GRAPH TRAVERSAL ALGORITHM

CSE [AIML], 6TH SEMESTER

DESIGN & ANALYSIS OF ALGORITHM

CONTENT

- Breadth First Search(BFS)
- Depth First Search(DFS)
- Classification of Edges
  - Tree
  - Forward
  - Back
  - Cross

Breadth First Search

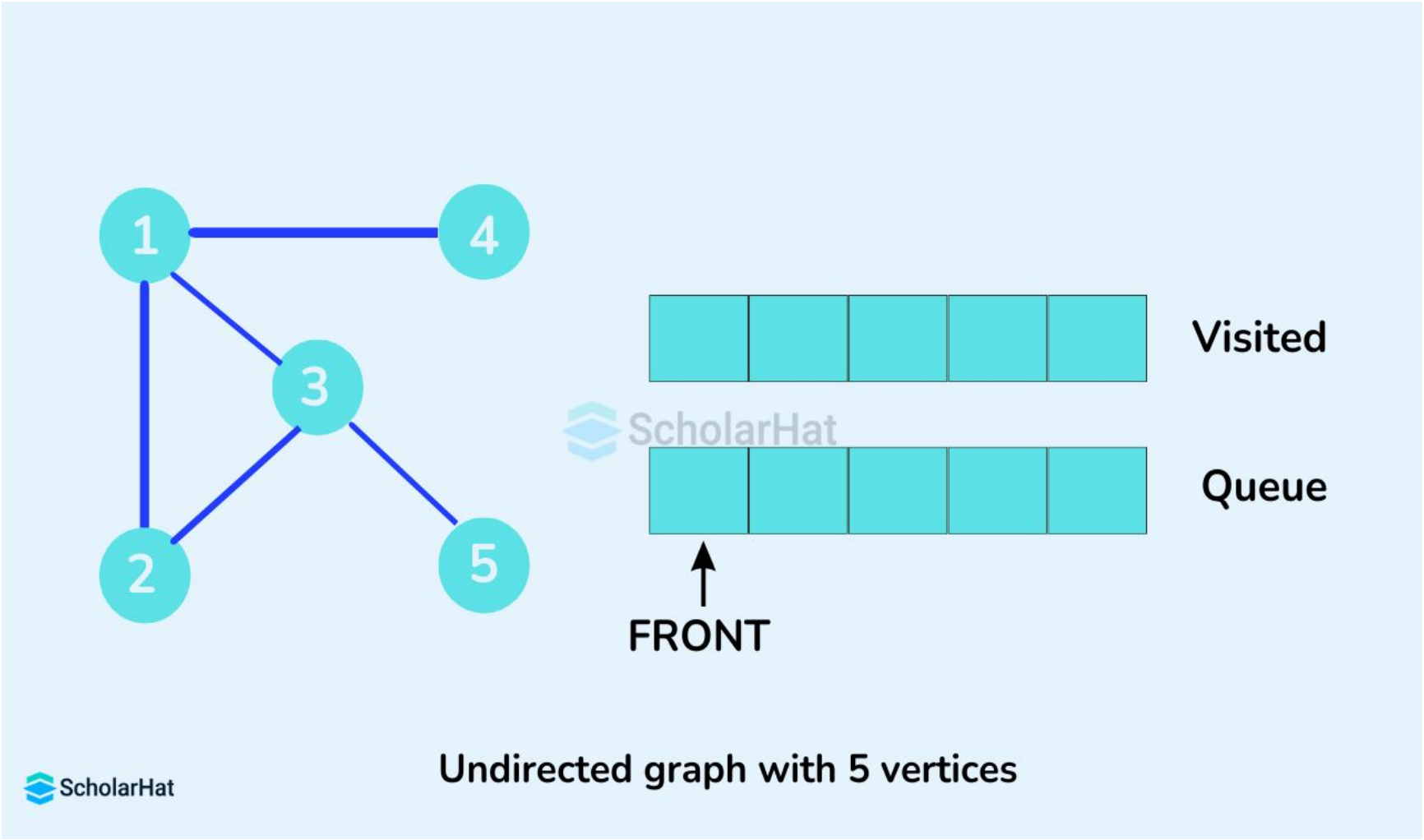
Breadth-first Search (BFS) is a graph traversal algorithm that explores all the vertices of a graph in a breadthwise order. It means it starts at a given vertex or node and visits all the vertices at the same level before moving to the next level. It is a recursive algorithm for searching all the vertices of a graph or tree data structure.

