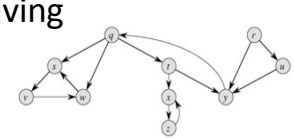
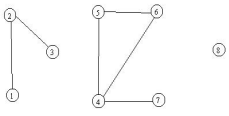
Graph Data Structures

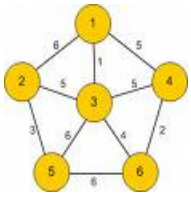
Atul Gupta

Graphs

- Most un-restricted form of data organization
- Final destination for problem solving
- Many variances
  - Directed and un-directed graphs
  - Connected and un-connected graphs
  - Weighted graphs



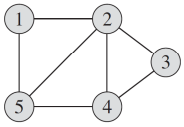




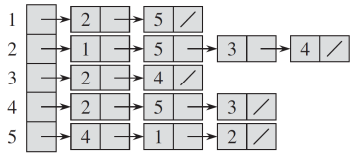
Representations of a Graph

Undirected Graph

- Adjacency List
- Adjacency Matrix



(a)



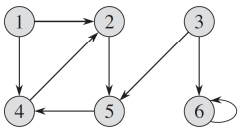
(b)

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

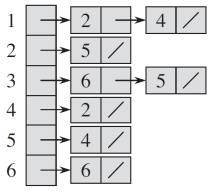
(c)

Representations (Di-Graph)

- Adjacency List
- Adjacency Matrix



(a)



(b)

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

(c)

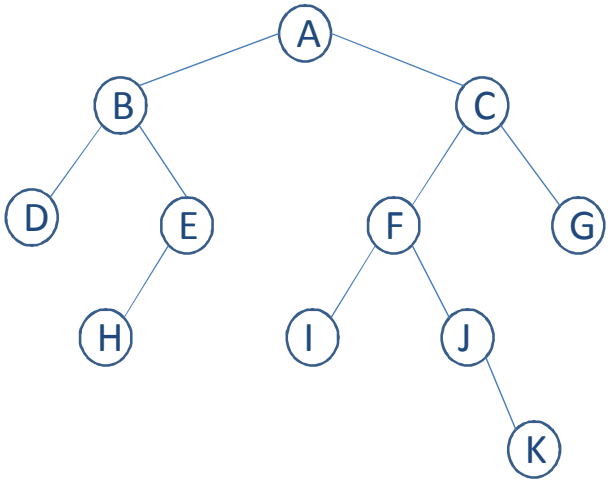
Graph Operations

- Insert
- Delete
- Search (Traversal)
  - Breadth-First Search
  - Depth-First Search

Tree Traversals

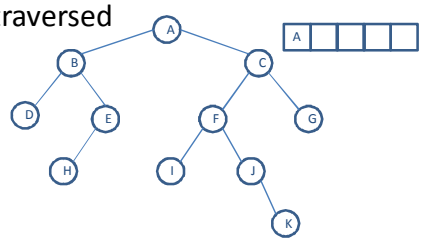
- Depth-First
  - In-order
  - Preorder
  - Post-order
- Breadth-First

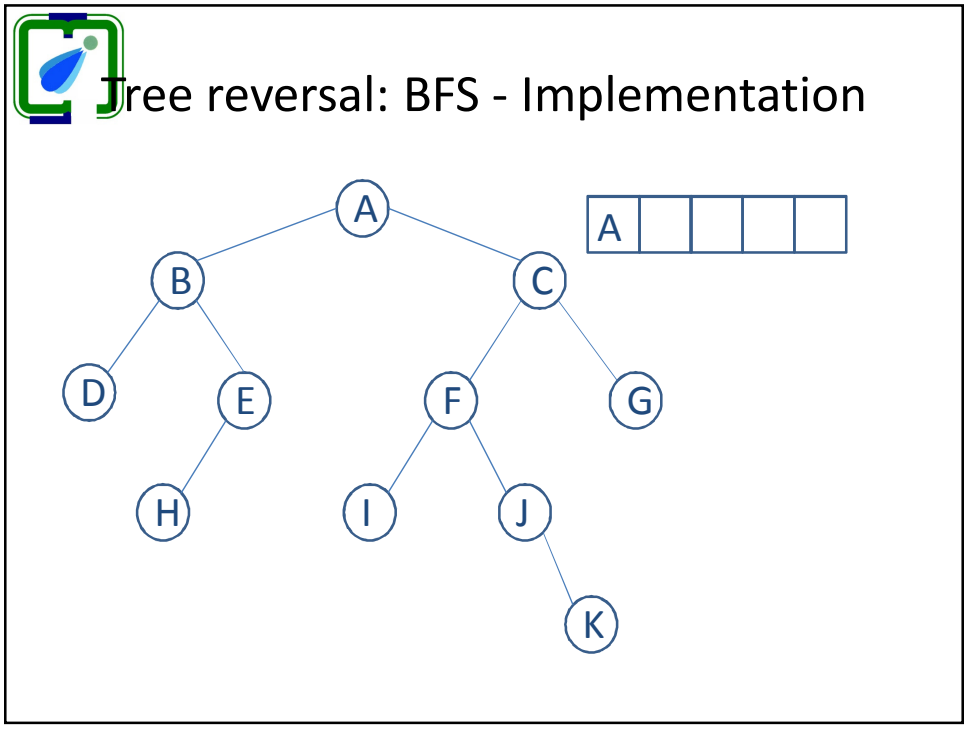
Tree reversal: BFS

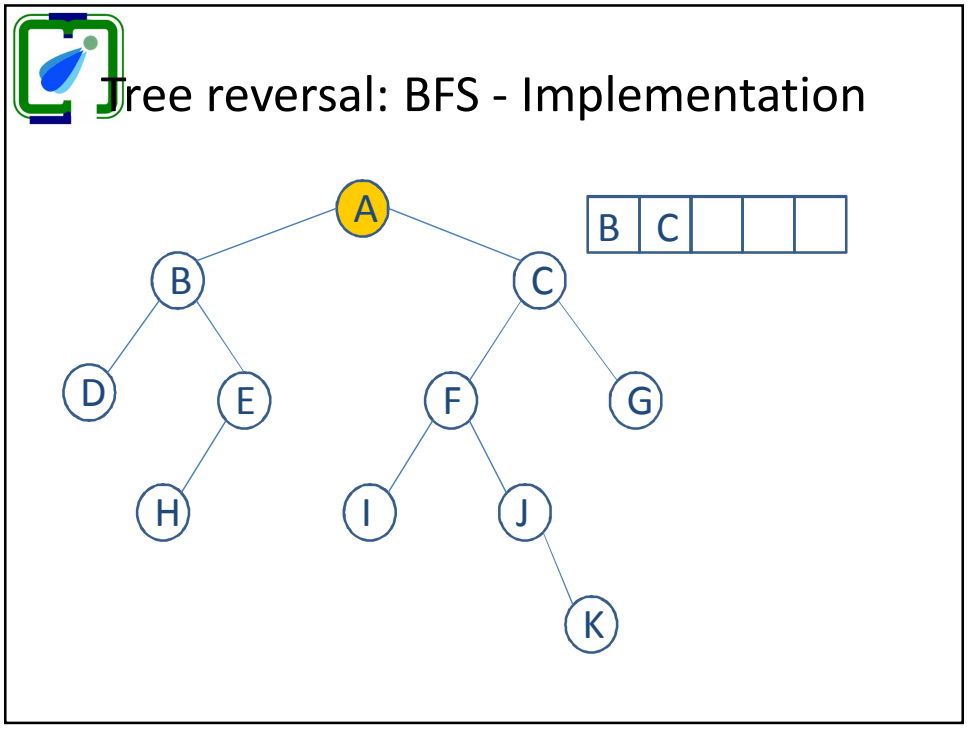


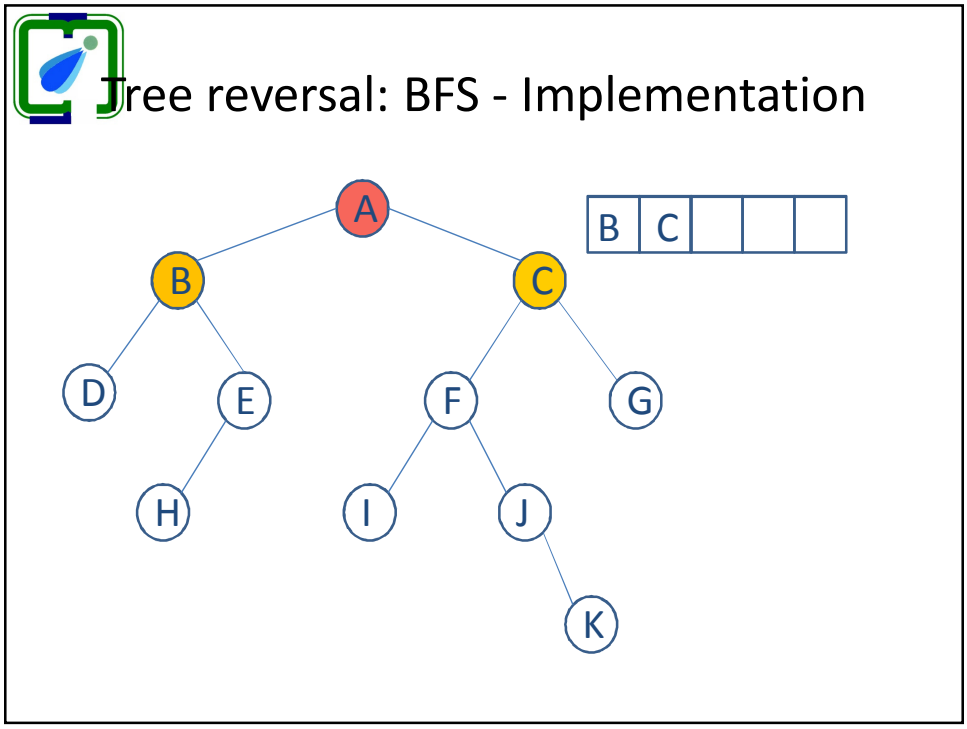
Tree reversal: BFS - Implementation

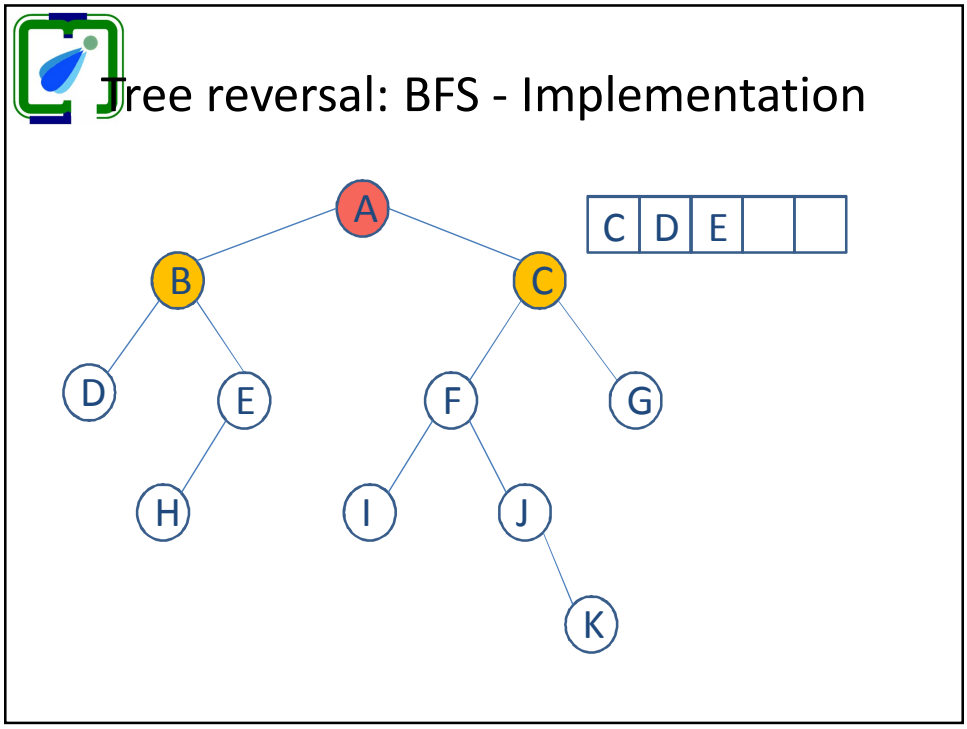
- Idea
  - Use of a queue data structure
  - A node is said to be traversed when its all successor nodes are generated, and queued
- Demo: Use of a coloring scheme
  - White: not accessed/encountered/generated
  - Yellow: encountered but not traversed
  - Red: traversed

