Graph Theory and Applications

SEDDIK Mohamed Taki Eddine

Batna 2 UNIVERSITY
Faculty of Mathematics and Computer Science
Department of Computer Science

December 15, 2023

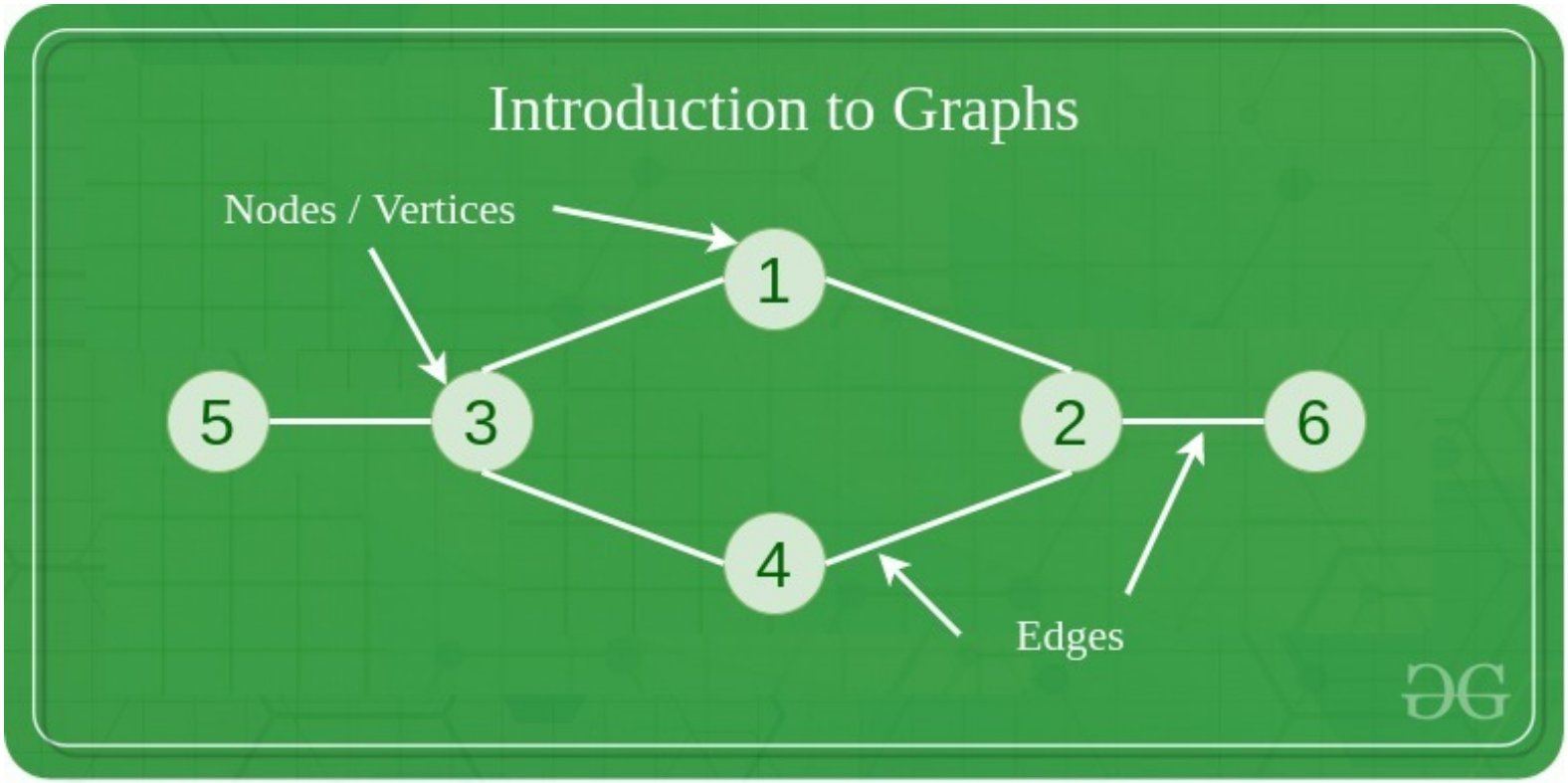
Sorting Algorithms Outline I

Introduction to Graph Theory

- Definition: Graph theory is a field of mathematics and computer science that studies the properties of graphs.
- Importance: Widely used in computer science, engineering, biology, social science, and many other fields.
- Applications: Network analysis, circuit design, scheduling, data organization, etc.

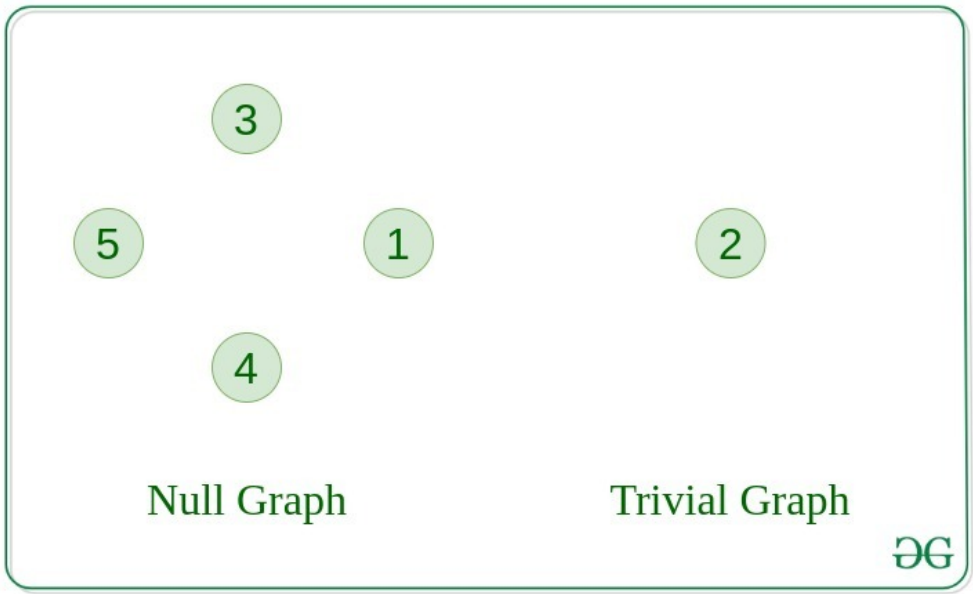
Defining Graphs in Computer Science I

- Graph Structure: A set of nodes (or vertices) and a set of edges connecting pairs of nodes.



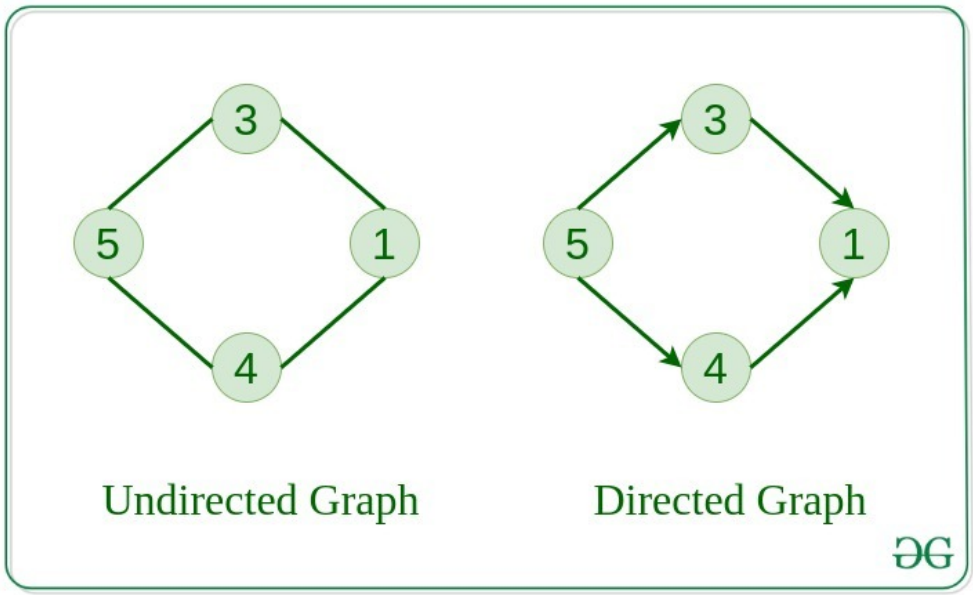
Defining Graphs in Computer Science II

- Types of Graphs:
  - Null Graph :A graph is known as a null graph if there are no edges in the graph.
  - Trivial Graph : Graph having only a single vertex, it is also the smallest graph possible.