BFS Algorithm

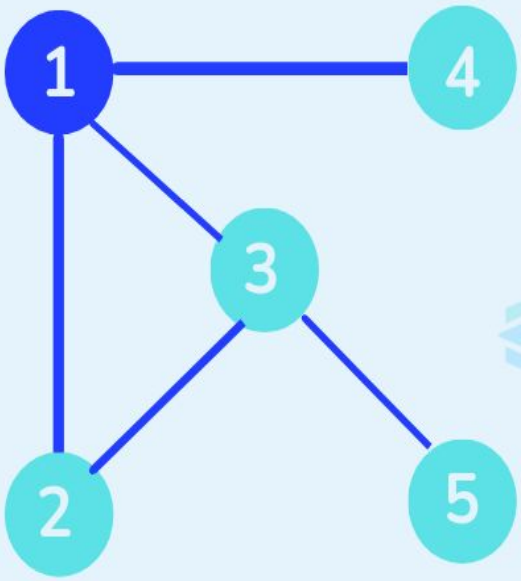
Here's a step-by-step explanation of how the BFS algorithm works:

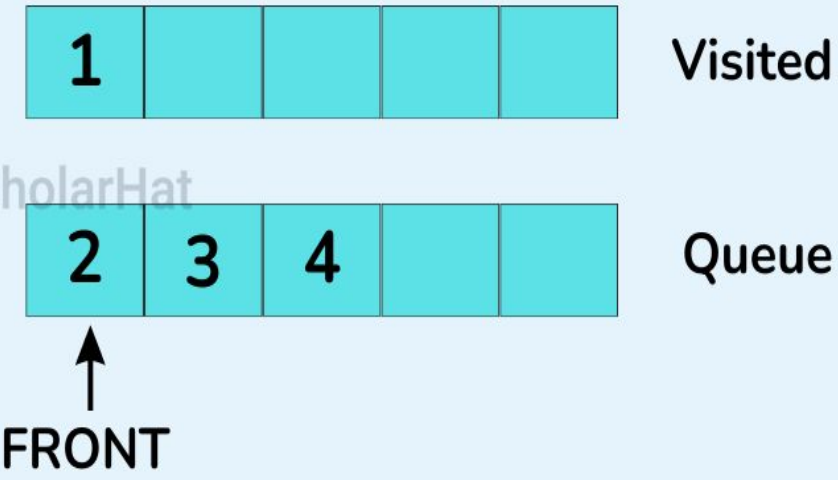
1. Start by selecting a starting vertex or node.
2. Add the starting vertex to the end of the queue.
3. Mark the starting vertex as visited and add it to the visited array/list.
4. While the queue is not empty, do the following steps:
    i) Remove the front vertex from the queue.
    ii) Visit the removed vertex and process it.
    iii) Enqueue all the adjacent vertices of the removed vertex that have not been visited yet.
    iv) Mark each visited adjacent vertex as visited and add it to the visited array/list.
5. Repeat step 4 until the queue is empty.

Example(copied from ScholarHat)