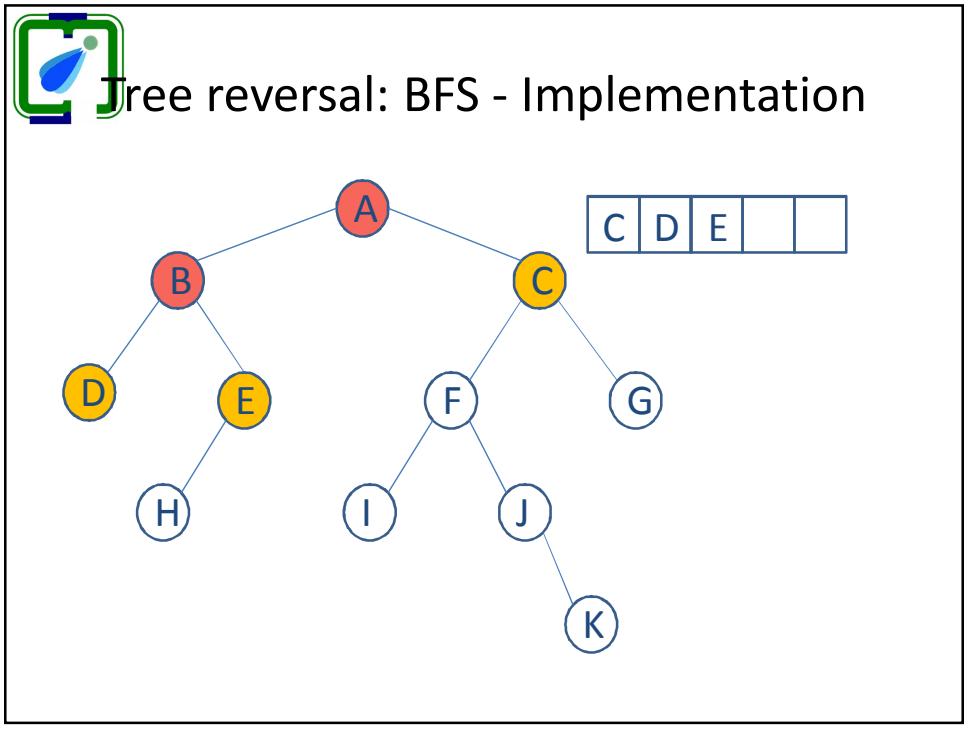


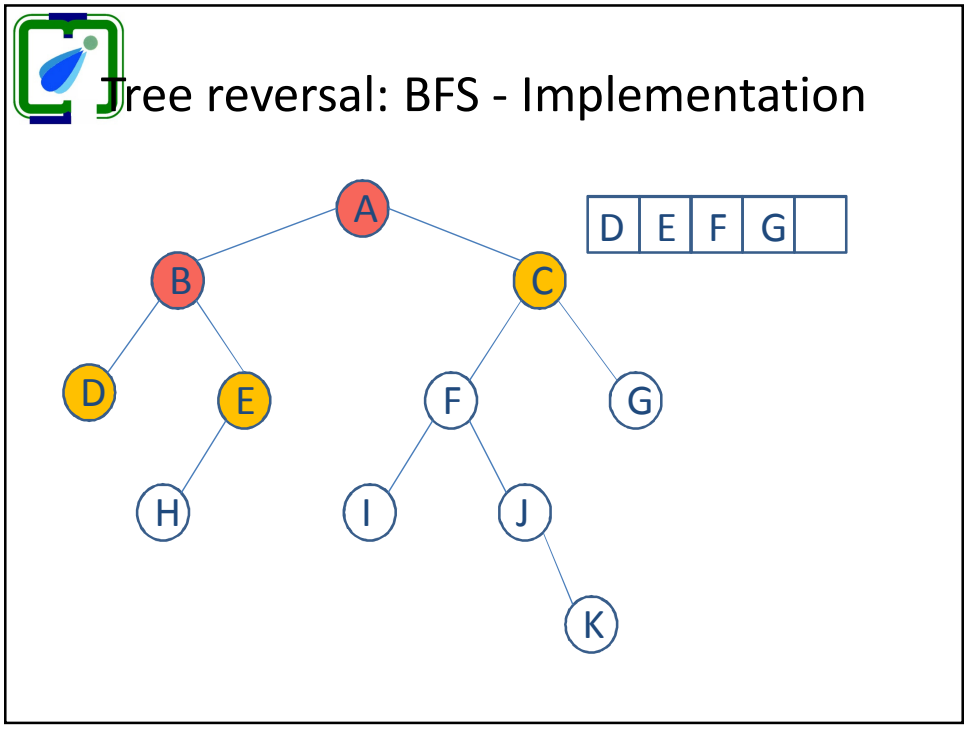

Tree reversal: BFS - Implementation
A
B
C
D
E
F
G
H
I
J
K
A

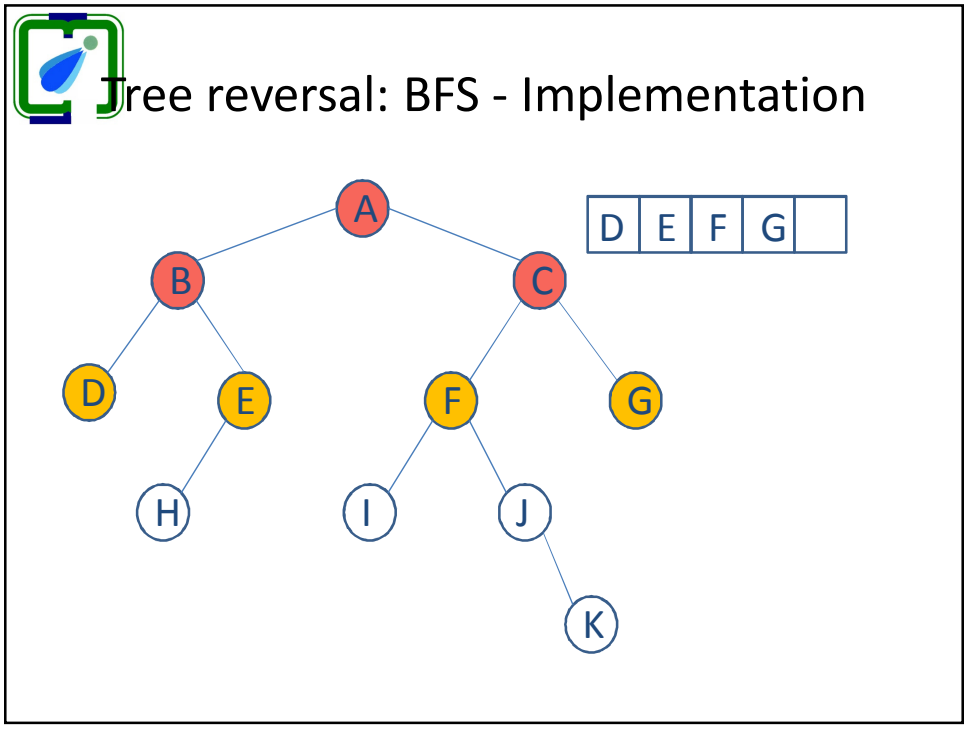

Tree reversal: BFS - Implementation
A
B
C
D
E
F
G
H
I
J
K
B C

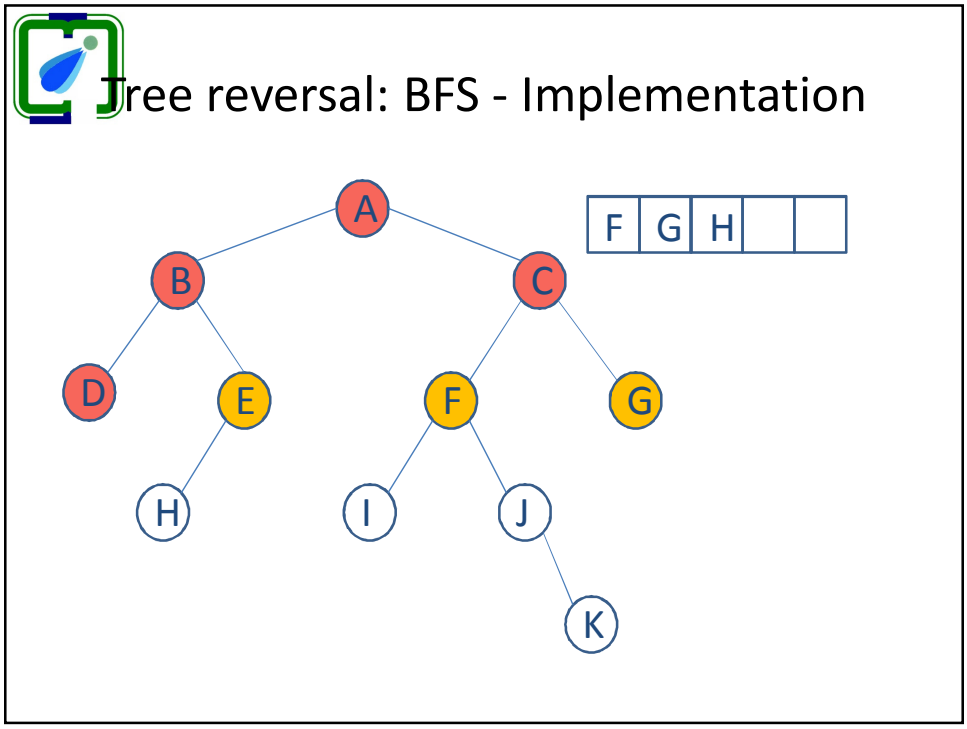

Tree reversal: BFS - Implementation
A
B
C
D
E
F
G
H
I
J
K
B C

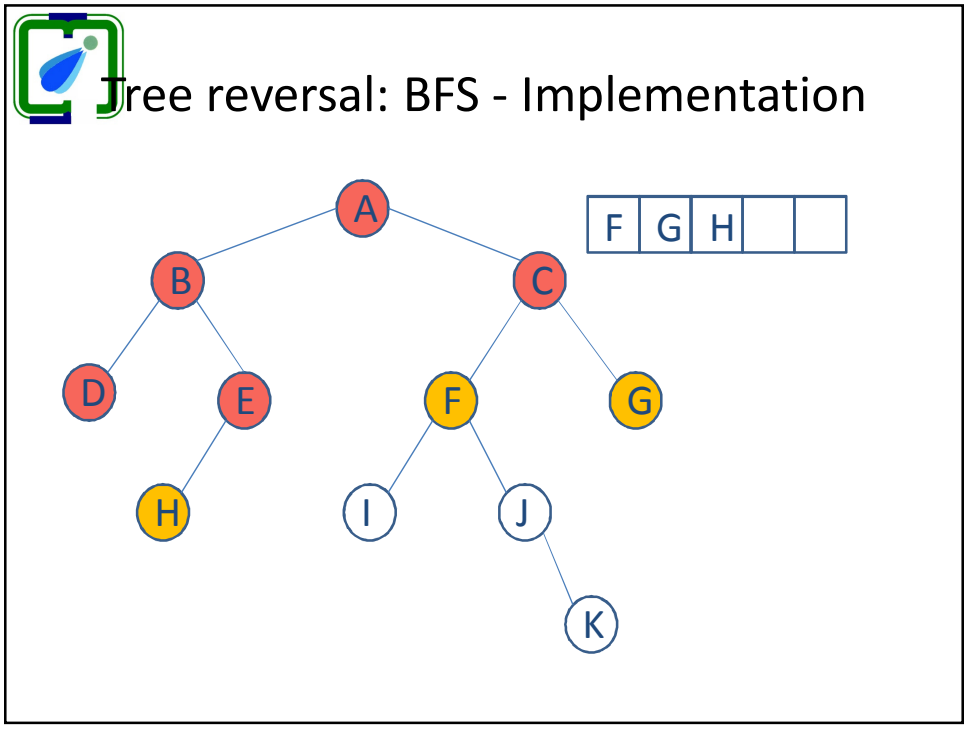

Tree reversal: BFS - Implementation
A
B
C
D
E
F
G
H
I
J
K
C D E

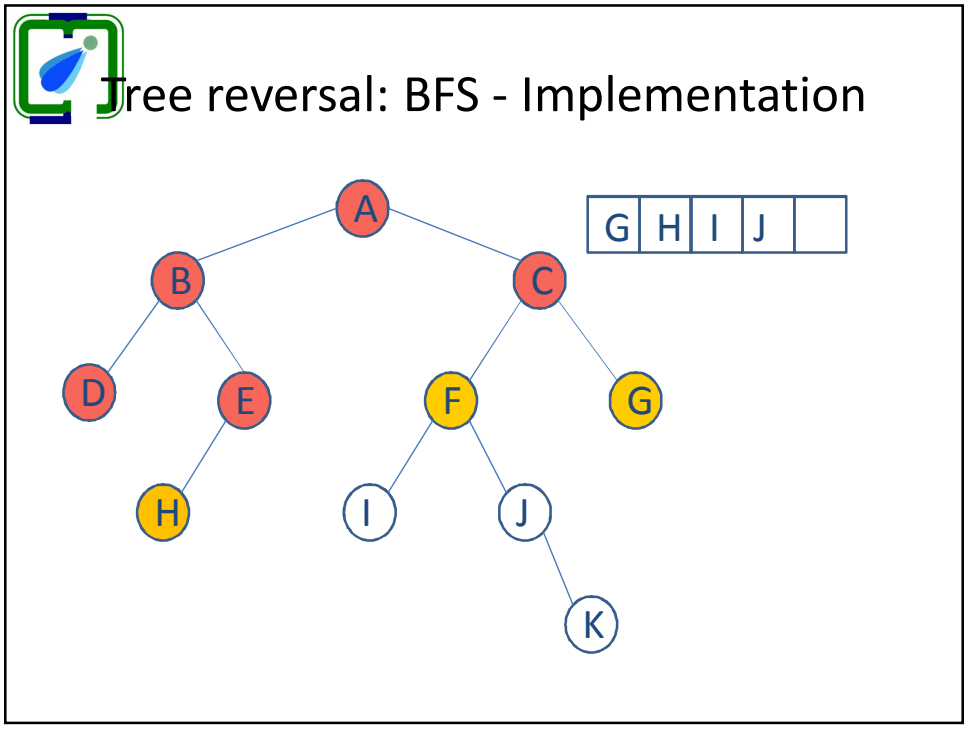

Tree reversal: BFS - Implementation
A
B
C
D
E
F
G
H
I
J
K
C D E

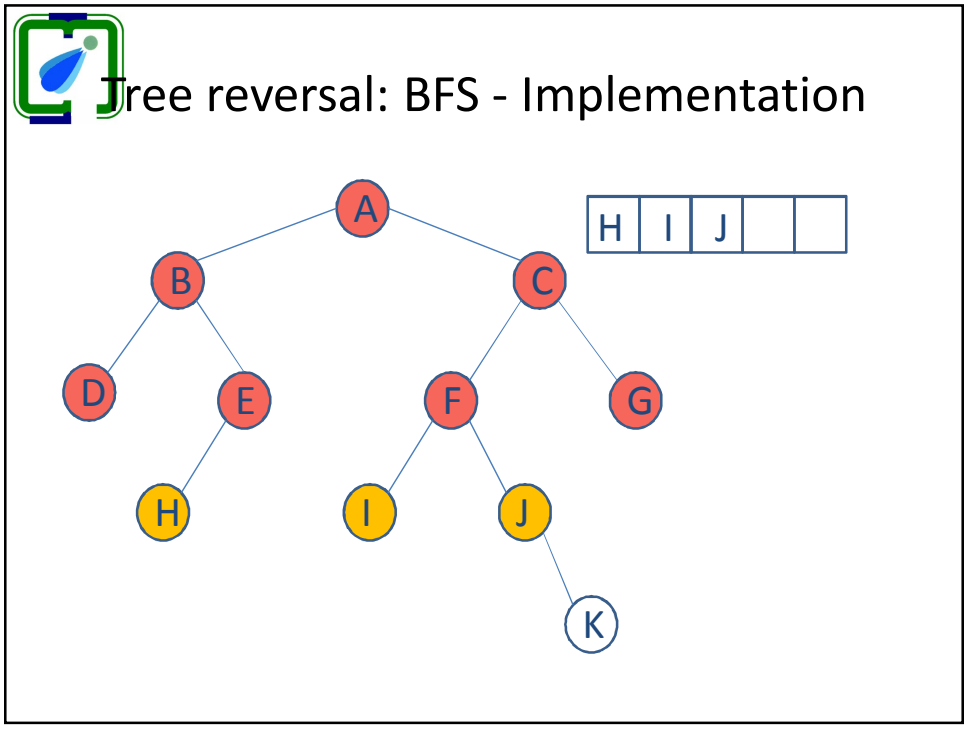

Tree reversal: BFS - Implementation
A
B
C
D
E
F
G
H
I
J
K
D E F G

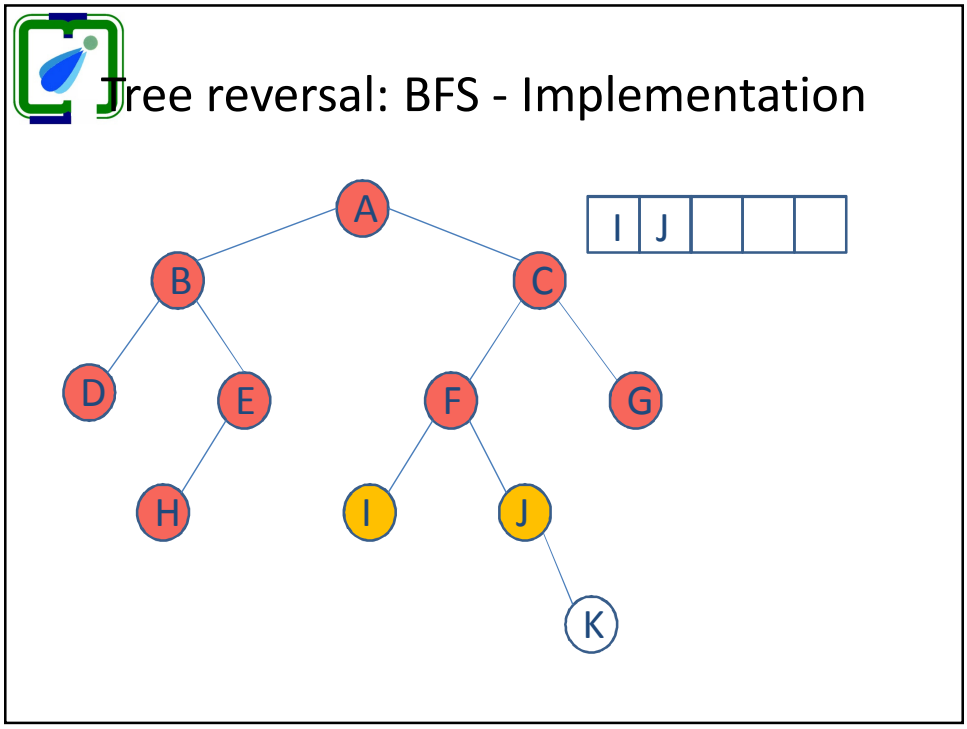

Tree reversal: BFS - Implementation
A
B
C
D
E
F
G
H
I
J
K
D E F G