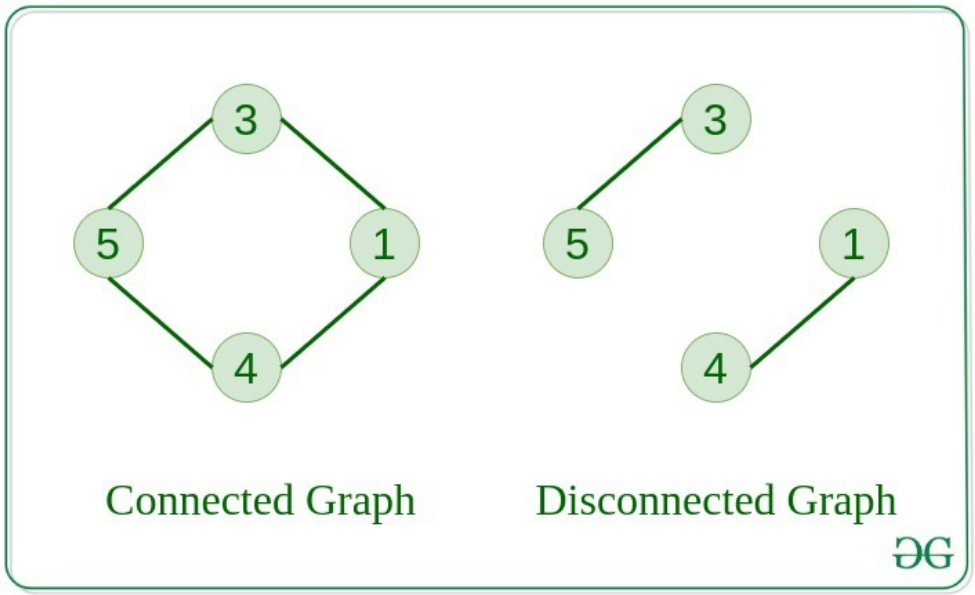
Defining Graphs in Computer Science III

- Directed:A graph in which edges do not have any direction. That is the nodes are unordered pairs in the definition of every edge.
- Undirected graphs:A graph in which edge has direction. That is the nodes are ordered pairs in the definition of every edge.



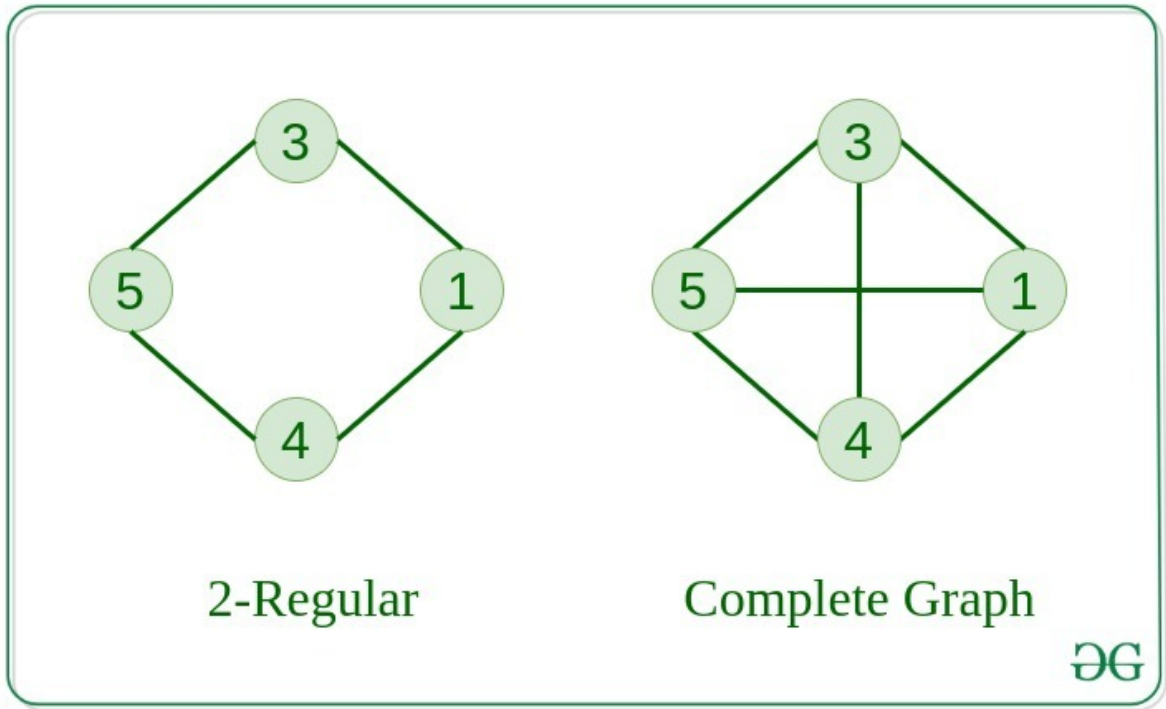
Defining Graphs in Computer Science IV

- Connected:The graph in which from one node we can visit any other node in the graph is known as a connected graph.
- Disconnected Graph: The graph in which at least one node is not reachable from a node is known as a disconnected graph.



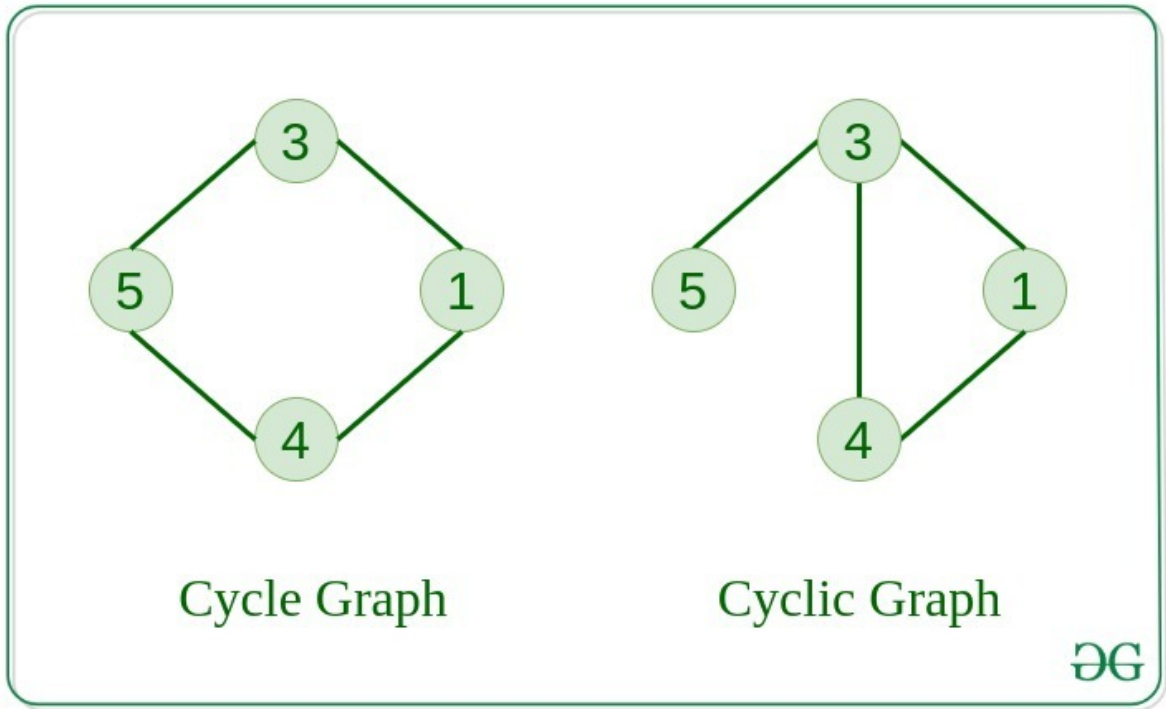
Defining Graphs in Computer Science V

- Regular Graph: The graph in which the degree of every vertex is equal to K is called K regular graph.
- Complete Graph : The graph in which from each node there is an edge to each other node.