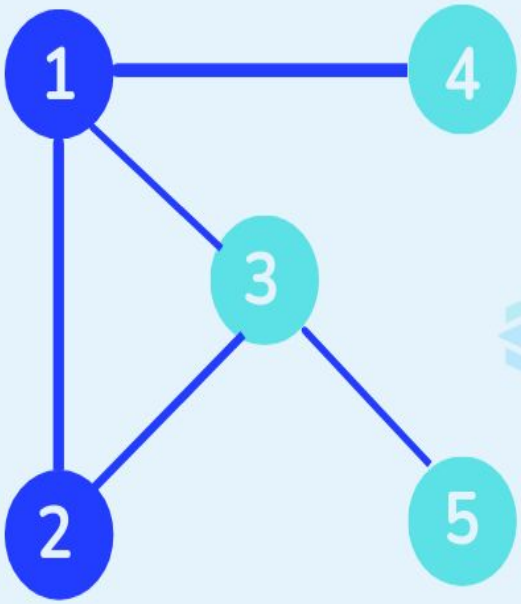
Example continues…





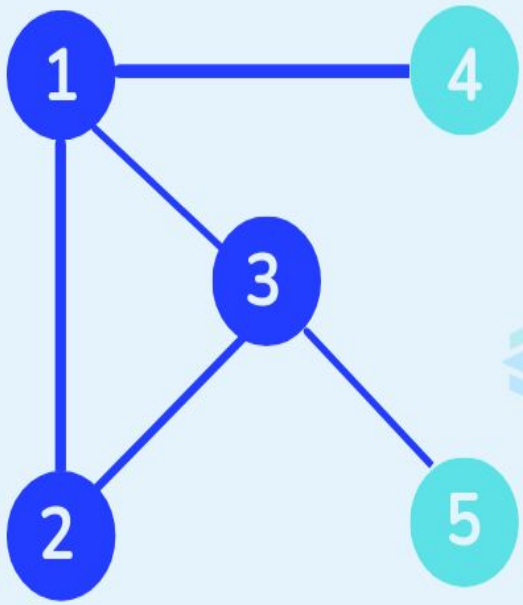
Visit start vertex and add its adjacent vertices to queue

Example continues…



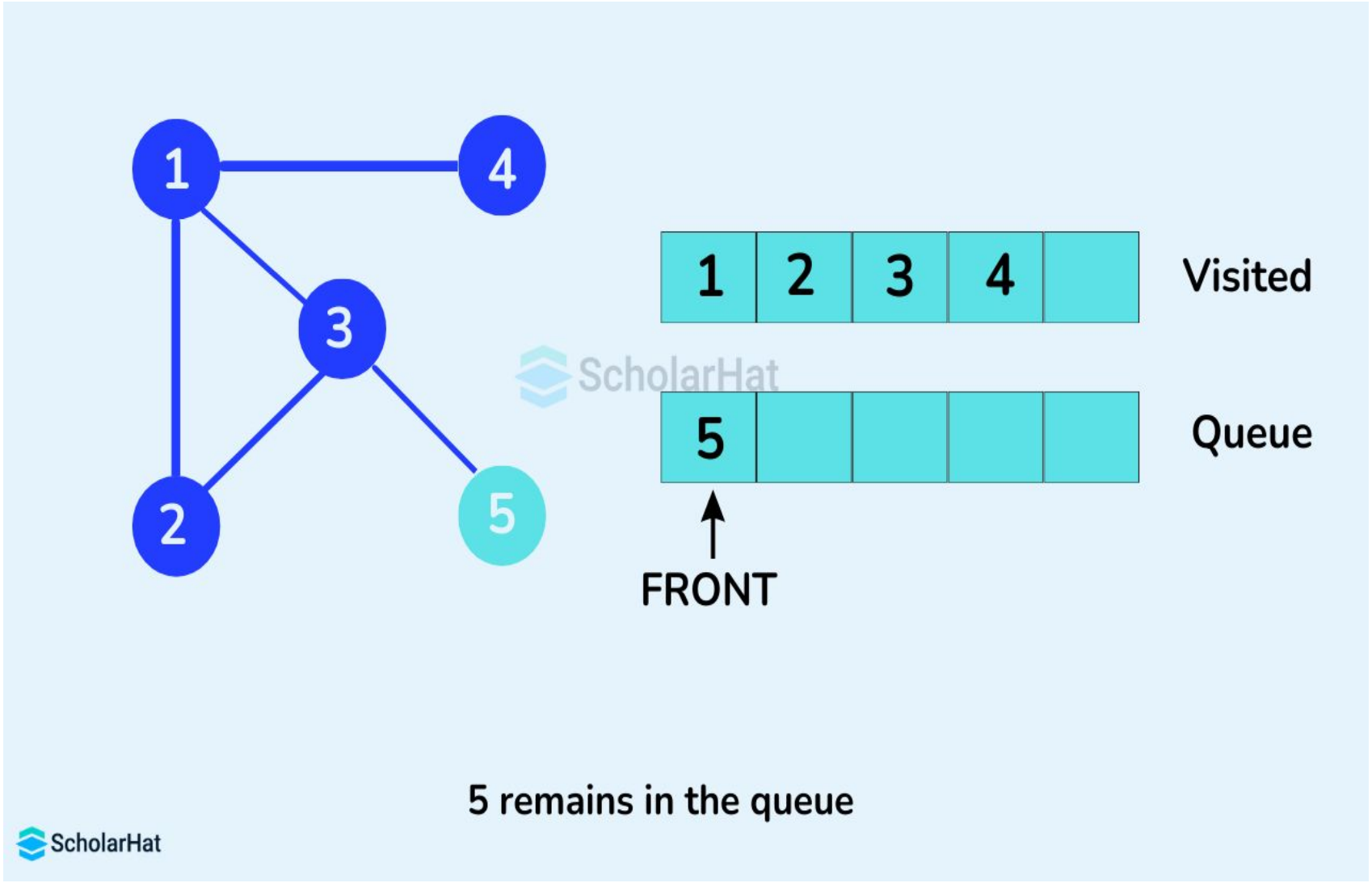
Visit the first neighbour of start node 1, which is 2