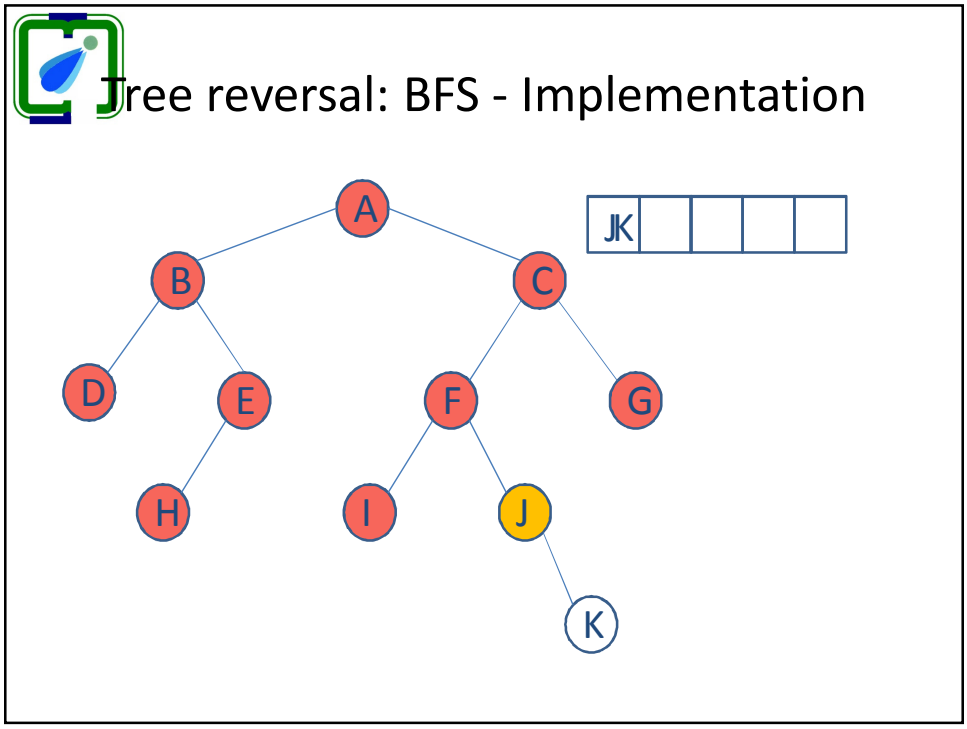

Tree reversal: BFS - Implementation
A
B
C
D
E
F
G
H
I
J
K
F G H

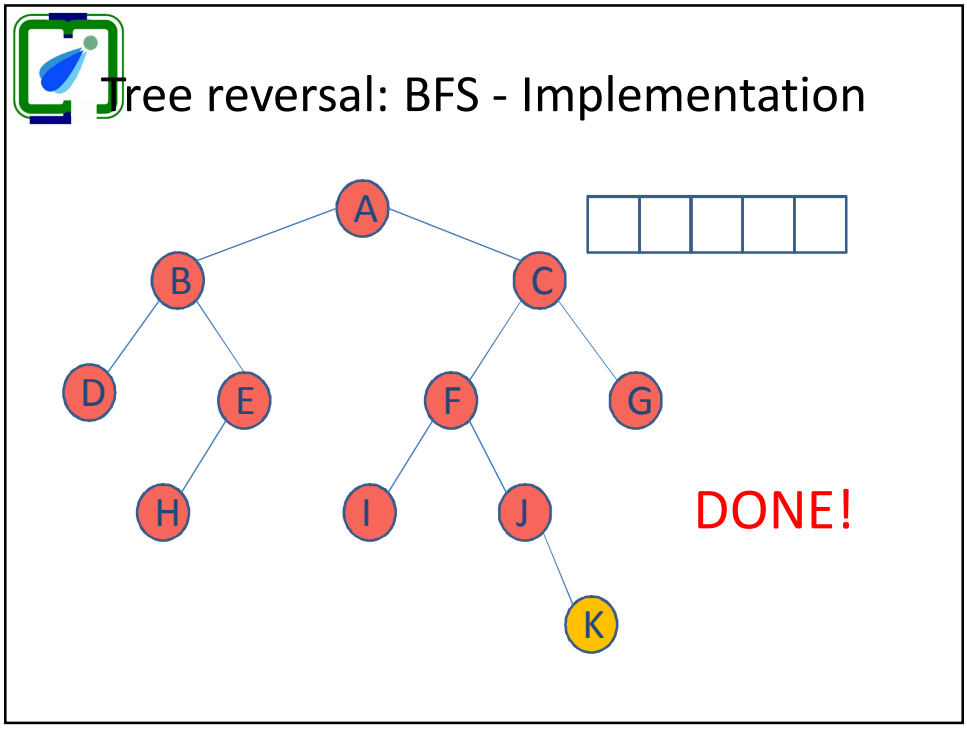

Tree reversal: BFS - Implementation
A
B
C
D
E
F
G
H
I
J
K
F G H

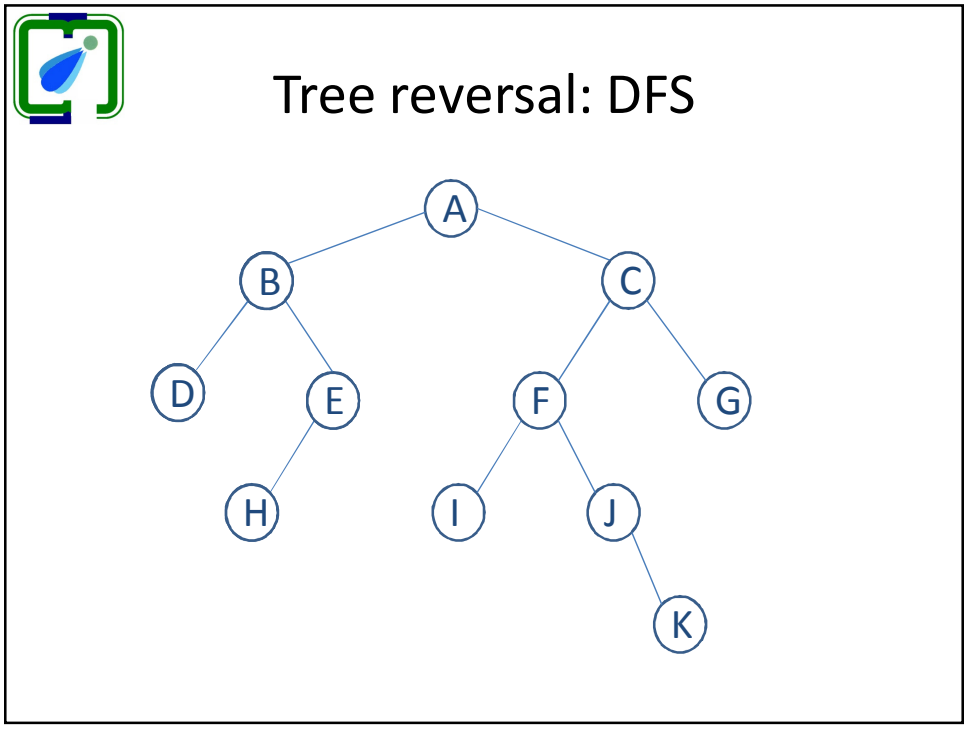

Tree reversal: BFS - Implementation
A
B
C
D
E
F
G
H
I
J
K
G H I J

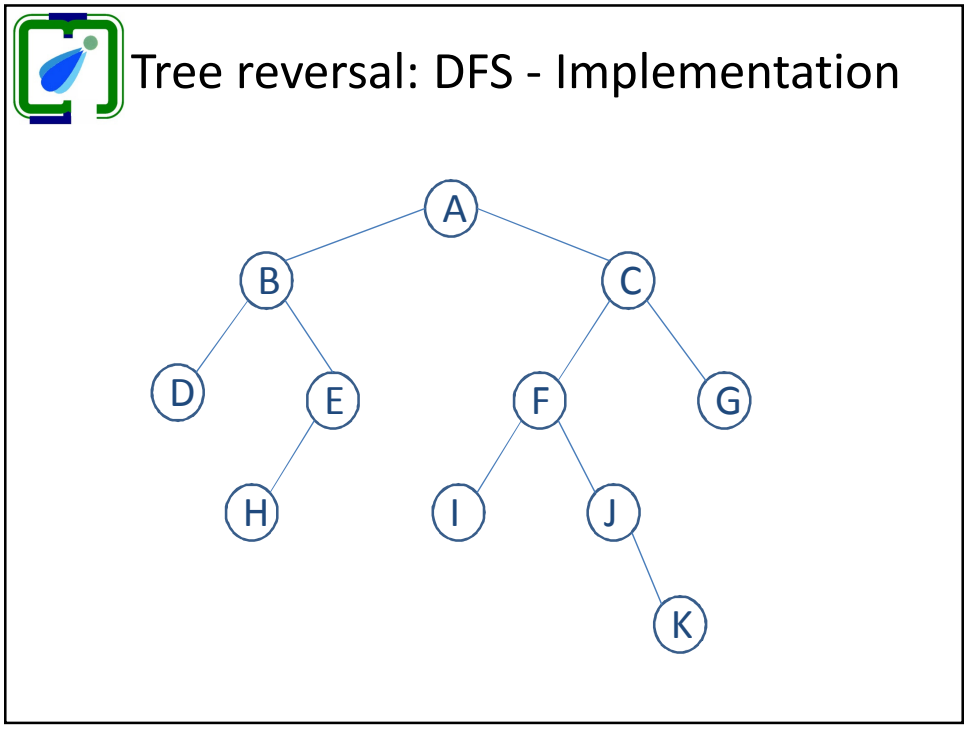

Tree reversal: BFS - Implementation
A
B
C
D
E
F
G
H
I
J
K
H I J

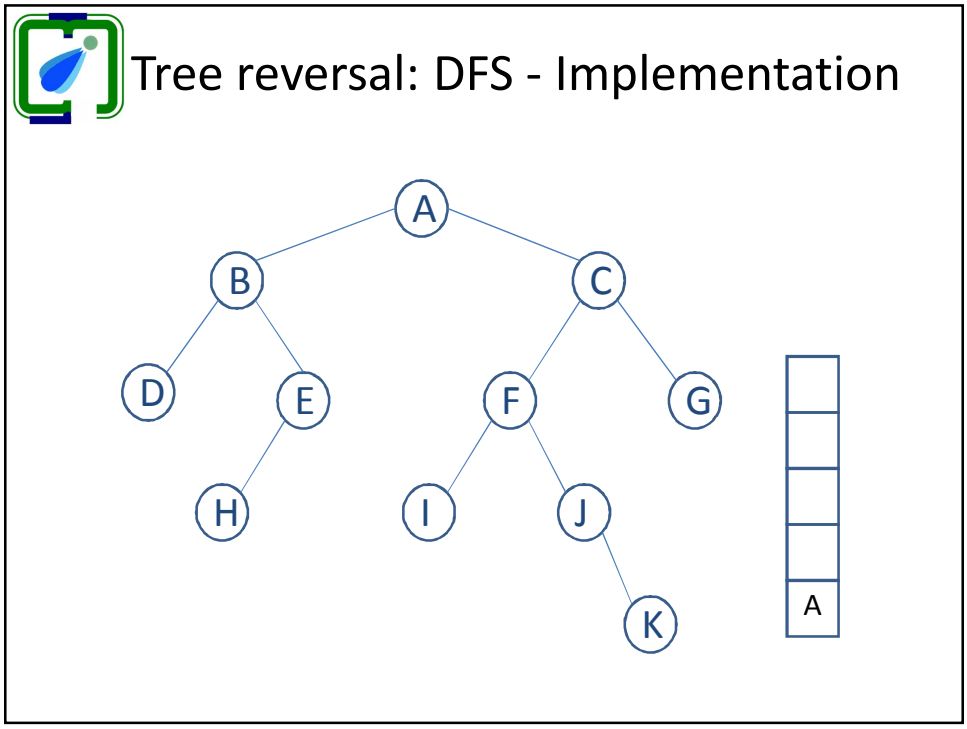

Tree reversal: BFS - Implementation
A
B
C
D
E
F
G
H
I
J
K
I J

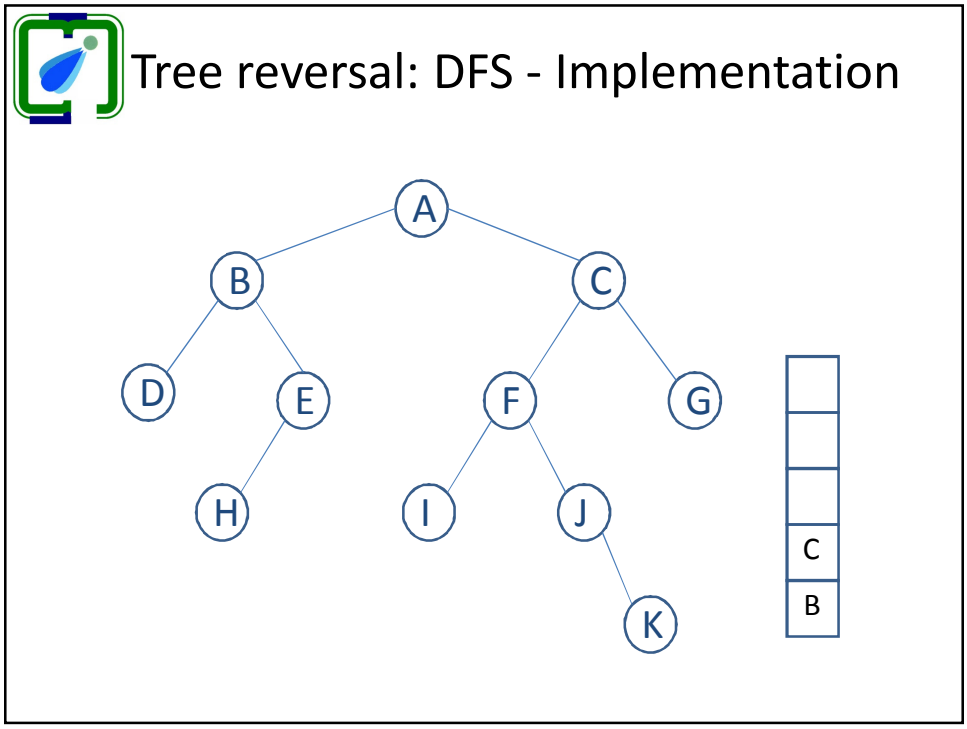

Tree reversal: BFS - Implementation
A
B
C
D
E
F
G
H
I
J
K
JK

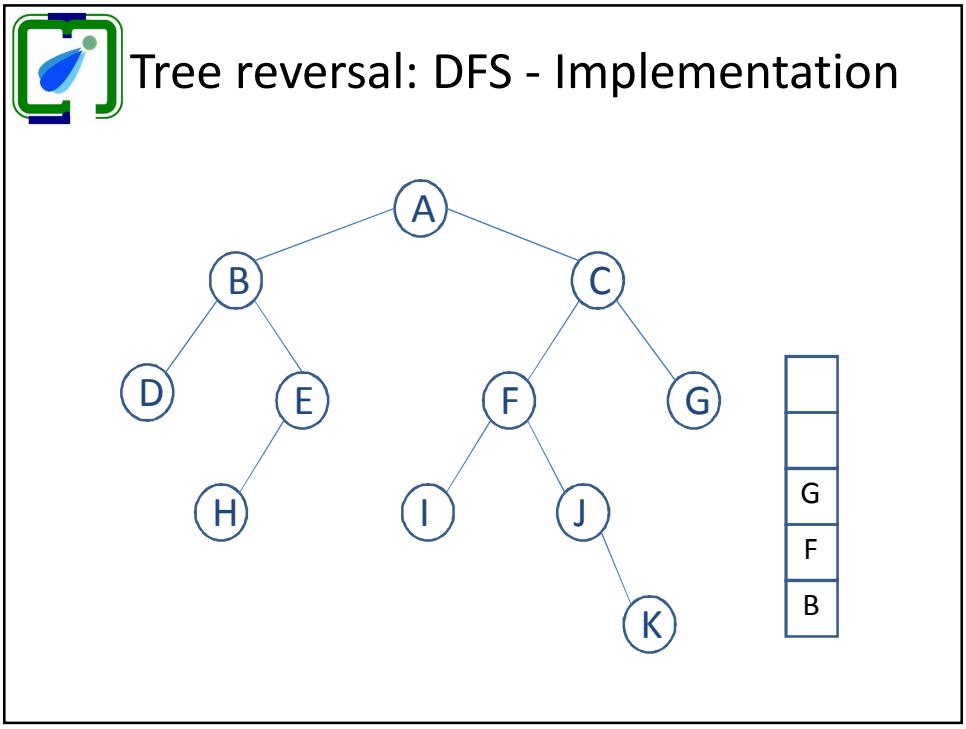

Tree reversal: BFS - Implementation
A
B
C
D
E
F
G
H
I
J
K
DONE!

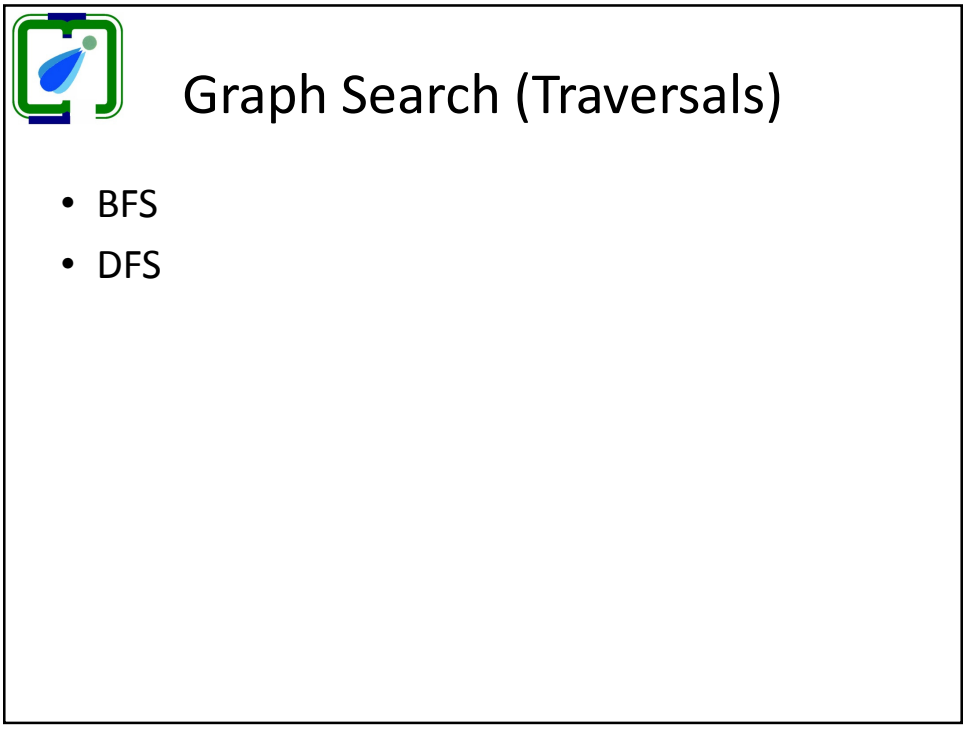

Tree reversal: DFS
A
B
C
D
E
F
G
H
I