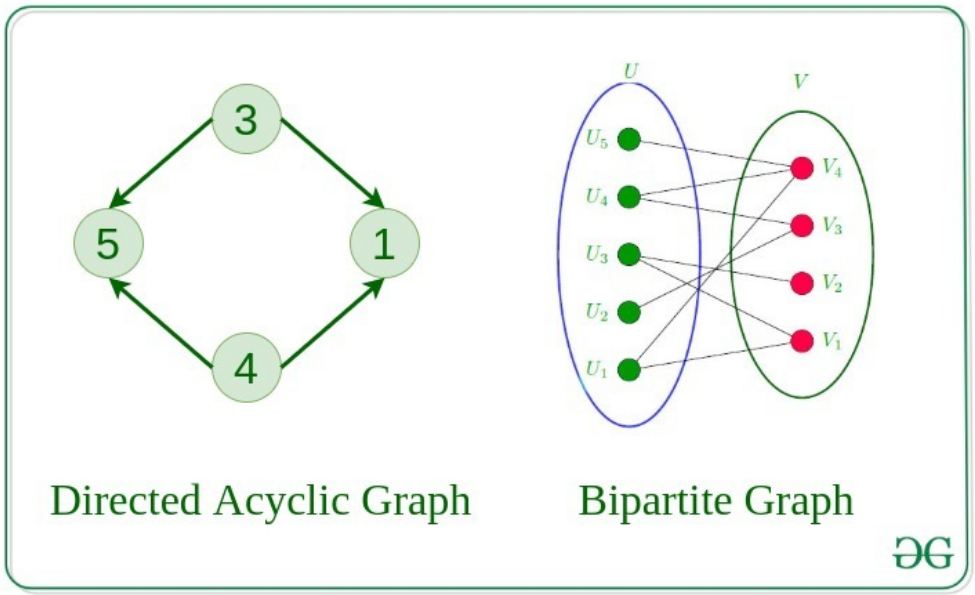
Defining Graphs in Computer Science VI

- Cycle Graph:The graph in which the graph is a cycle in itself, the degree of each vertex is 2.
- Cyclic Graph: A graph containing at least one cycle is known as a Cyclic graph.



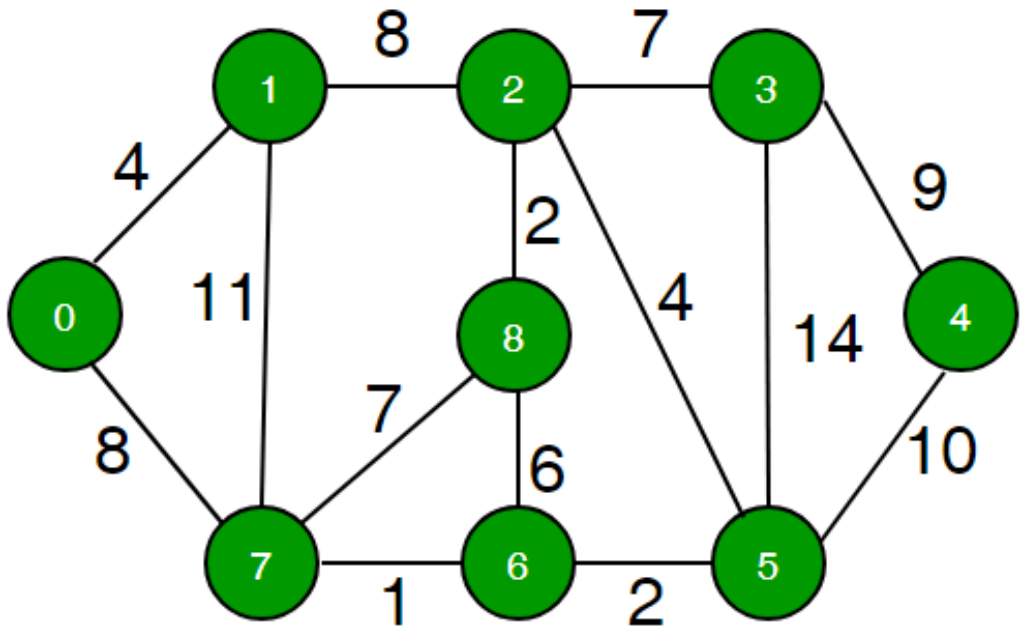
Defining Graphs in Computer Science VII

- Directed Acyclic Graph A Directed Graph that does not contain any cycle.
- Bipartite Graph: A graph in which vertex can be divided into two sets such that vertex in each set does not contain any edge between them.



Defining Graphs in Computer Science VIII

- Weighted Graph:A graph in which the edges are already specified with suitable weight is known as a weighted graph.



- Trees and Lists.

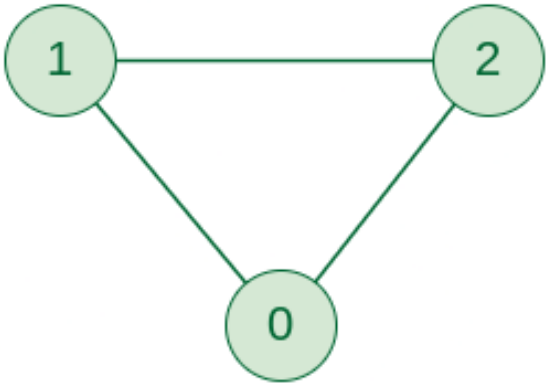
Overview of Graph Representation

- Importance: Effective representation of graphs is crucial for efficient graph algorithms.
- Criteria: Space complexity, ease of implementing operations, and suitability for specific algorithms.
- Common Methods: Adjacency matrices, incidence matrices, and adjacency lists.

Adjacency Matrix Representation I

- Definition: A square matrix used to represent a finite graph.
- Representation: The element at row i and column j represents the presence (or absence) of an edge between vertex i and vertex j.
- Pros and Cons: Simple representation but may consume more space, especially for sparse graphs.

Adjacency Matrix Representation II



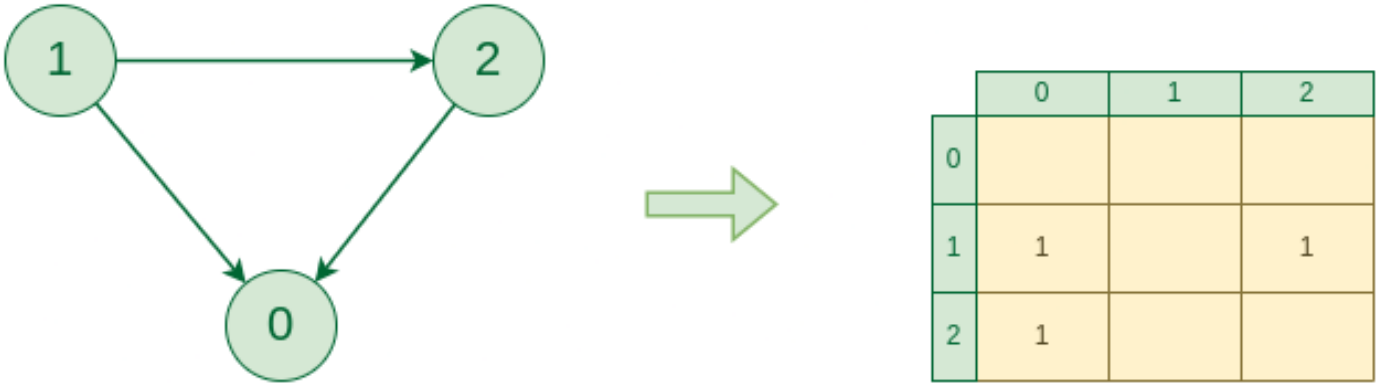
Undirected Graph

	0	1	2
0		1	1
1	1		1
2	1	1	

Adjacency Matrix

Graph Representation of Undirected graph to Adjacency Matrix

Adjacency Matrix Representation III

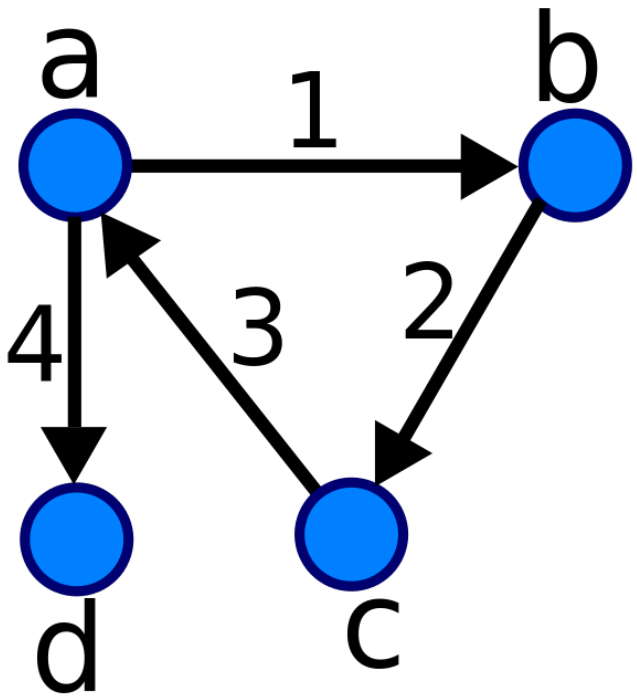


Graph Representation of Directed graph to Adjacency Matrix

Incidence Matrix Representation I

- Definition: A matrix that shows the relationship between vertices and edges.
- Representation: Rows represent vertices, columns represent edges, and matrix entries indicate which vertex is incident to which edge.
- Usage: Useful for edge-centric operations and for graphs with more edges than vertices.