Example continues…

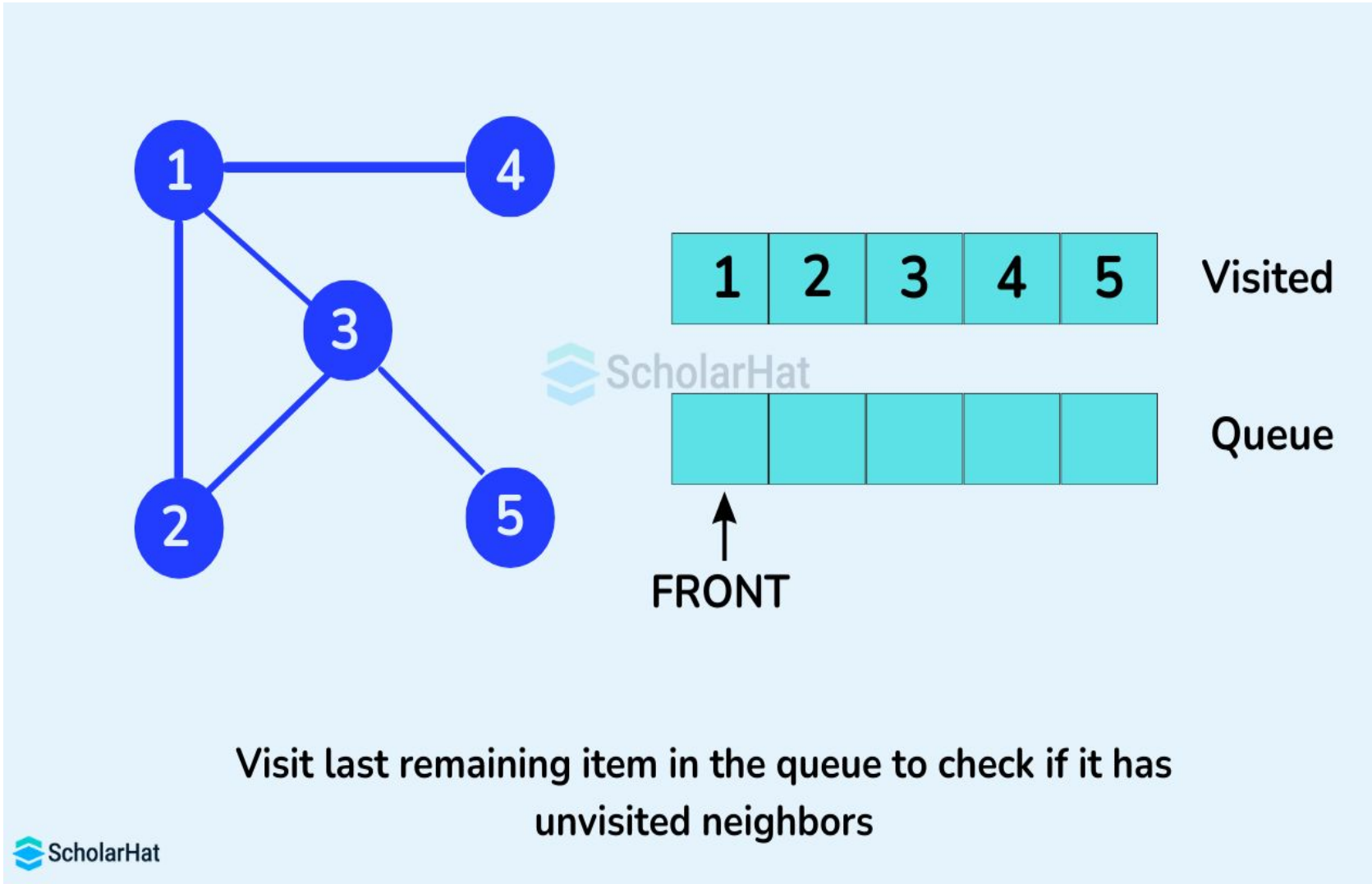


Visit 3 which was added to queue earlier to add its neighbours

Example continues…



Example continues…



Depth First Traversal

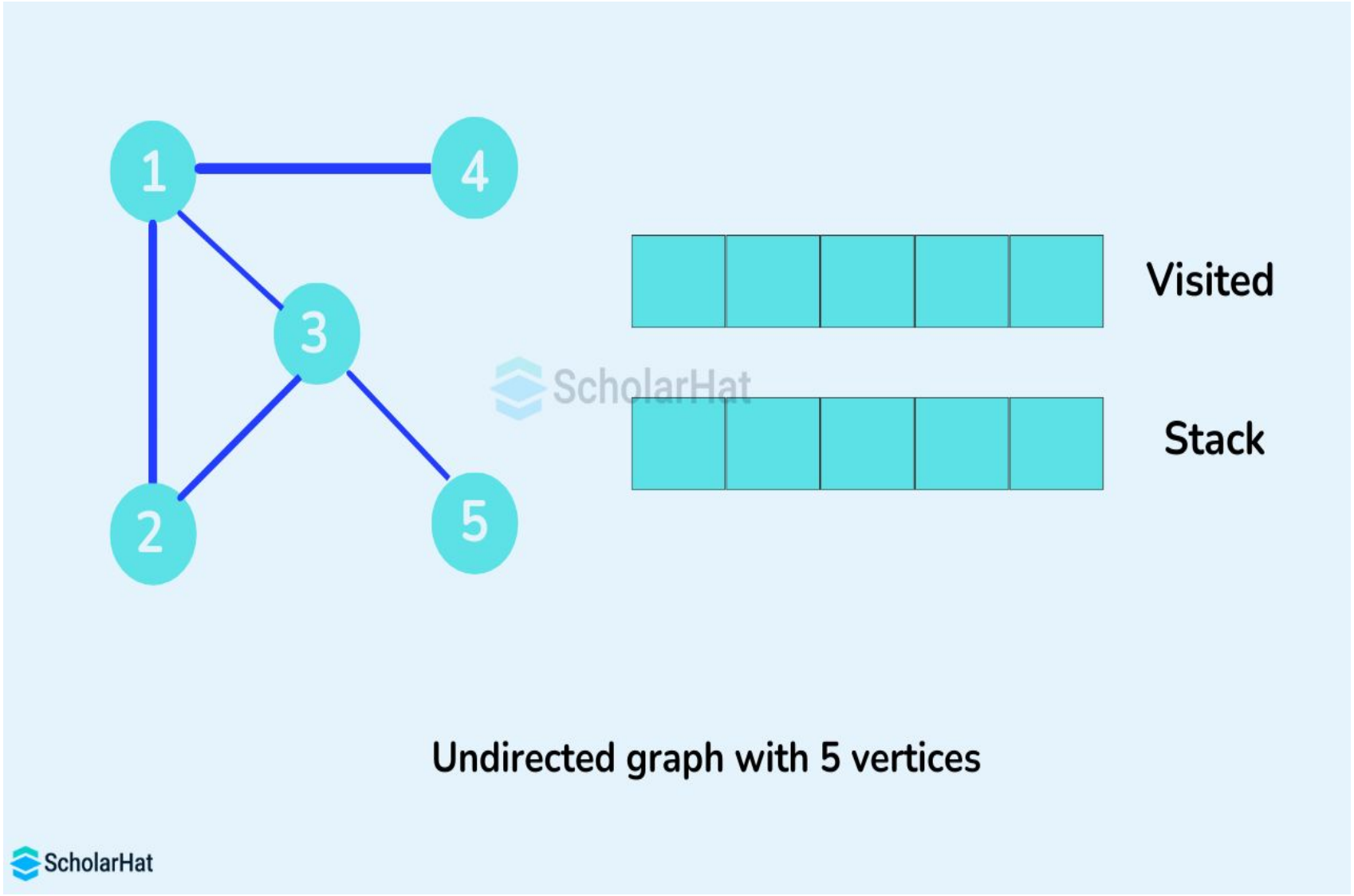
The depth-first search algorithm works by starting from an arbitrary node of the graph and exploring as far as possible before backtracking i.e. moving to an adjacent node until there is no unvisited adjacent node. After backtracking, it repeats the same process for all the remaining vertices which have not been visited till now. It is a recursive algorithm for searching all the vertices of a graph or tree data structure.