J
K


Tree reversal: DFS - Implementation
A
B
C
D
E
F
G
H
I
J
K


Tree reversal: DFS - Implementation
A
B
C
D
E
F
G
H
I
J
K
A


Tree reversal: DFS - Implementation
A
B
C
D
E
F
G
H
I
J
K
C
B

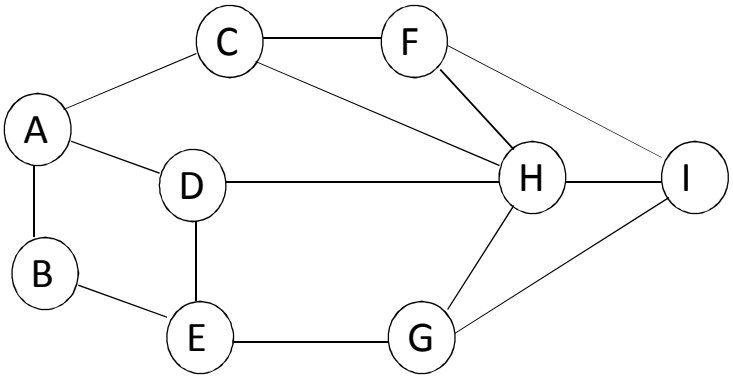

Tree reversal: DFS - Implementation
A
B
C
D
E
F
G
H
I
J
K
G
F
B


Graph Search (Traversals)
• BFS
• DFS

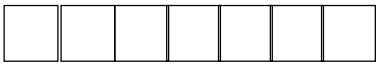
Breadth-First Search (Traversal)

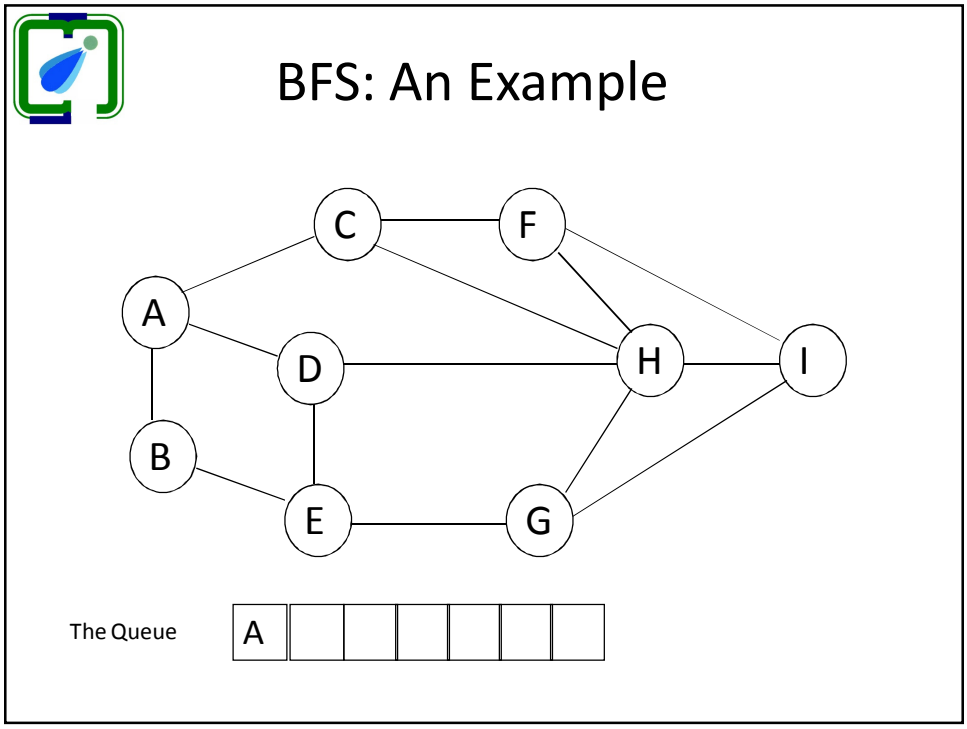
- Idea
  - Identify the starting node (this will be given to you)
  - Generate a traversal tree while traversing the graph
  - A node is said to be traversed when its all successor nodes are generated
- Implementation: Use of a coloring scheme
  - White: not accessed/encountered/generated
  - Yellow: encountered but not traversed
  - Red: traversed