Incidence Matrix Representation II



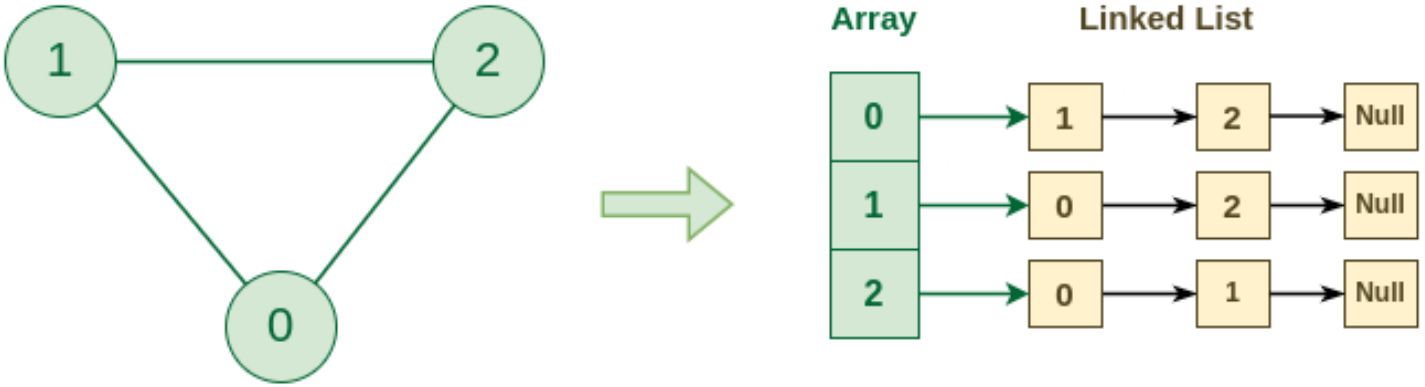
	1	2	3	4
a	1	0	-1	1
b	-1	1	0	0
c	0	-1	1	0
d	0	0	0	-1

Figure: Graph Representation of directed graph to Incedence Matrix

Adjacency List Representation I

- Definition: A collection of lists used to represent a finite graph.
- Structure: Each list describes the set of neighbors of a vertex in the graph.
- Efficiency: More space-efficient for sparse graphs, and allows for faster traversal operations.

Adjacency List Representation II

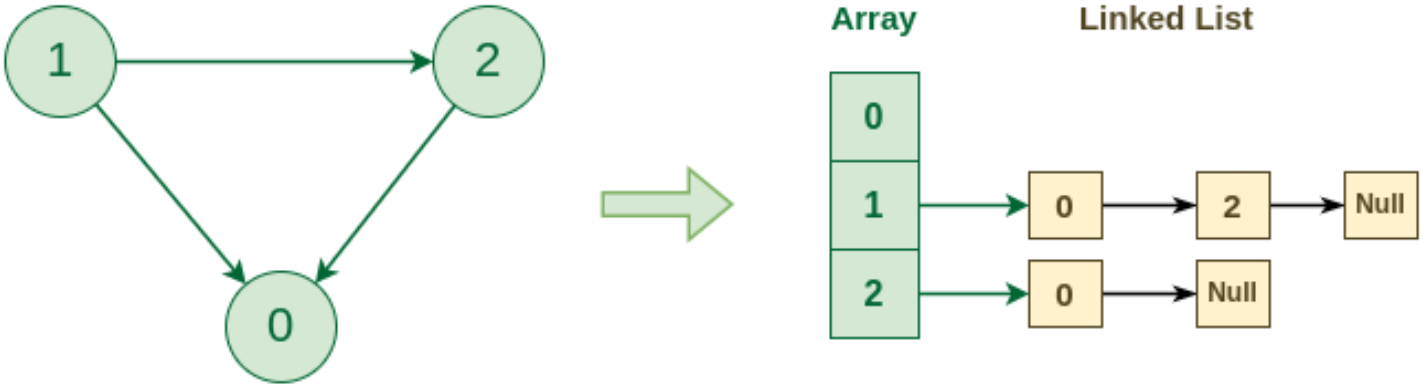


Undirected Graph

Adjacency List

Graph Representation of Undirected graph to Adjacency List

Adjacency List Representation III



Directed Graph

Adjacency List

Graph Representation of Directed graph to Adjacency List

Overview of Graph Traversal

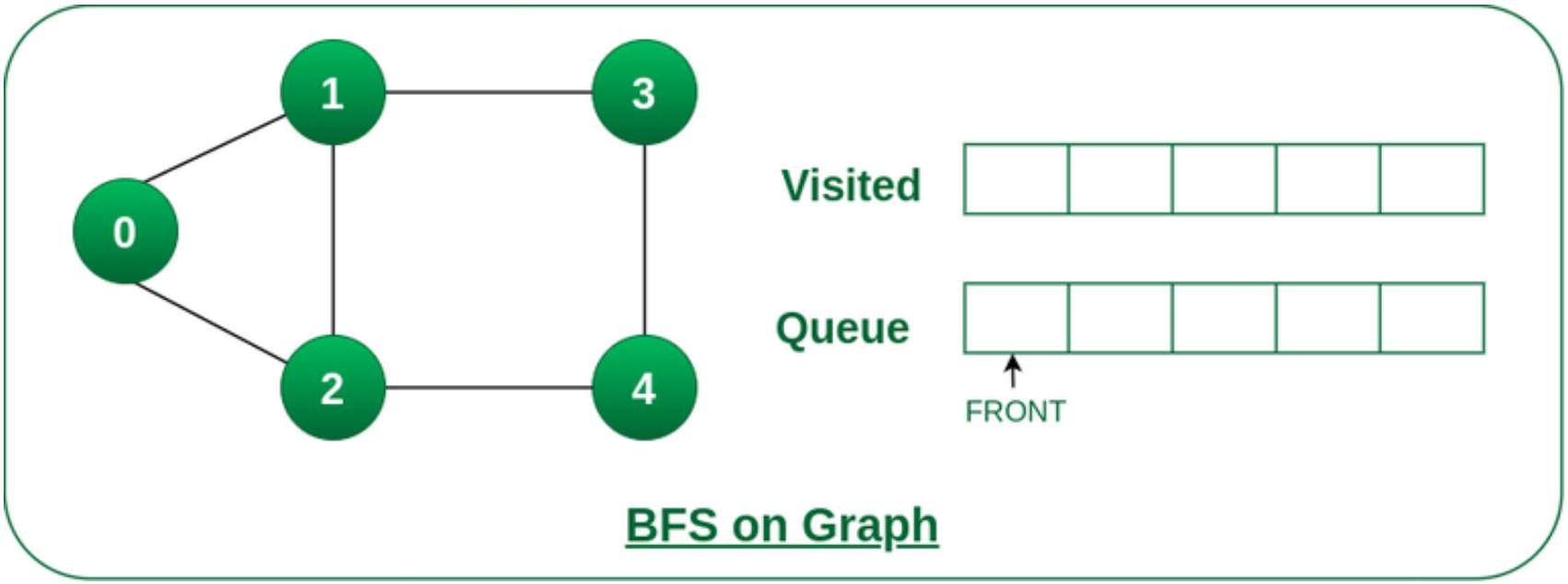
- Definition: Graph traversal refers to the process of visiting all the vertices in a graph.
- Purpose: Used for searching a graph, path finding, and analyzing structure.
- Common Methods: Breadth-First Search, Depth-First Search, Dijkstra's Algorithm, etc.