DFS Algorithm

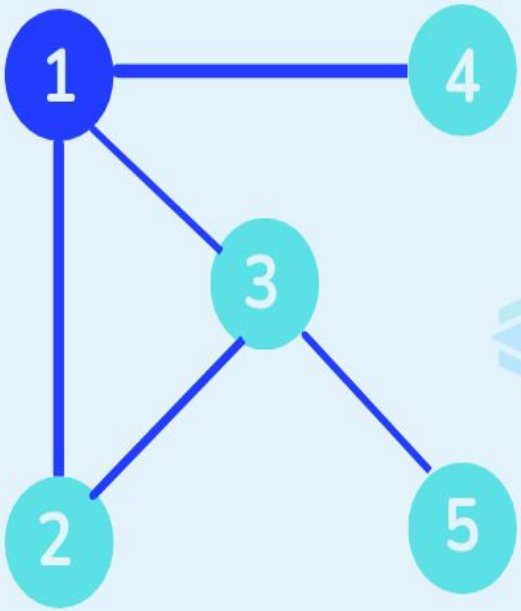
Here's a step-by-step explanation of how the DFS algorithm works::

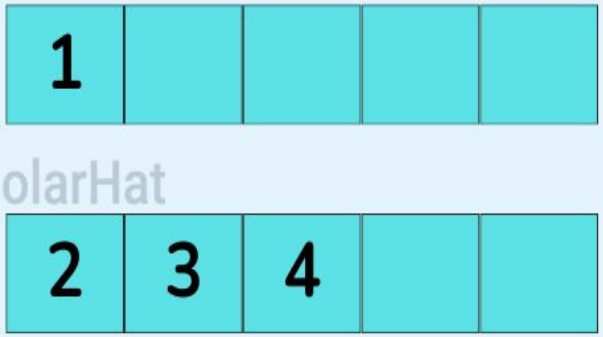
- Start by selecting a starting vertex or node.
- Add the starting vertex on top of a stack.
- Take the top item of the stack and add it to the visited list/array.
- Create a list of that vertex's adjacent nodes. Add the ones that aren't in the visited list to the top of the stack.
- Keep repeating steps 3 and 4 until the stack is empty.

Example(copied from ScholarHat)