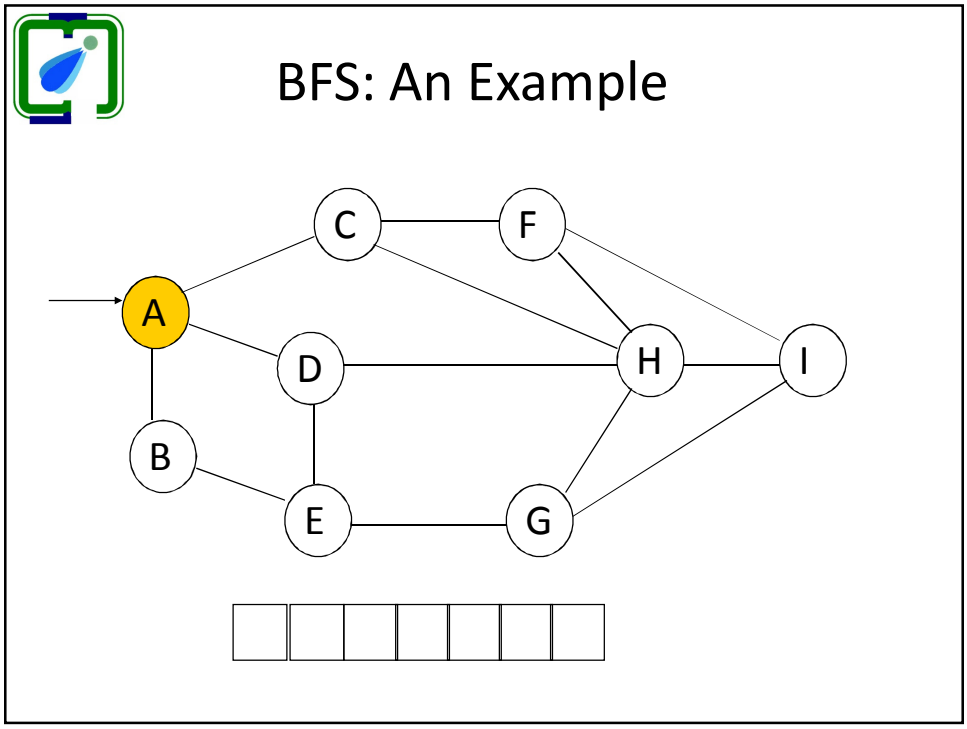
BFS: An Example

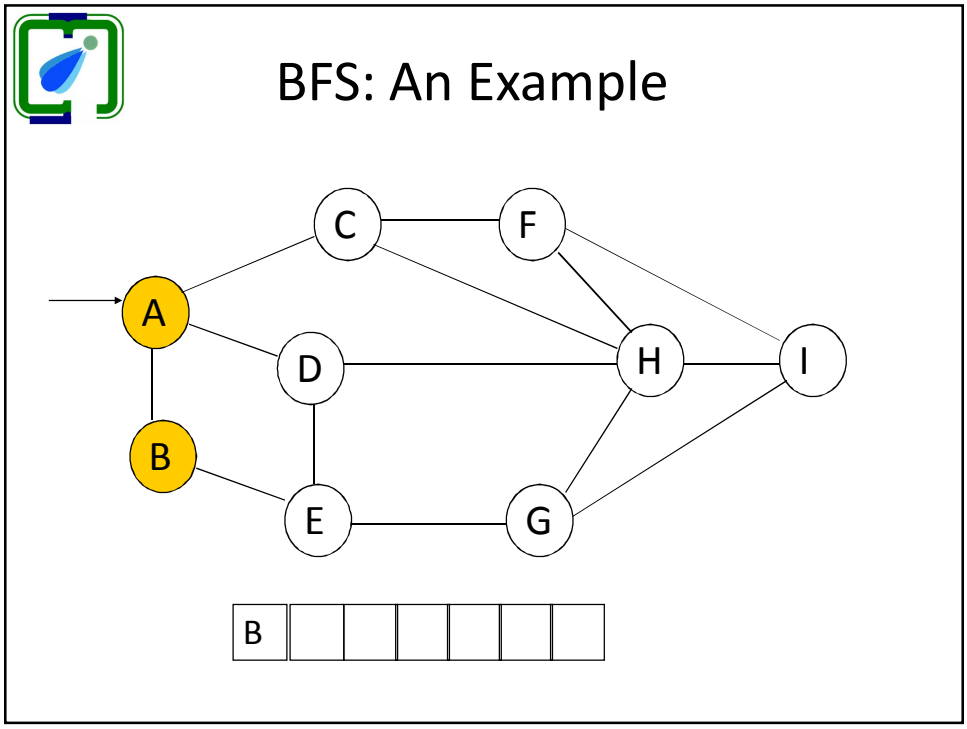


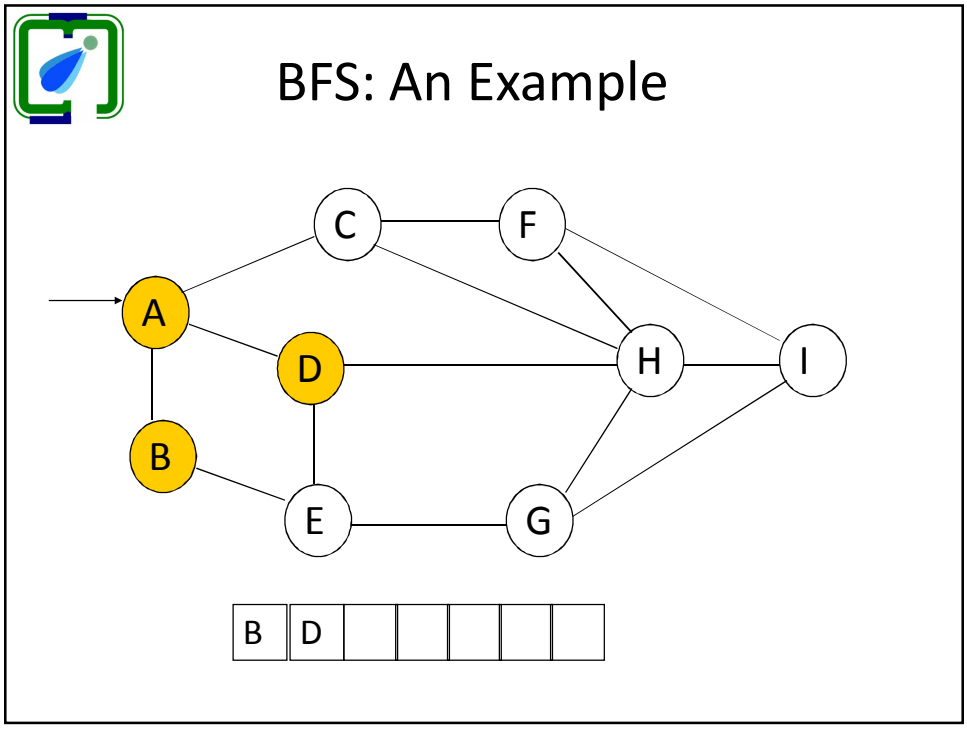
The Queue

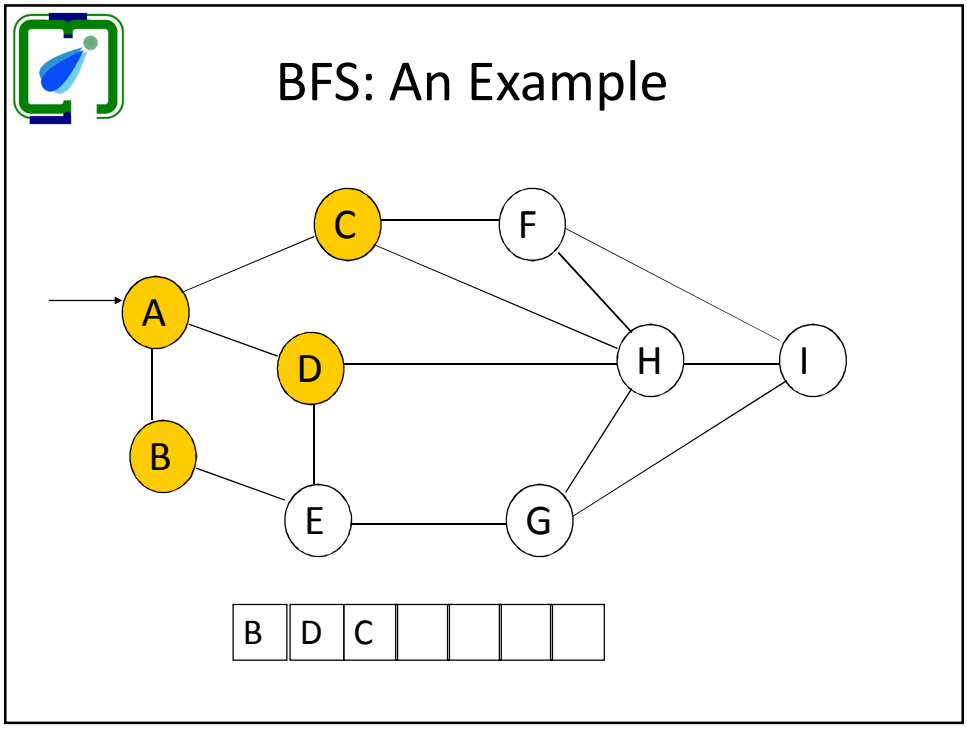


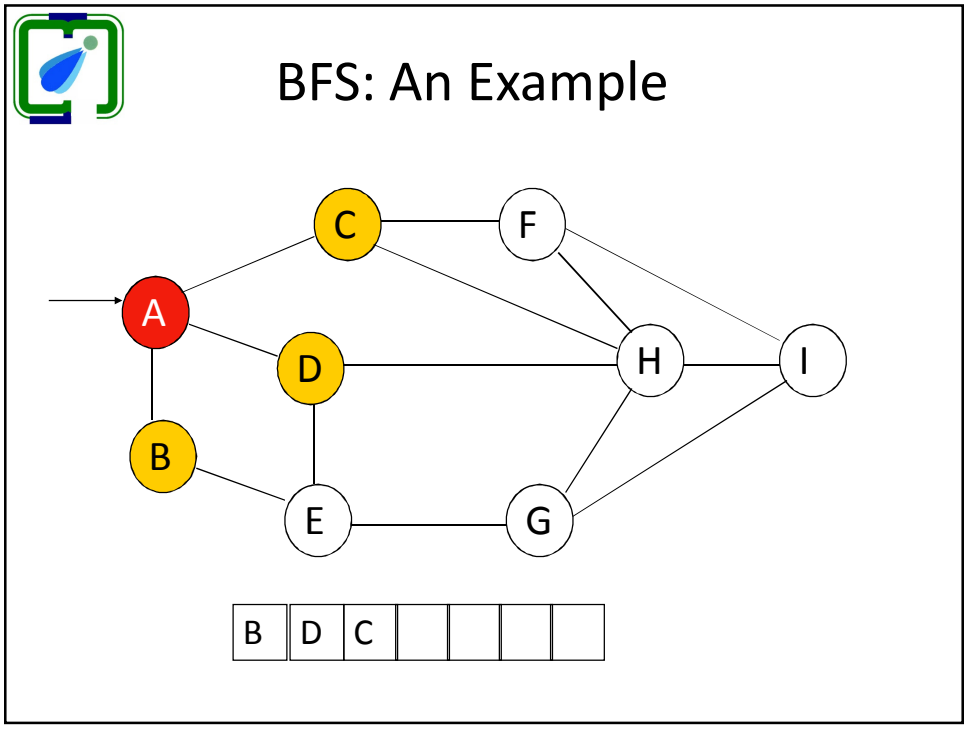

BFS: An Example
C
F
A
D
H
I
B
E
G
The Queue
A

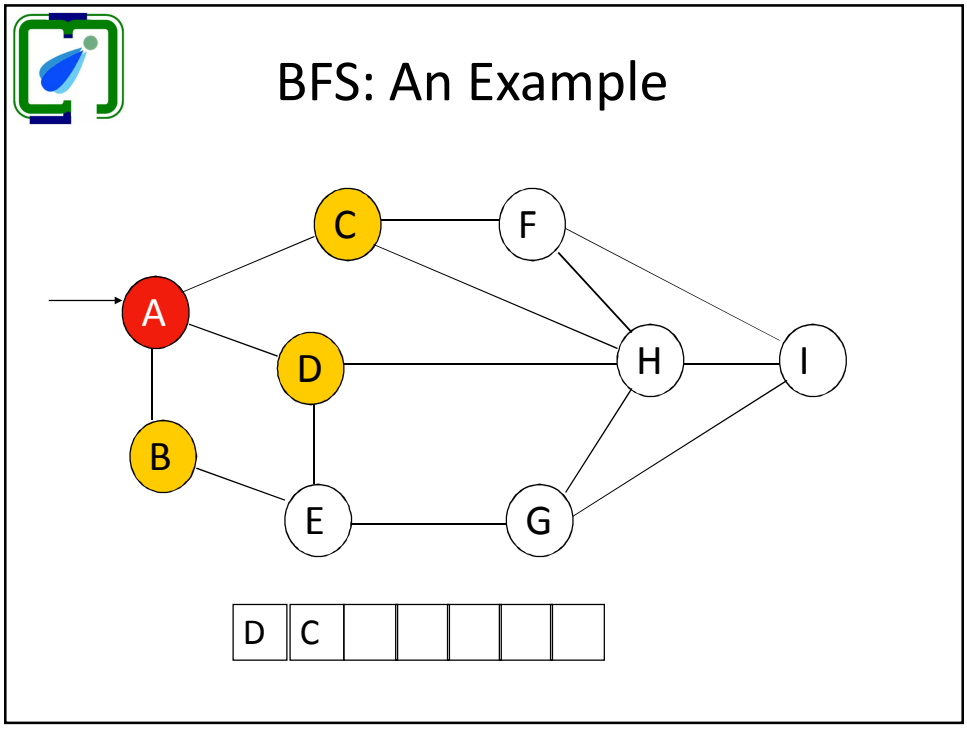

BFS: An Example
C
F
A
D
H
I
B
E
G

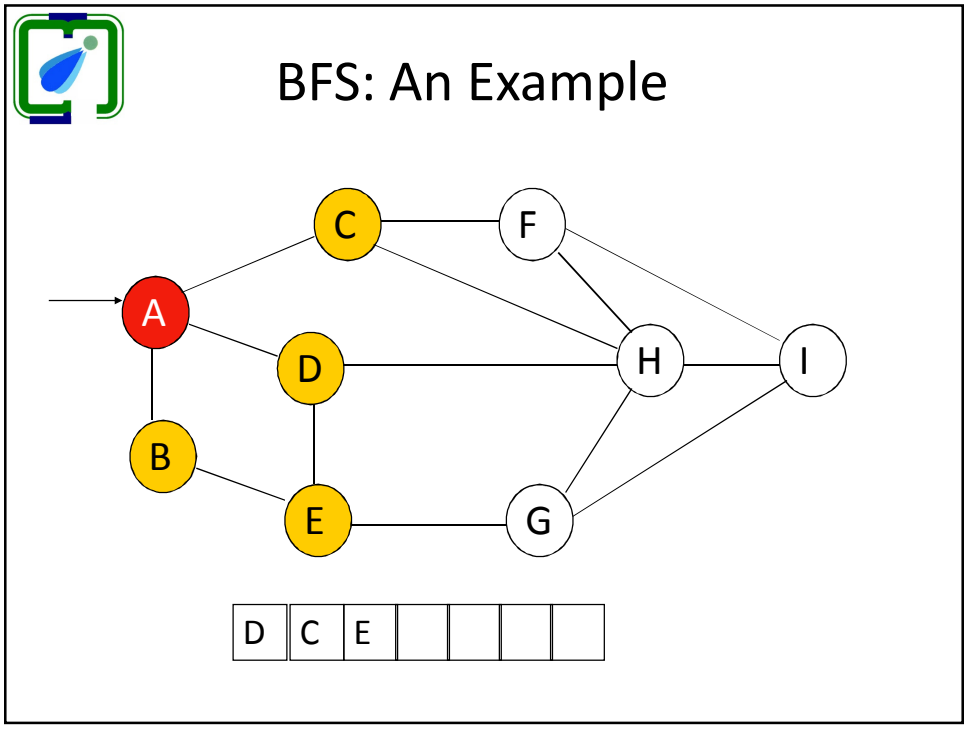

BFS: An Example
C
F
A
D
H
I
B
E
G
B

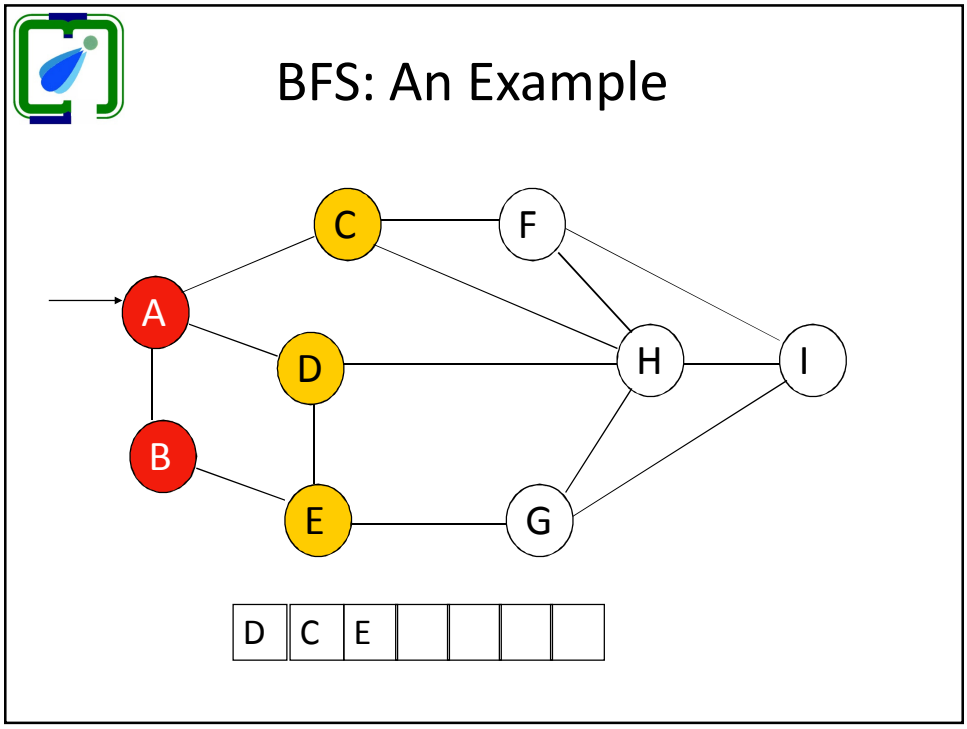

BFS: An Example
C
F
A
D
H
I
B
E
G
B
D

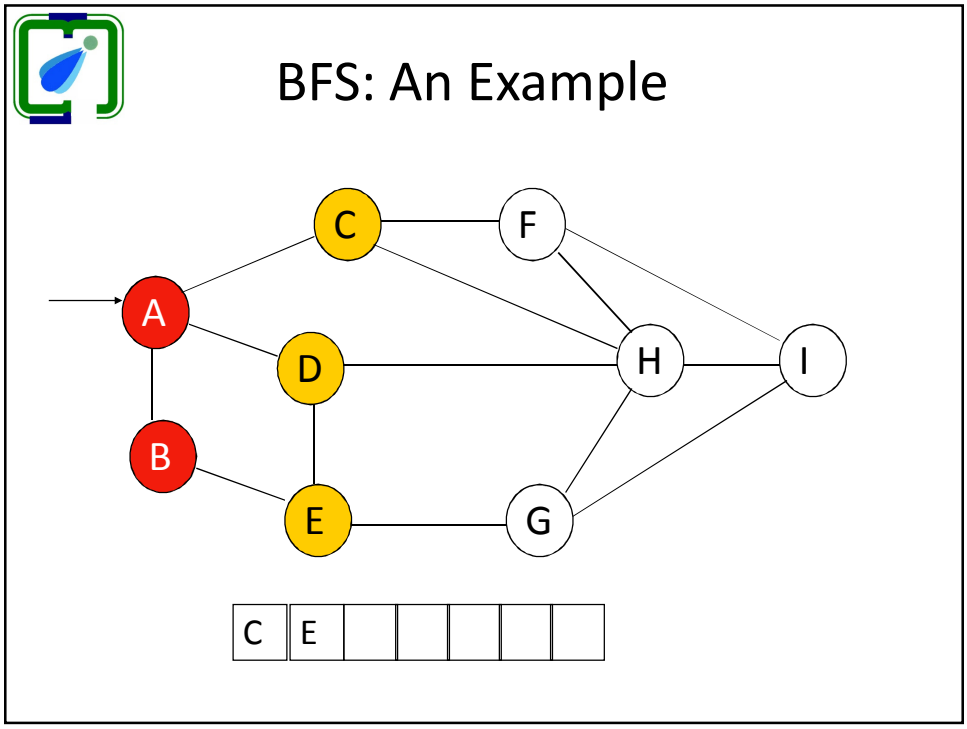

BFS: An Example
C
F
A
H
I
D
B
E
G
B
D
C

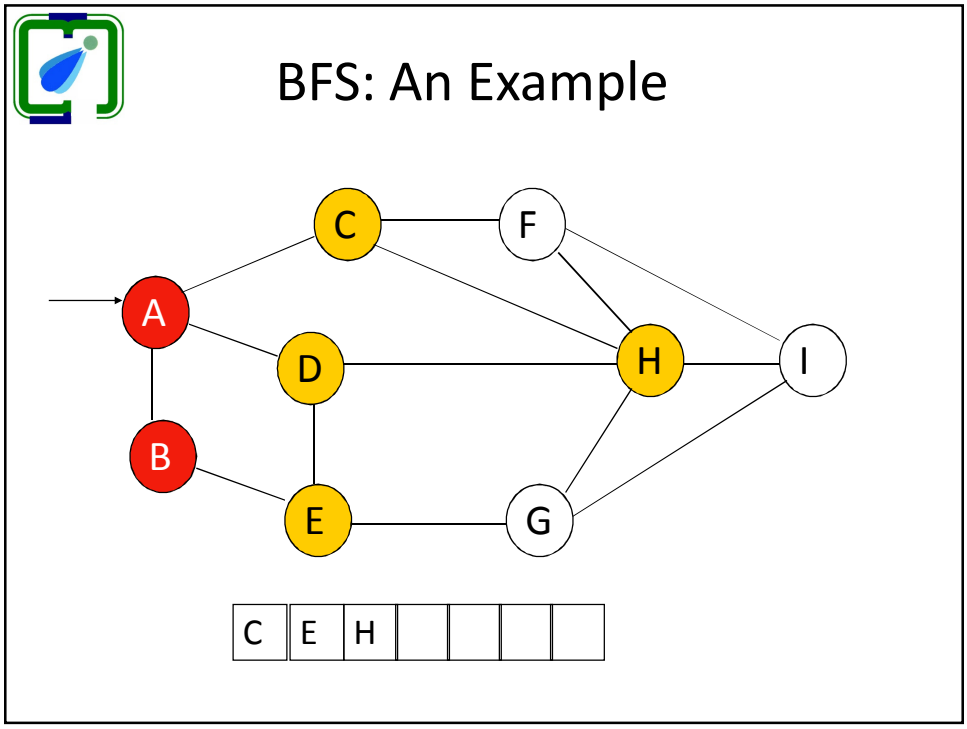

BFS: An Example
C
F
A
H
I
D
B
E
G
B
D
C

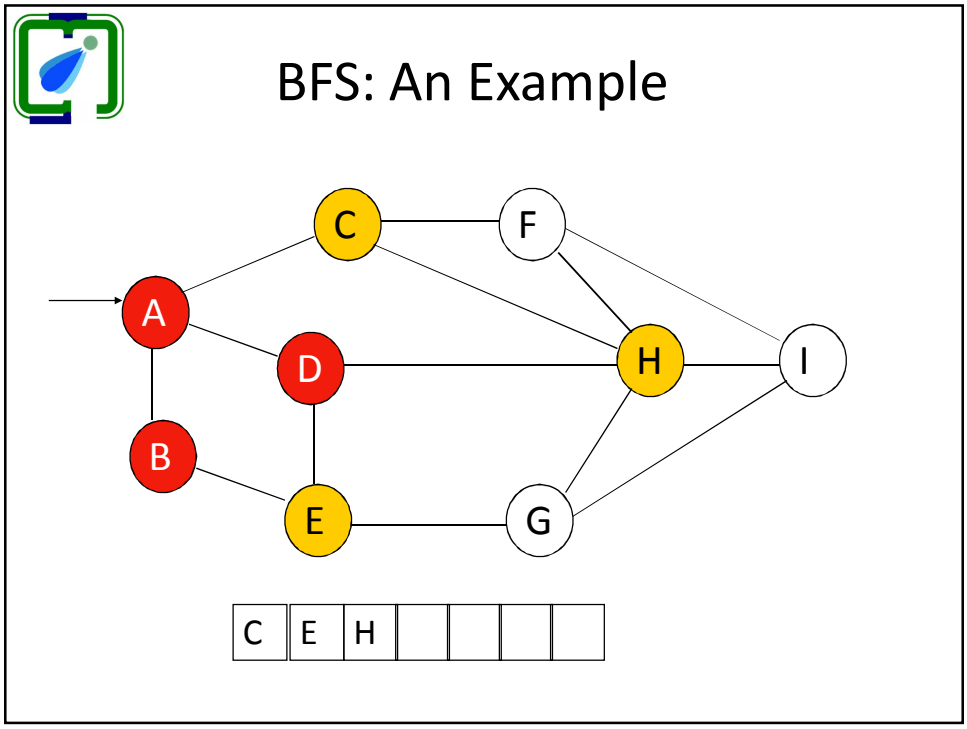

BFS: An Example
C
F
A
H
I
D
B
E
G
D
C

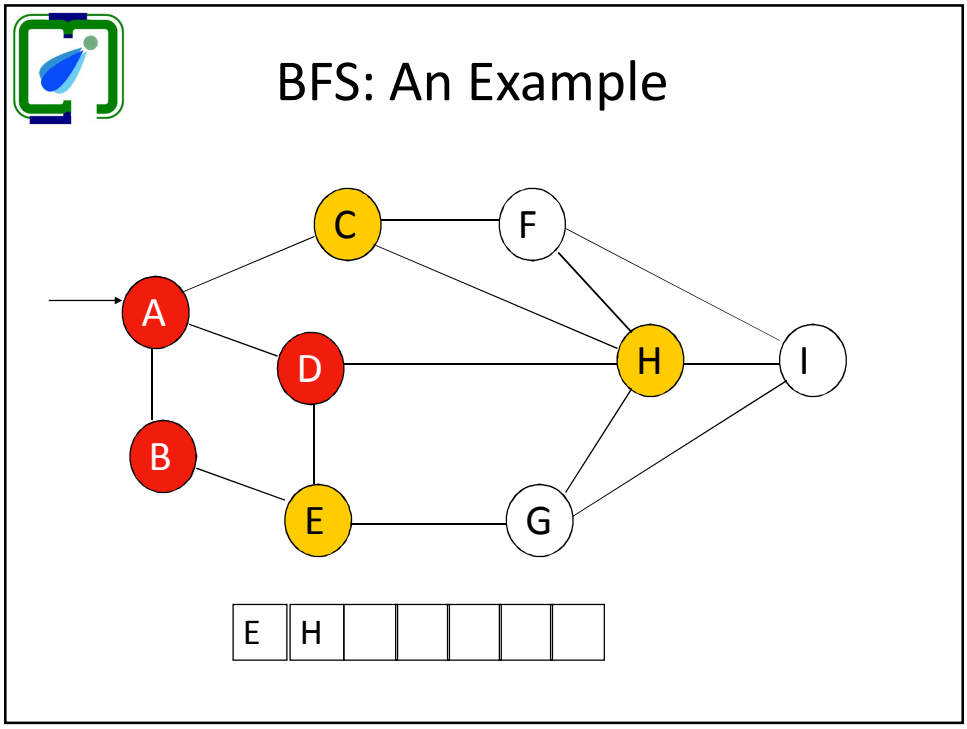

BFS: An Example
C
F
A
H
I
D
B
E
G
D
C
E

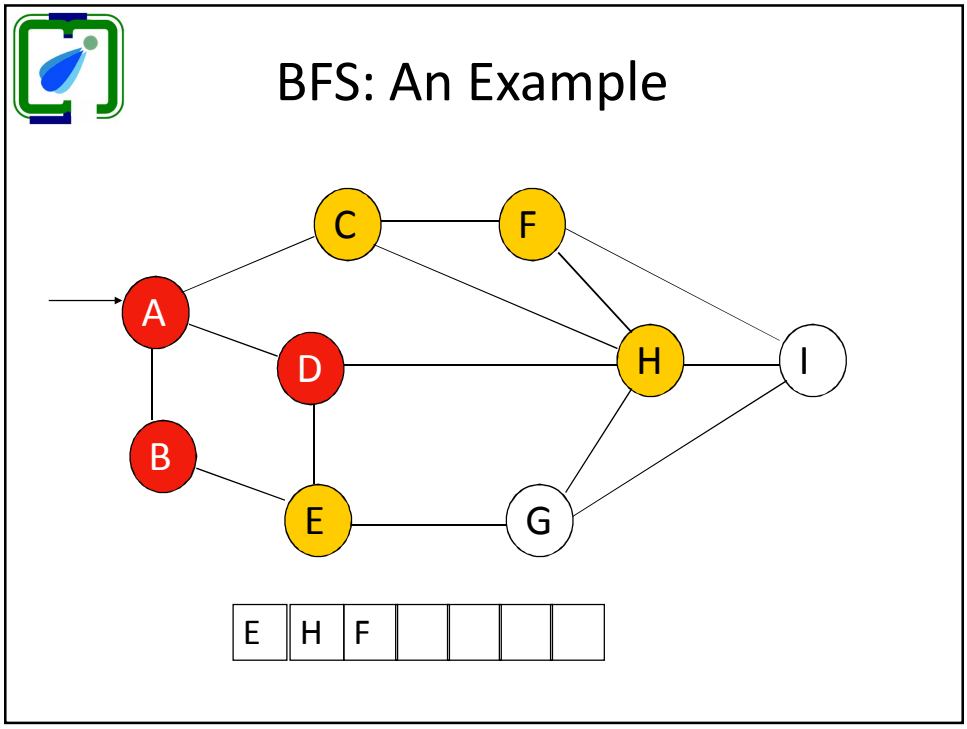

BFS: An Example
C
F
A
D
H
I
B
E
G
D
C
E

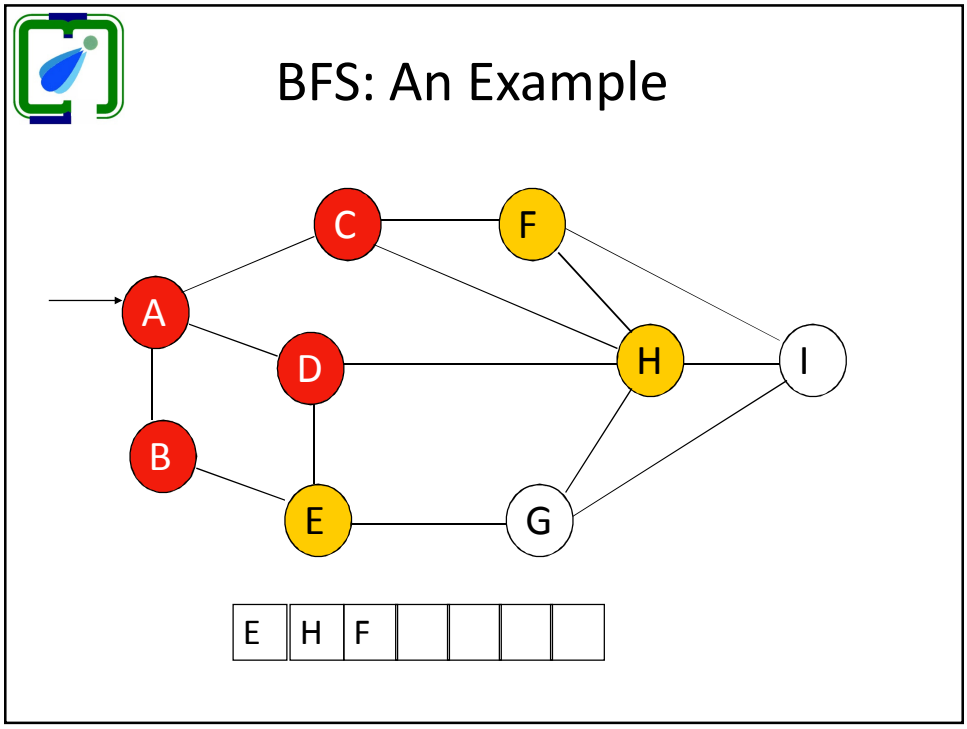