Breadth-First Search

- Concept: BFS explores the graph level by level starting from a selected node.
- Process: It uses a queue to keep track of the next vertex to visit.
- Application: Used in shortest path finding in unweighted graphs, networking, and more.

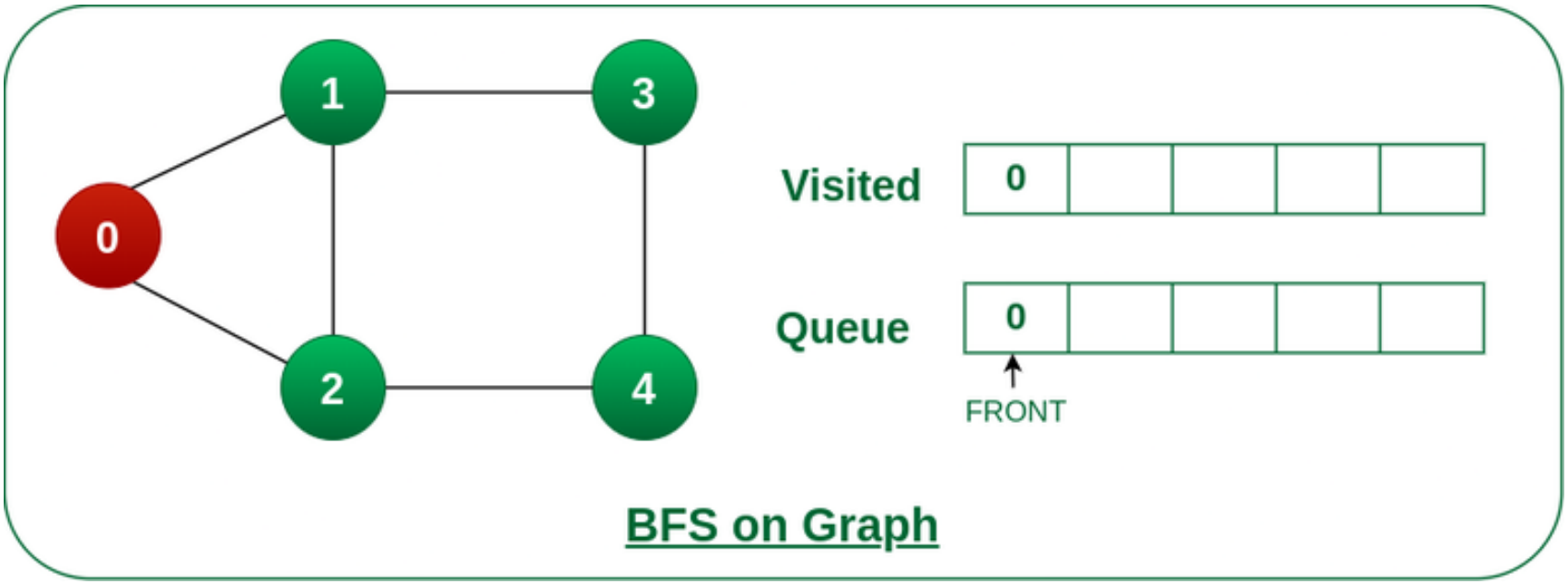
Example of Breadth-First Search I

- Step1: Initially queue and visited arrays are empty.



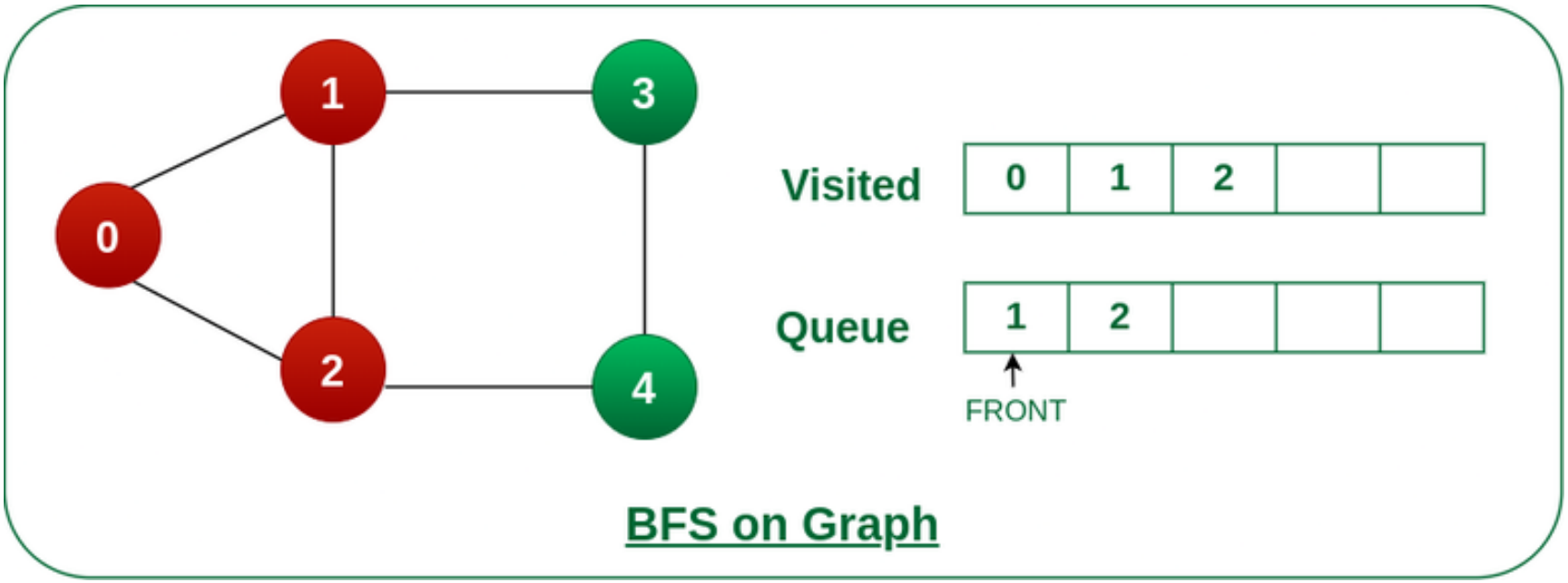
Example of Breadth-First Search II

- Step2: Push node 0 into queue and mark it visited.