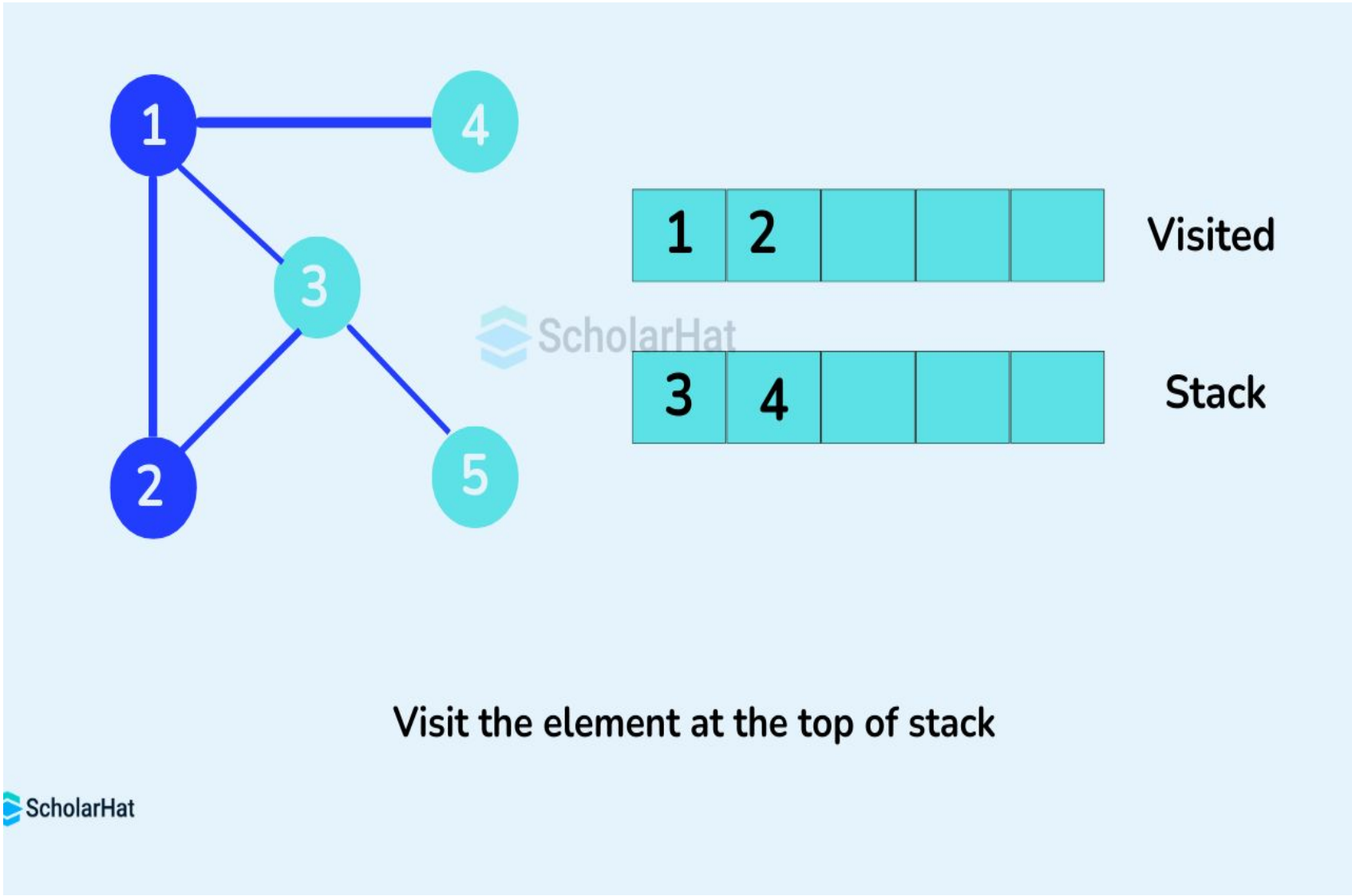
Example continues…



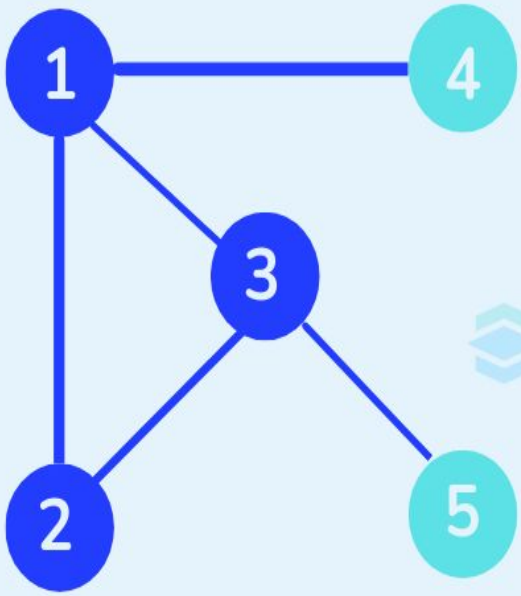


Visit the element and put it in the visited list

Example continues…



Example continues...



1	2	3		

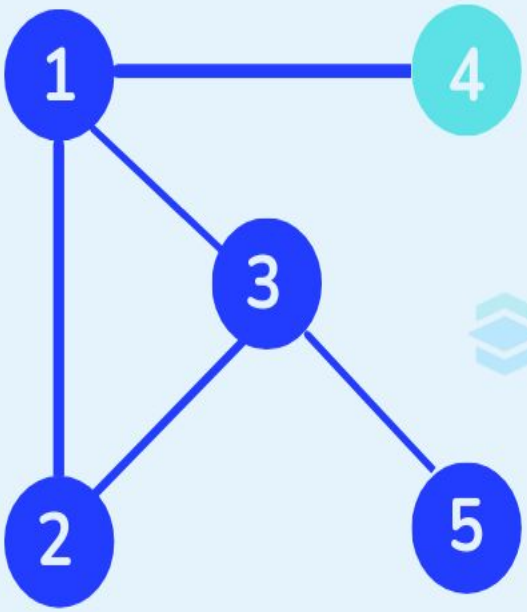
Visited

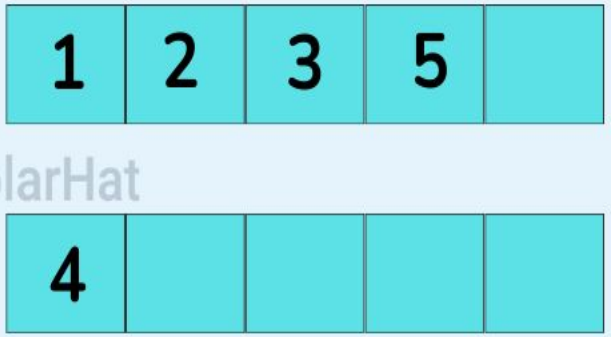
5	4			

Stack

Vertex 3 has an unvisited adjacent vertex in 5, so we add that to the top of the stack and visit it.