BFS: An Example
C
F
A
D
H
I
B
E
G
C
E

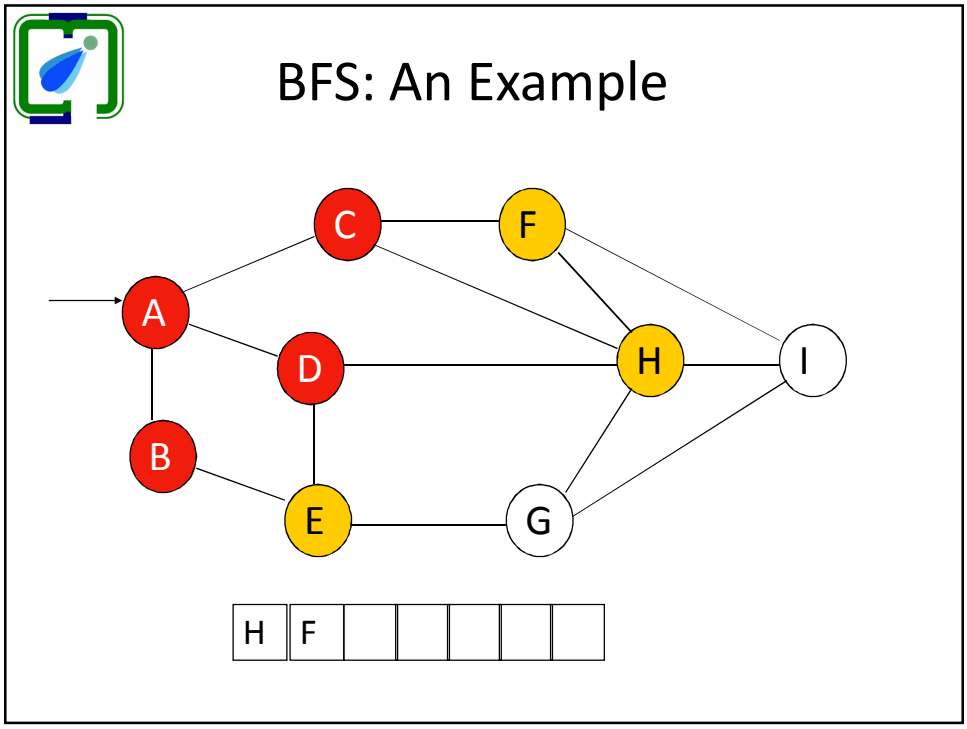

BFS: An Example
C
F
A
H
I
D
B
E
G
C
E
H

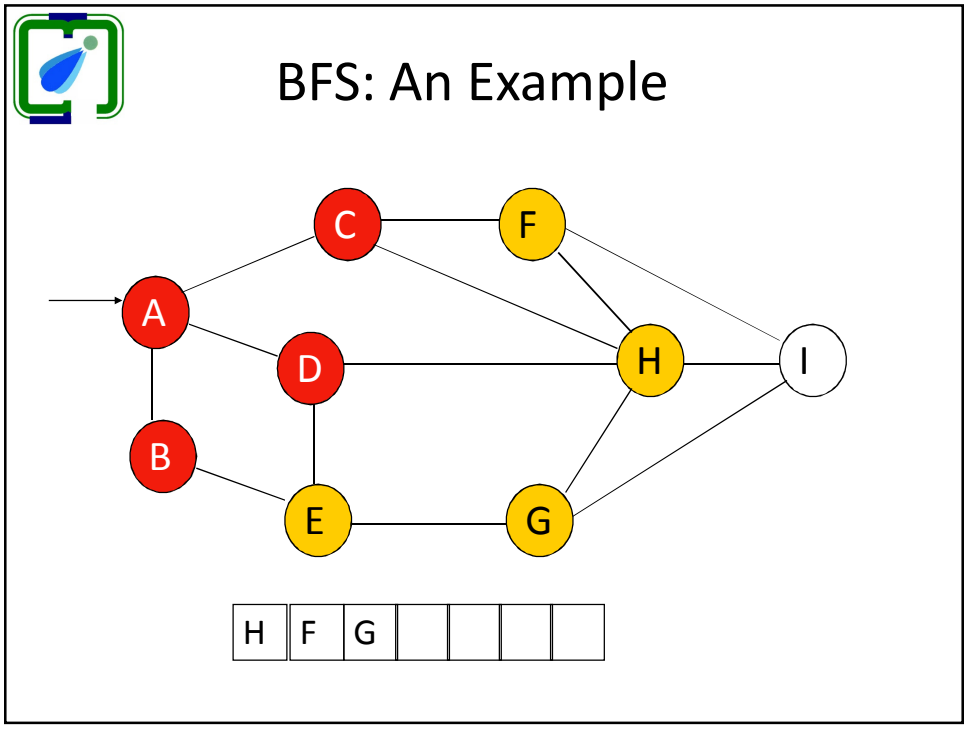

BFS: An Example
C
F
A
H
I
D
B
E
G
C
E
H

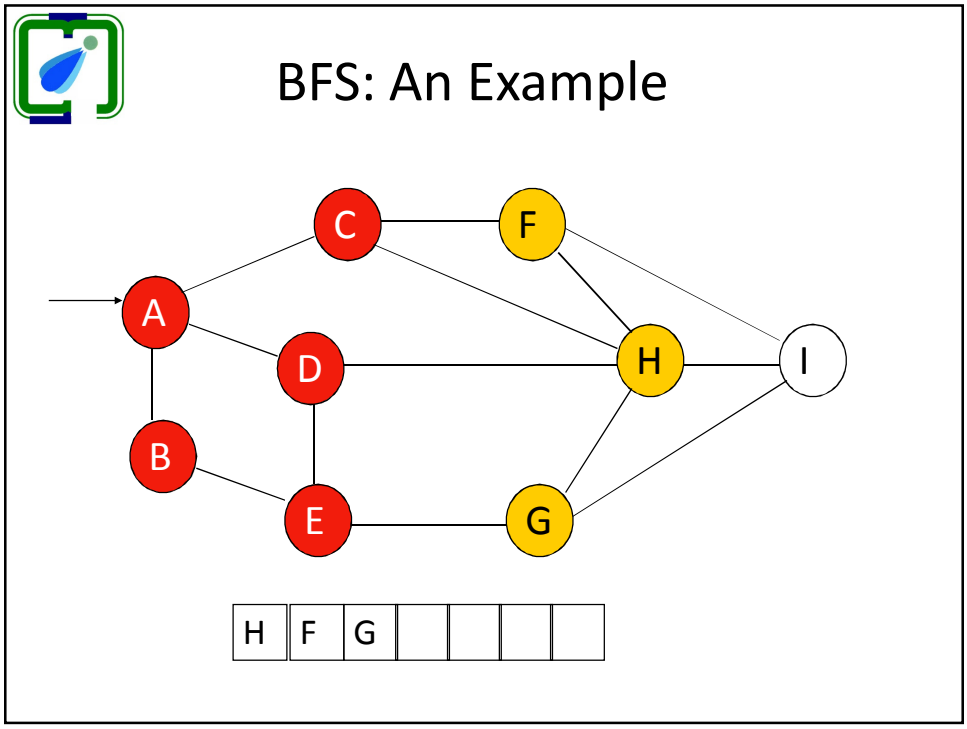

BFS: An Example
C
F
A
H
I
D
B
E
G
E
H

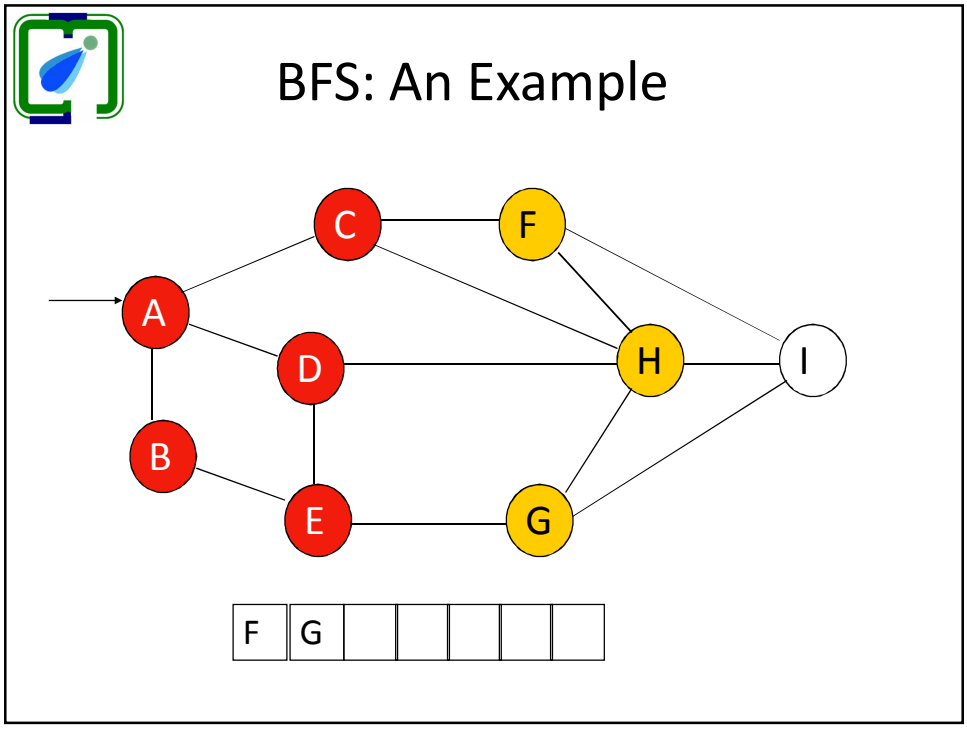

BFS: An Example
C
F
A
H
I
D
B
E
G
E
H
F

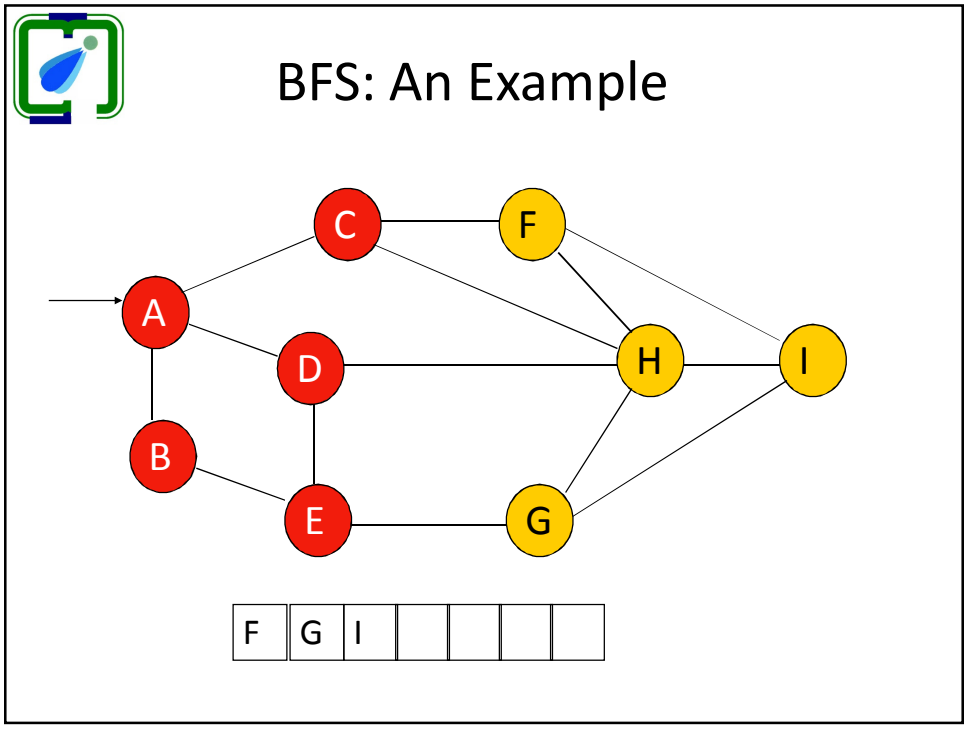

BFS: An Example
C
F
A
H
I
D
B
E
G
E H F

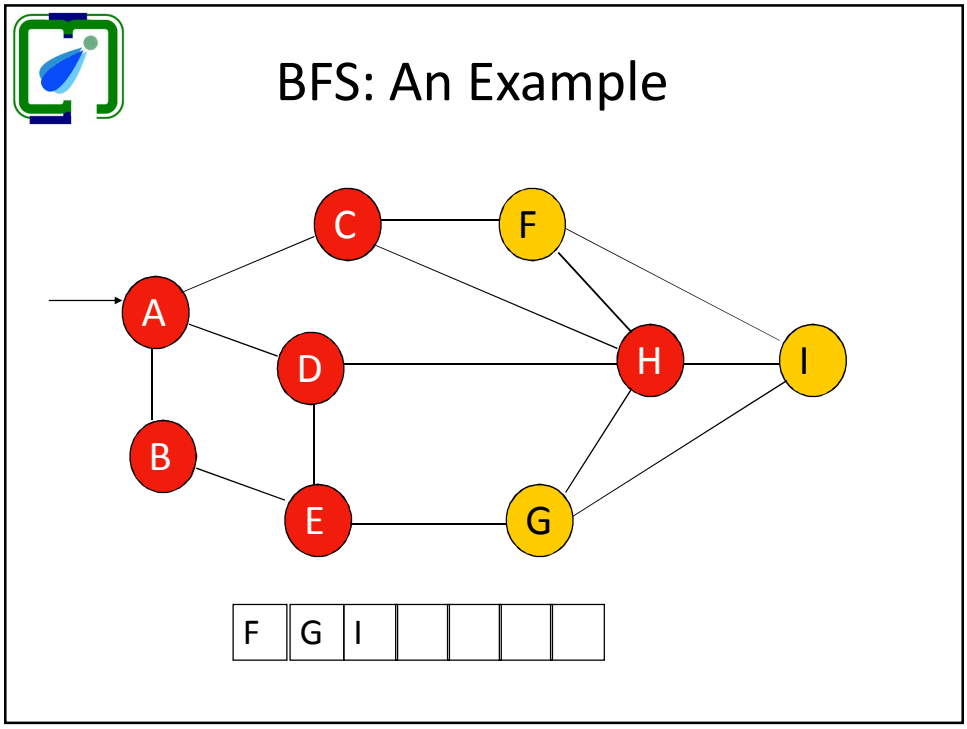

BFS: An Example
C
F
A
H
I
D
B
E
G
H F

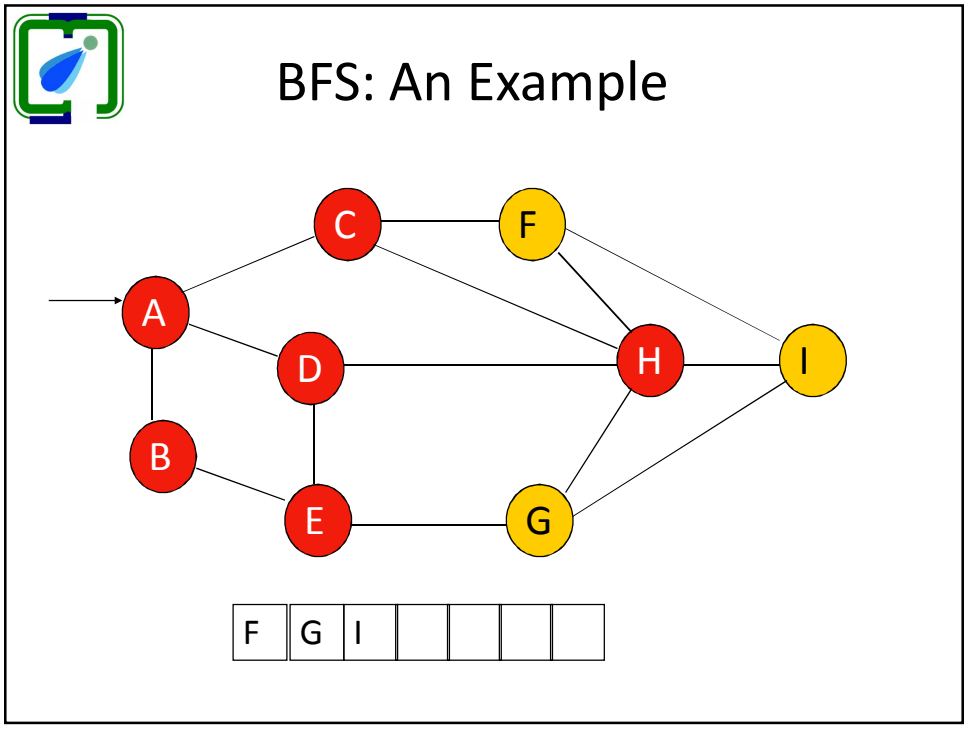

BFS: An Example
C
F
A
H
I
D
B
E
G
H
F
G

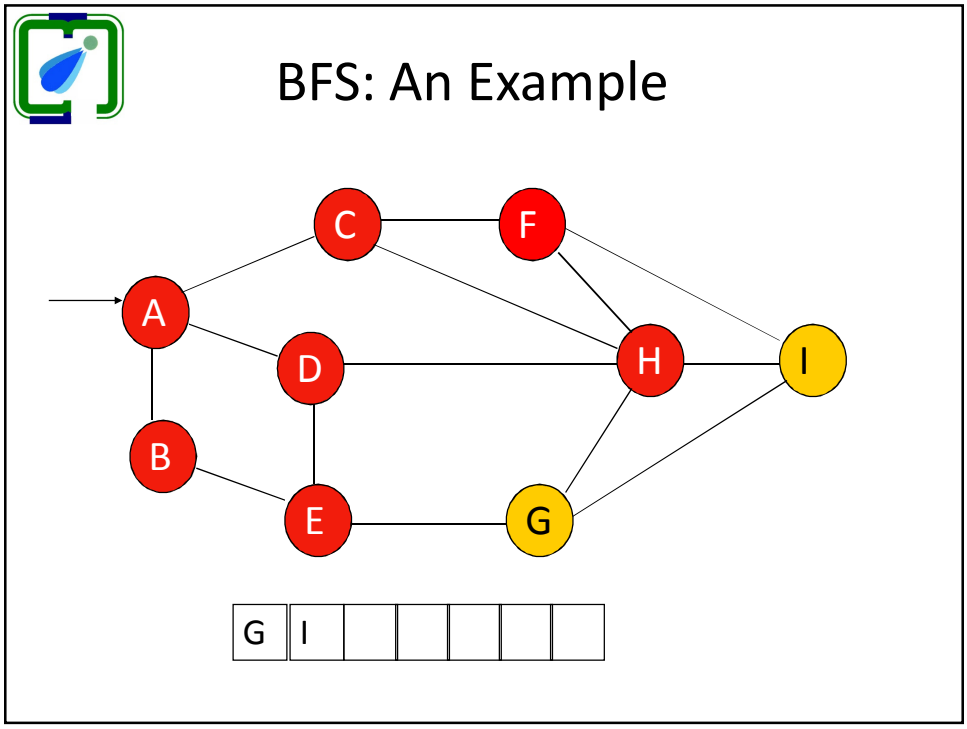

BFS: An Example
C
F
A
H
I
D
B
E
G
H
F
G

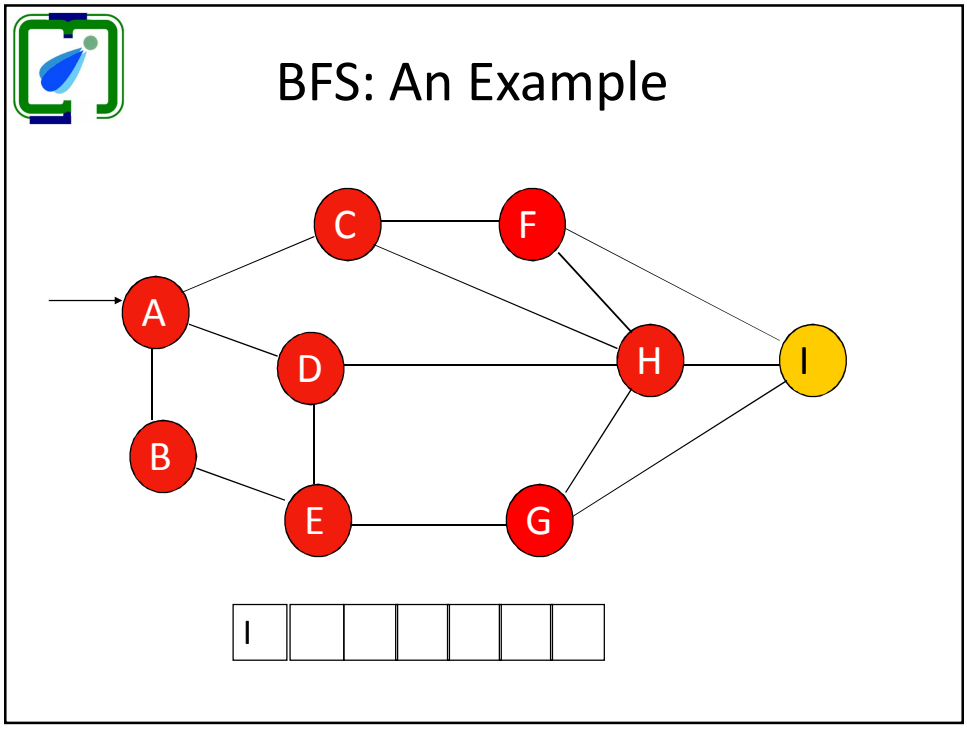

BFS: An Example
C
F
A
H
I
D
B
E
G
F
G

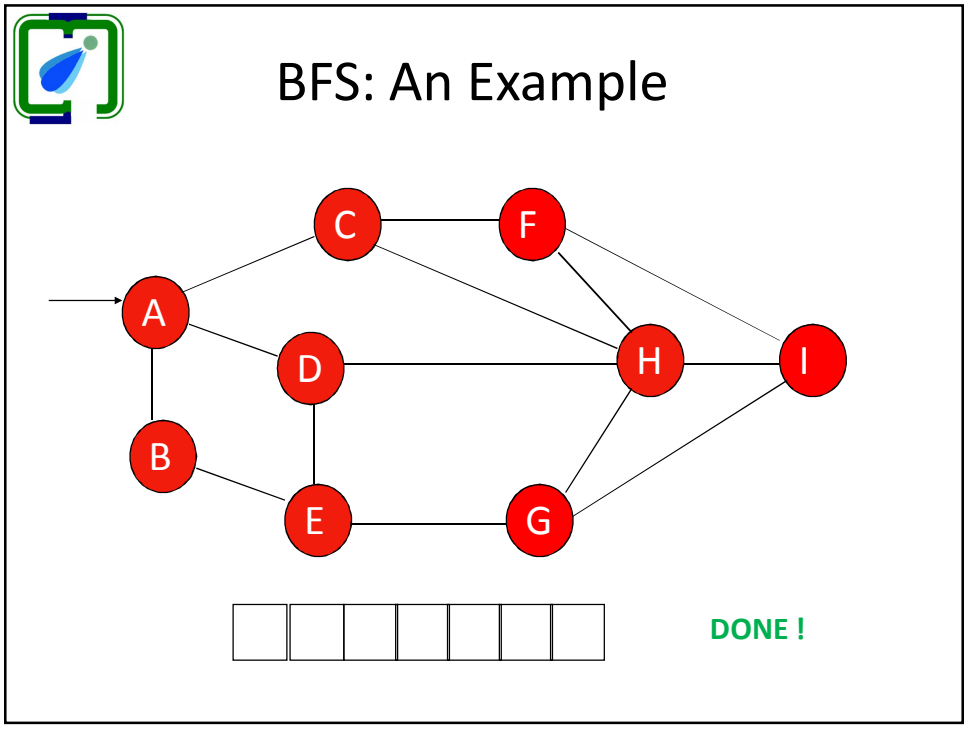

BFS: An Example
C
F
A
H
I
D
B
E
G
F
G
I


BFS: An Example
C
F
A
D
H
I
B
E
G
F G I

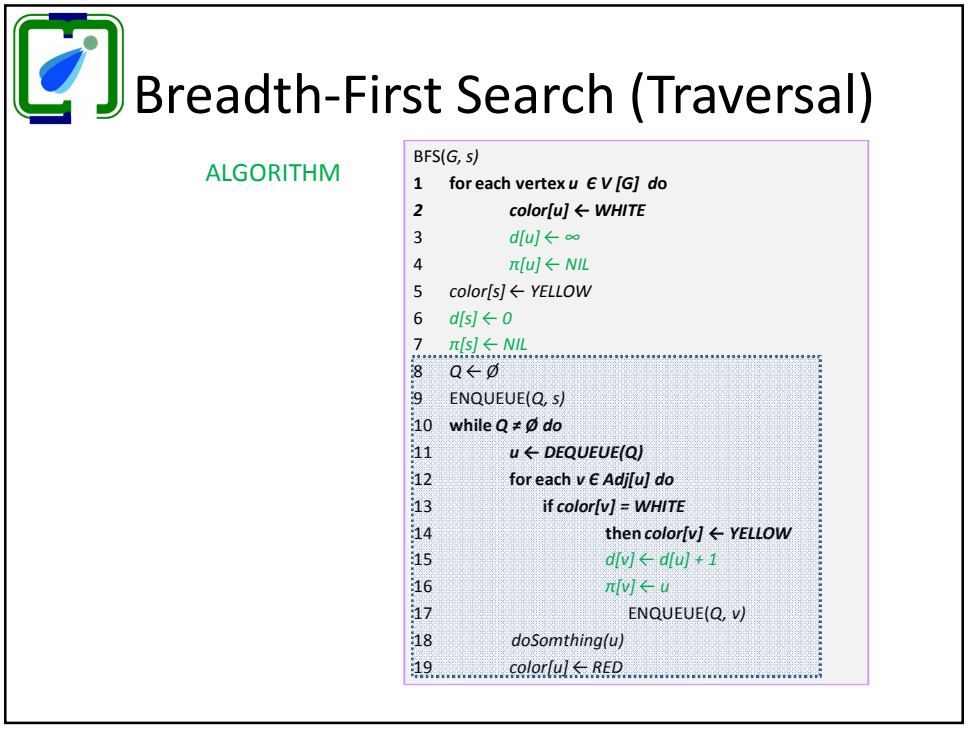

BFS: An Example