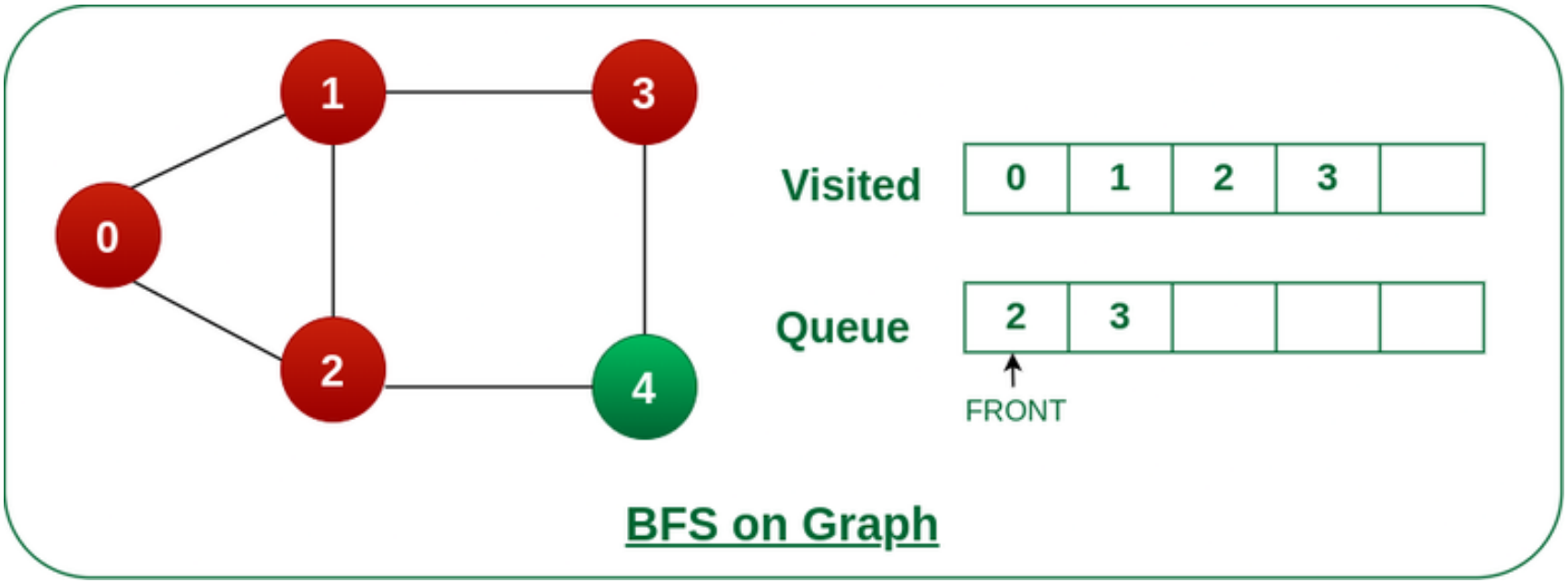
Example of Breadth-First Search III

- Step 3: Remove node 0 from the front of queue and visit the unvisited neighbours and push them into queue.



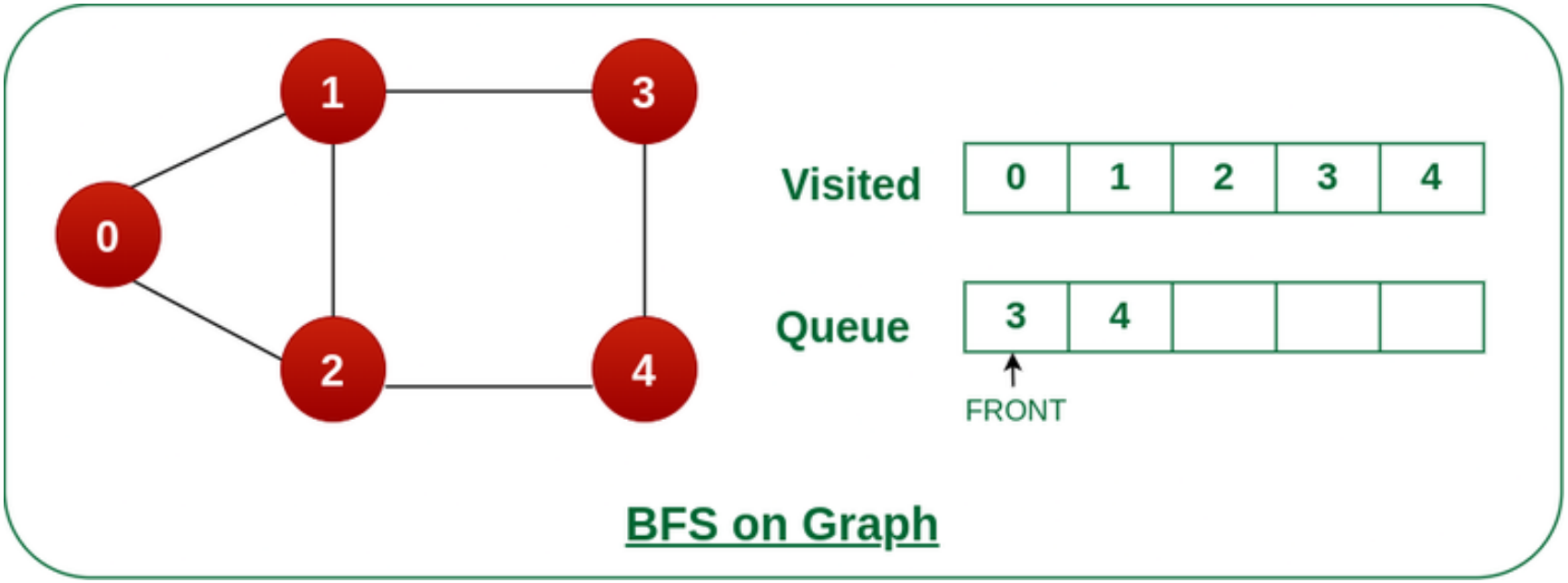
Example of Breadth-First Search IV

- Step 4: Remove node 1 from the front of queue and visit the unvisited neighbours and push them into queue.



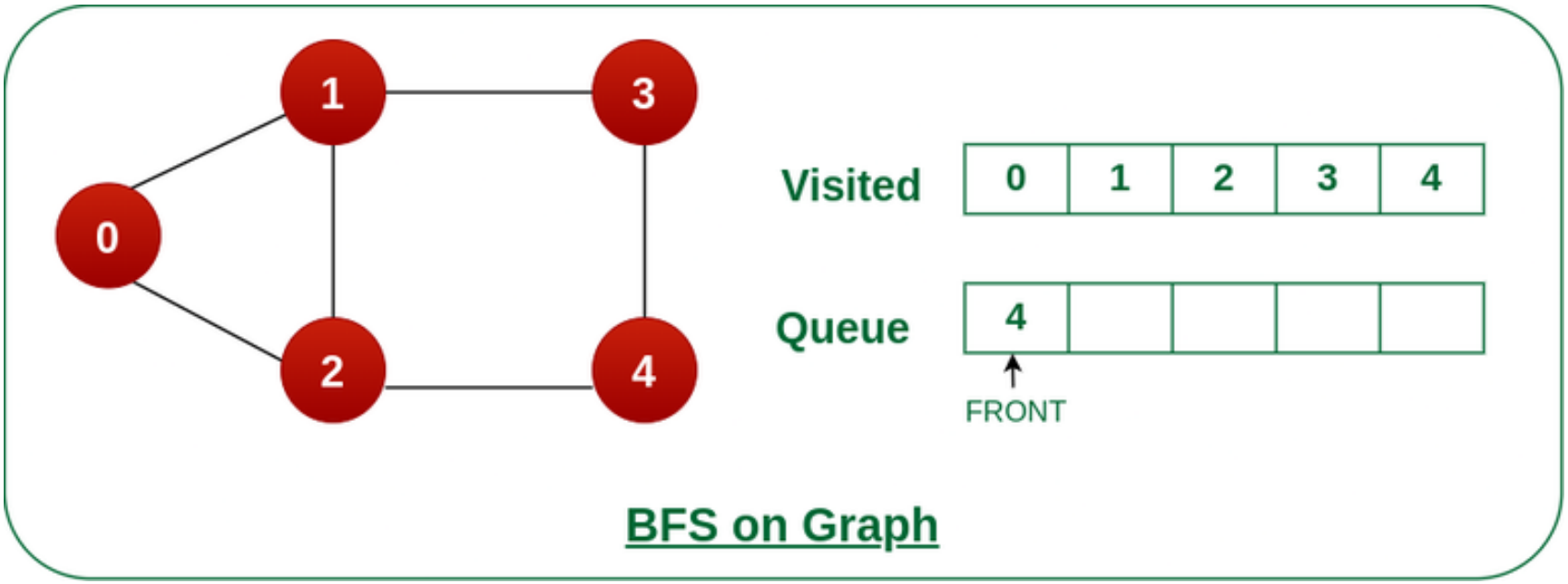
Example of Breadth-First Search V

- Step 5: Remove node 2 from the front of queue and visit the unvisited neighbours and push them into queue.