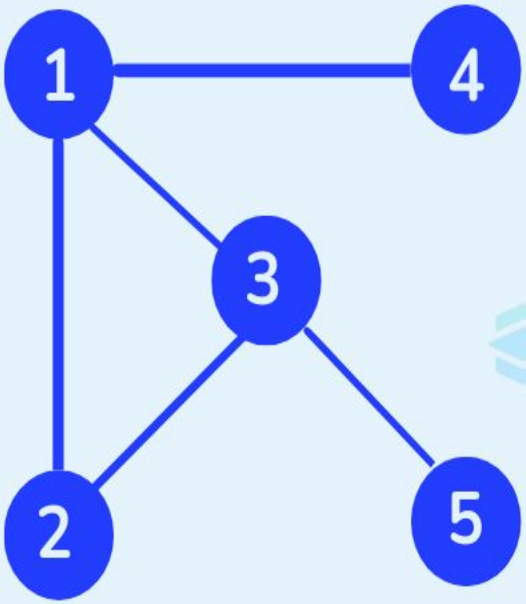
Example continues...

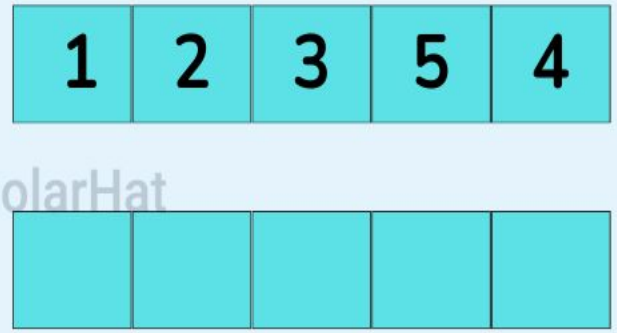




Vertex 1 has an unvisited adjacent vertex in 4, so we add that to the top of the stack and visit it.

Example continues...





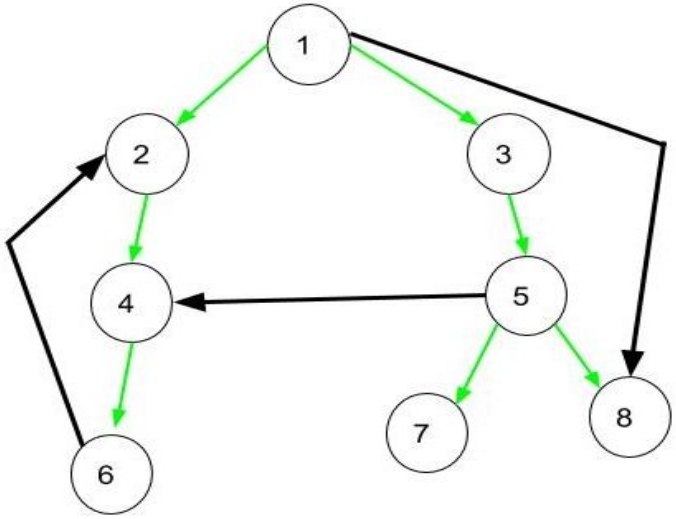
After we visit the last element 4, it doesn't have any unvisited adjacent nodes, so we have completed the Depth First Traversal of the graph.

Complexity Analysis of BFS and DFS Algorithms

ALGORITHM	TIME COMPLEXITY	SPACE COMPLEXITY
BFS	O(V + E)	O(V)
DFS	O(V + E)	O(V)

CLASSIFICATION OF EDGES

Given a directed graph, the task is to identify tree, forward, back and cross edges present in the graph.
Note: There can be multiple answers.



Output:
Tree Edges: 1->2, 2->4, 4->6, 1->3, 3->5, 5->7, 5->8
Forward Edges: 1->8
Back Edges: 6->2
Cross Edges: 5->4

CLASSIFICATION OF EDGES CONTINUES…

- Tree Edge: Tree edges connect a parent to its child node. It is an edge which is present in the tree obtained after applying DFS/BFS on the graph.
- Forward Edge: Forward edges link a node to its descendent that has already been traversed..
- Back edge: Back edges link a node to its ancestor that has already been traversed. Presence of back edge indicates a cycle in directed graph.
- Cross Edge: Cross edges connect two nodes that don't share any ancestor-descendent relation.

EDGE TYPE IN GRAPH TRAVERSAL

Edge Types in Graph Traversal		
Traversal Algorithm	Type of Graph	
	Undirected	Directed
Breadth-First Search (BFS)	Tree, Cross	Tree, Back, Cross
Depth-First Search (DFS)	Tree, Back	Tree, Forward, Back, Cross

END OF PRESENTATION

THANK YOU