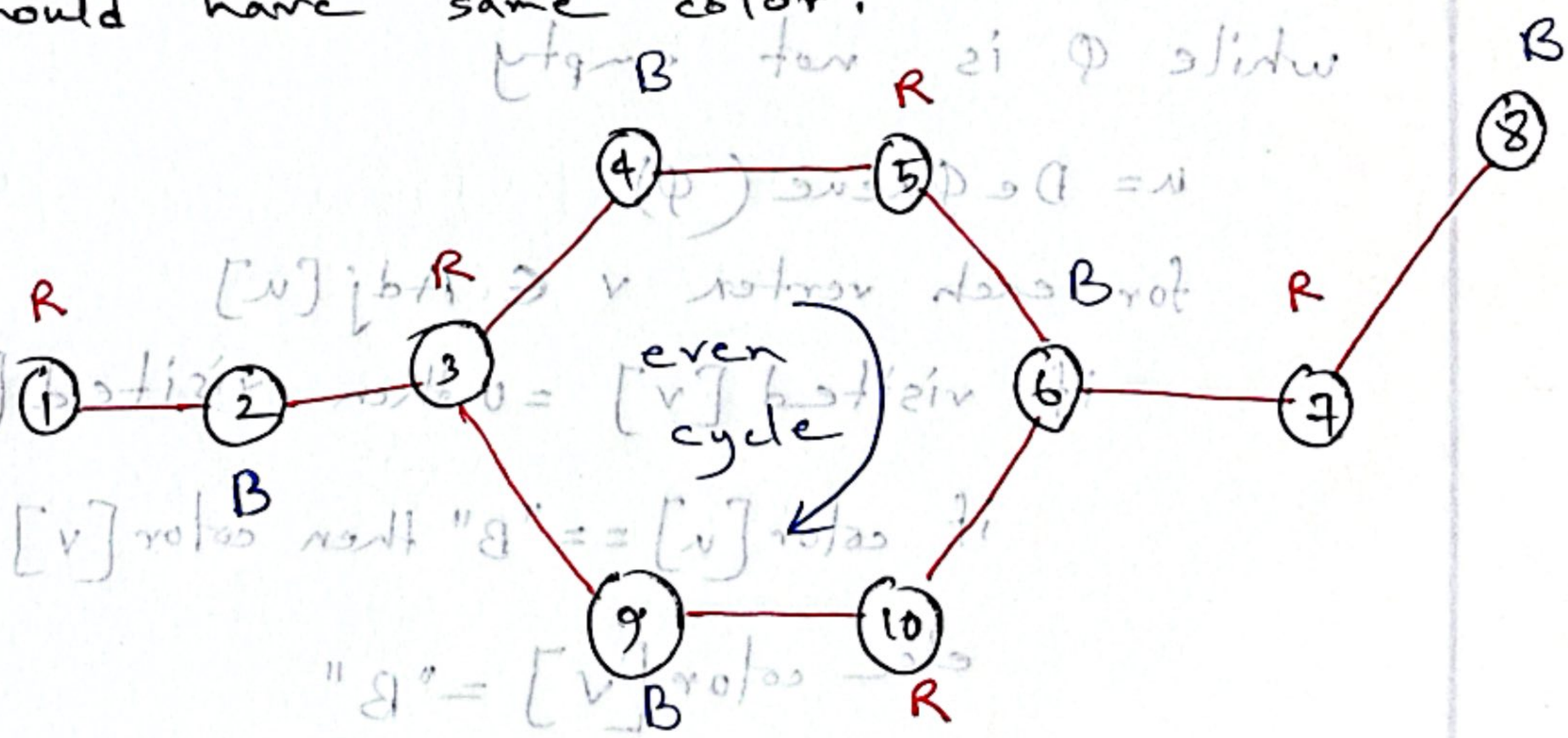
not suitable for detecting cycle in a graph.

Time Complexity : $O(n+m)$

space Complexity : $O(n+m)$.

Bipartite/ Bicolorable Graph:-

- A graph that can be colored using exactly two colors.
- However, no two color / no two adjacent vertices should have same color.



যদি পাশাপাশি 2টি color same হয়, তাহলে সেটি Bipartite Graph হবে না।

Subject _____ Sat Sun Mon Tue Wed Thu Fri

Date : / /

cycle exist করলে ⟶

- even cycle ⟶ Bipartite
- odd cycle ⟶ Bipartite নয়।

BFS use করে এই problem solve করা যায়।

```
Bipartite(G,S)
Q=[], visited=[], color=[]
visited[s]=1, color[s]="B"
Enqueue(Q,s)
while Q is not empty
    u= DeQueue(Q)
    for each vertex v ∈ Adj[u]
        if visited[v] =0 then visited[v]=1
            if color[u]=="B" then color[v]="G"
            else color[v]="B"
            ENQUEUE(Q,v)
        else if color[u]==color[v] return False
return True;
```