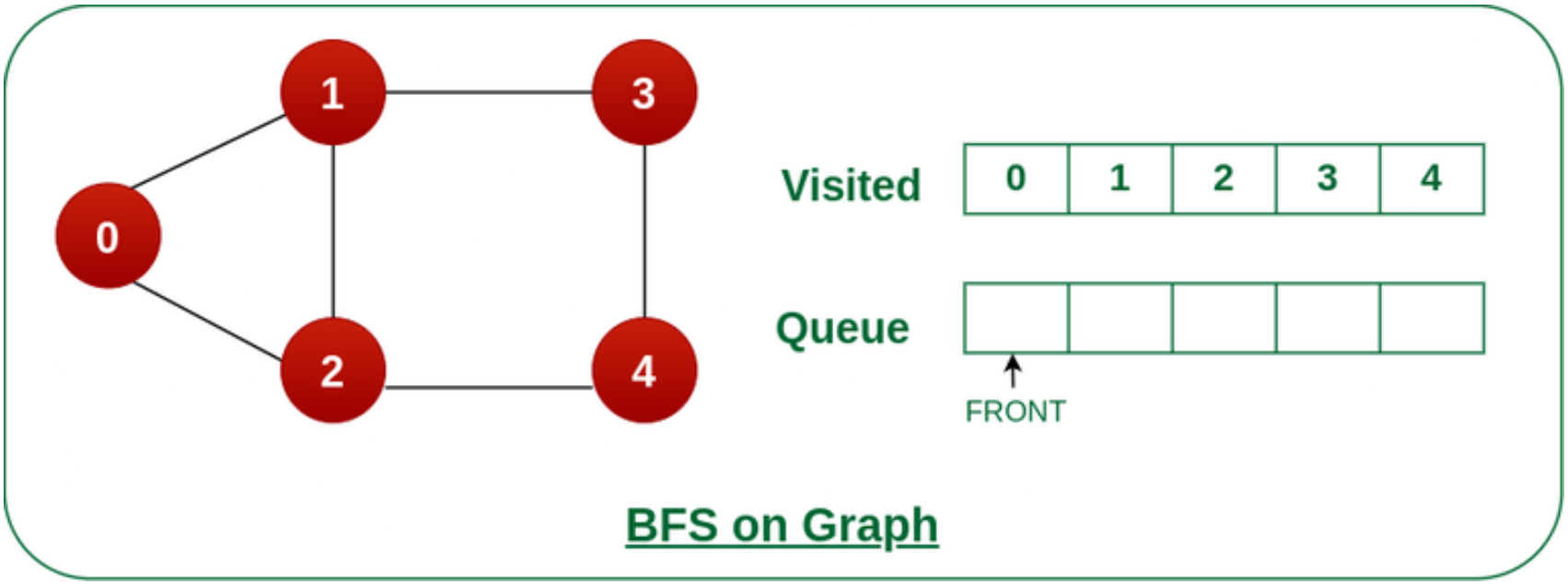
Example of Breadth-First Search VI

- Step 6: Remove node 3 from the front of queue and visit the unvisited neighbours and push them into queue. As we can see that every neighbours of node 3 is visited, so move to the next node that are in the front of the queue.



Example of Breadth-First Search VII

- Steps 7: Remove node 4 from the front of queue and visit the unvisited neighbours and push them into queue. As we can see that every neighbours of node 4 are visited, so move to the next node that is in the front of the queue.



Example of Breadth-First Search VIII

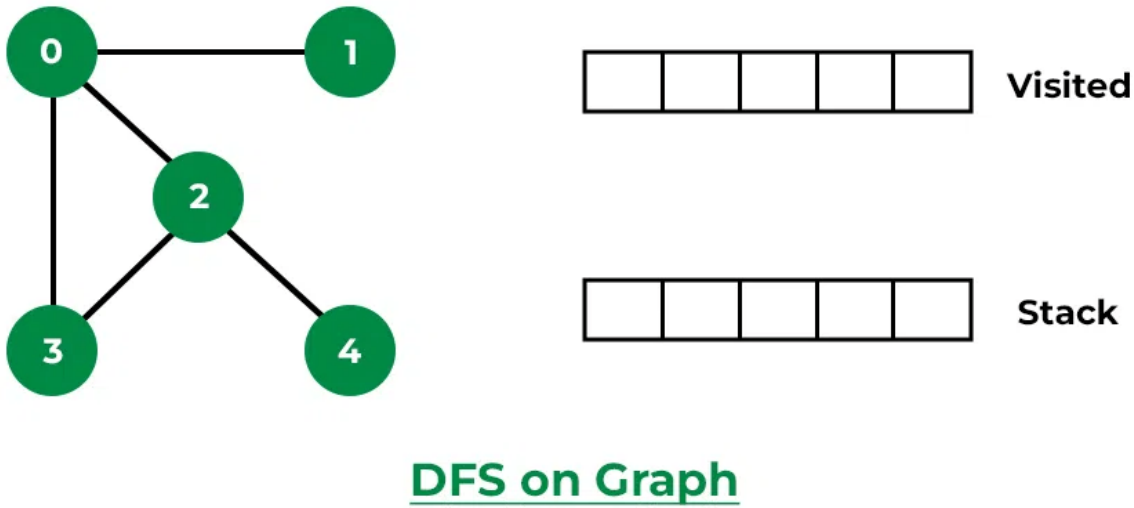
- Now, Queue becomes empty, So, terminate these process of iteration.

Depth-First Search

- Concept: DFS explores as far as possible along each branch before backtracking.
- Process: It uses a stack (either explicit or via recursion) for traversal.
- Application: Used for path finding, topological sorting, and solving puzzles with only one solution.

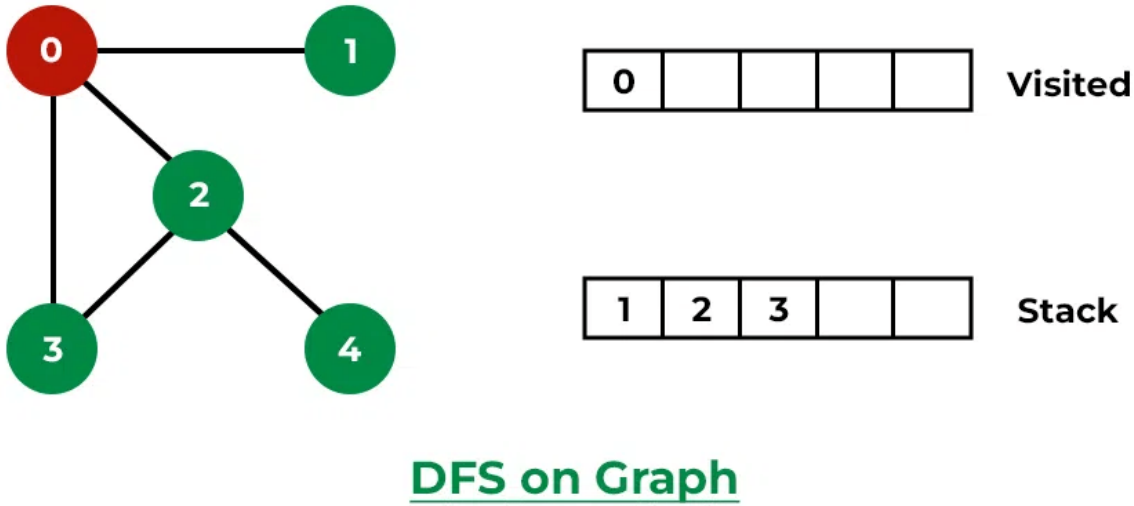
Example of Depth-First Search I

- Step1: Initially stack and visited arrays are empty.



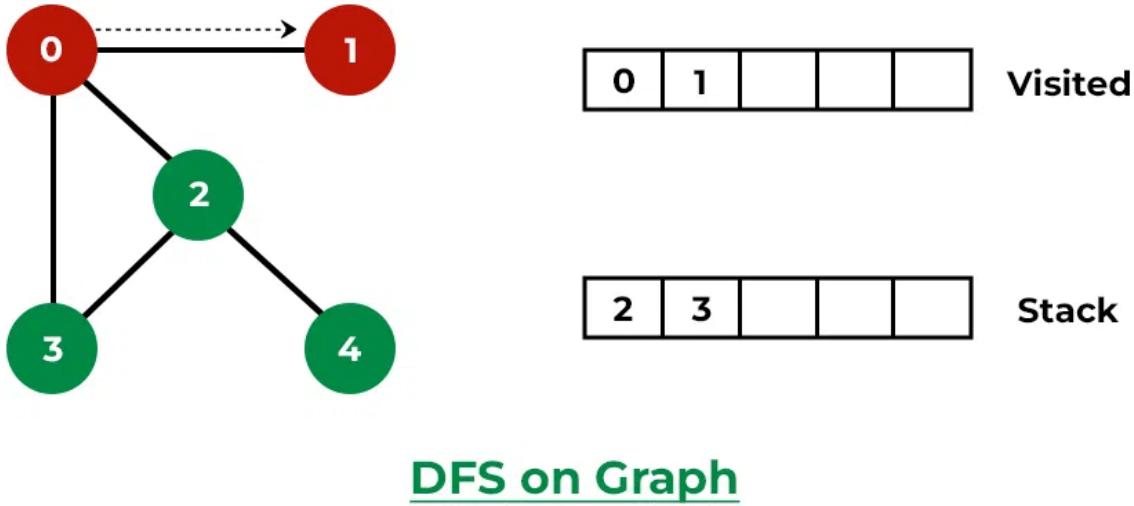
Example of Depth-First Search II

- Step 2: Visit 0 and put its adjacent nodes which are not visited yet into the stack.