C
F
A
D
H
I
B
E
G
F G I


BFS: An Example
C
F
A
D
H
I
B
E
G
G
I


BFS: An Example
C
F
A
D
H
I
B
E
G
I



BFS: An Example



Breadth-First Search (Traversal)

ALGORITHM

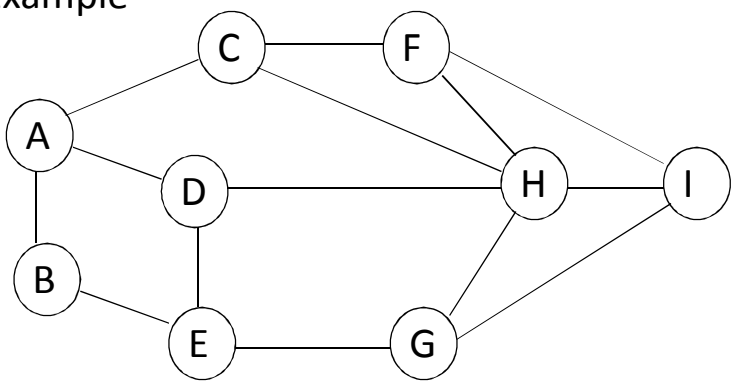
```
BFS(G, s)
1   for each vertex u ∈ V[G] do
2         color[u] ← WHITE
3         d[u] ← ∞
4         π[u] ← NIL
5   color[s] ← YELLOW
6   d[s] ← 0
7   π[s] ← NIL
8   Q ← Ø
9   ENQUEUE(Q, s)
10  while Q ≠ Ø do
11        u ← DEQUEUE(Q)
12        for each v ∈ Adj[u] do
13              if color[v] = WHITE
14                    then color[v] ← YELLOW
15                    d[v] ← d[u] + 1
16                    π[v] ← u
17                          ENQUEUE(Q, v)
18        doSomthing(u)
19        color[u] ← RED
```

Graph: Depth-First Traversal

- The Idea
  - Same as with tree traversal using DFS
  - Use a stack
  - Use three colors White, Yellow, and Red, similarly

Graph: Depth-First Traversal

- Example



Depth-First Traversal

```
DFS(G)
1 for each vertex u Є V [G] do
2         color[u] ← WHITE
3         π[u] ← NIL
4         time ← 0
5 for each vertex u Є V [G] do
6         if color[u] = WHITE then
7                 DFS-VISIT(u)
```

```
DFS-VISIT(u)
1 color[u] ← YELLOW
2 time ← time +1
3 d[u] ← time      //discovery time
4 for each v Є Adj[u] do
5         if color[v] == WHITE
6                 then π[v] ← u
7                 DFS-VISIT(v)
8 color[u] ← RED
9 f [u] ← time ← time +1
        //finish time
```

Problems Specific with Graphs

- Minimum Spanning Tree
- Path Problems
  - Simple Paths
  - Shortest Path Problem
    - Single source shortest paths
    - All-pair shortest paths
  - Find Cycles
  - Euler Path and Circuit Problem
  - Hamiltonian Path and Circuit Problem (or TSP)
- Graph Coloring
- Connected Components
- Isomorphic graphs
- Search Graphs

Graphs: Summary

- Two variants:
  - Directed and Undirected
- Representation
  - Array and Link-List Representation
- Traversals
  - BFS
  - DFS