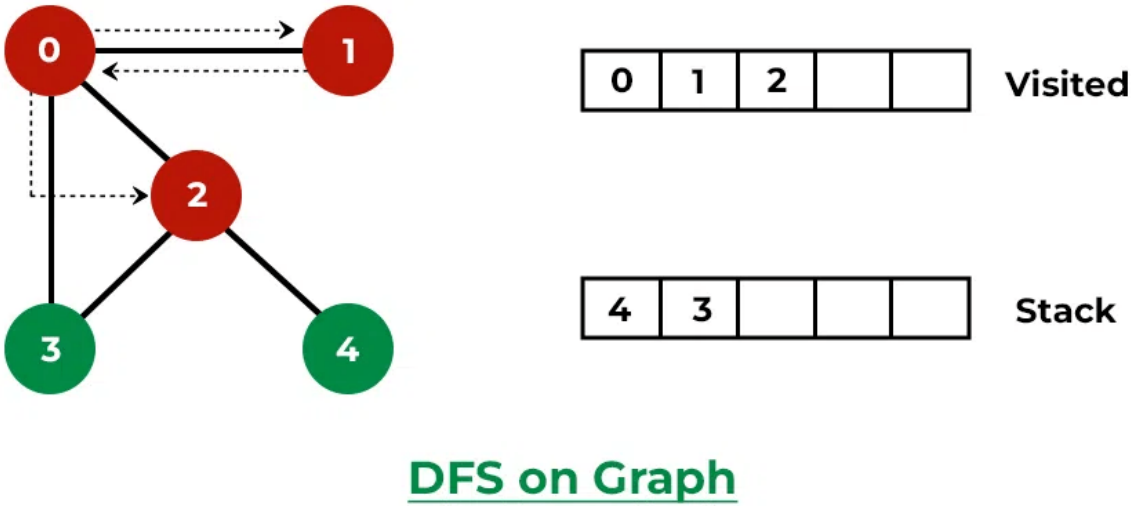
Example of Depth-First Search III

- Step 3: Now, Node 1 at the top of the stack, so visit node 1 and pop it from the stack and put all of its adjacent nodes which are not visited in the stack.



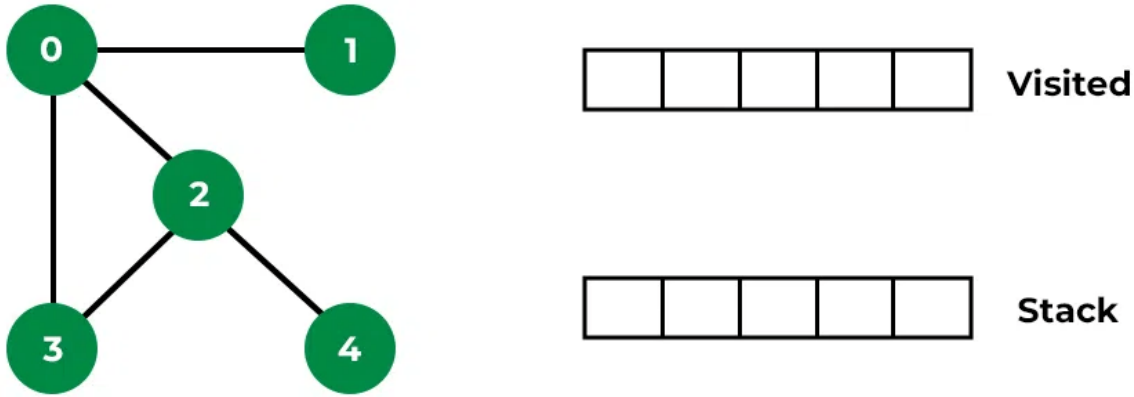
Example of Depth-First Search IV

- Step 4: Now, Node 2 at the top of the stack, so visit node 2 and pop it from the stack and put all of its adjacent nodes which are not visited (i.e, 3, 4) in the stack.



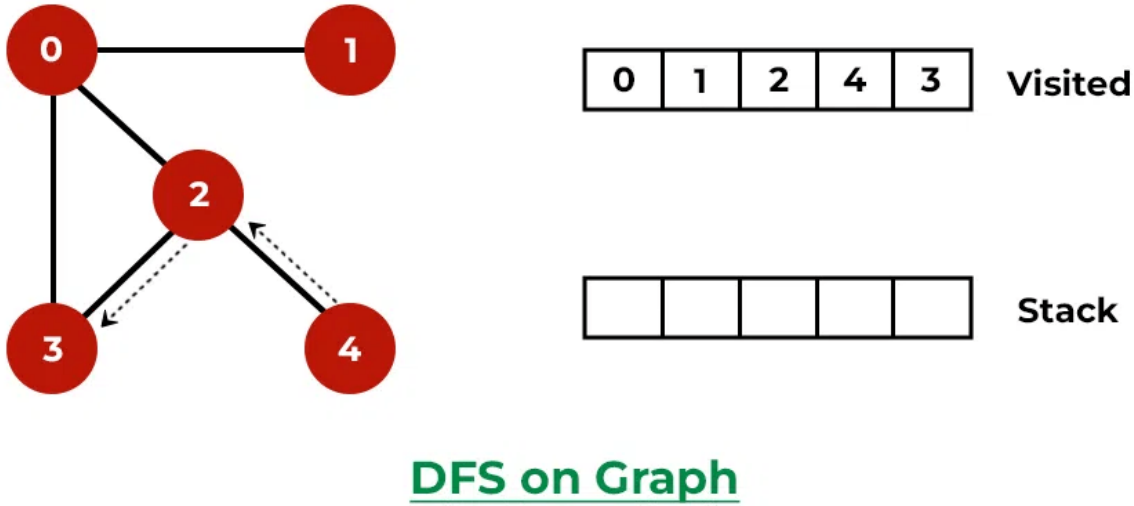
Example of Depth-First Search V

- Step 5: Now, Node 4 at the top of the stack, so visit node 4 and pop it from the stack and put all of its adjacent nodes which are not visited in the stack.



Example of Depth-First Search VI

- Step 6: Now, Node 3 at the top of the stack, so visit node 3 and pop it from the stack and put all of its adjacent nodes which are not visited in the stack.



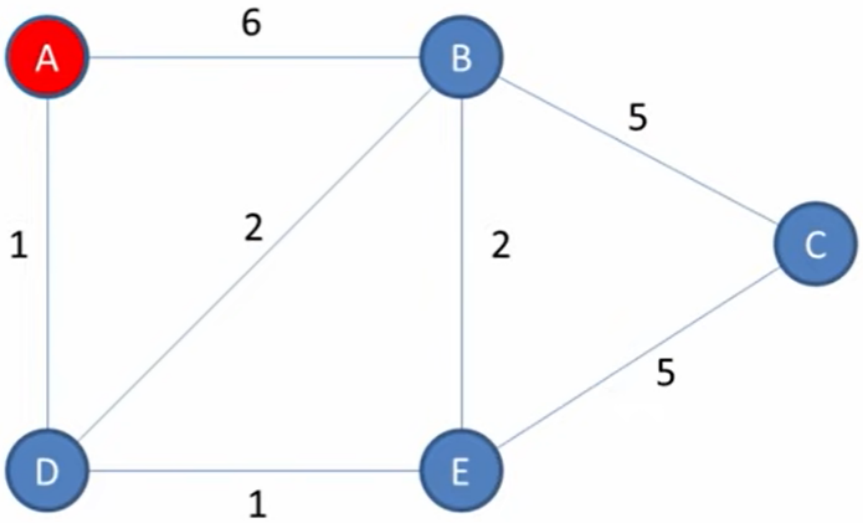
Example of Depth-First Search VII

- Now, Stack becomes empty, which means we have visited all the nodes and our DFS traversal ends.

Implementing Dijkstra's Algorithm

- Purpose: Finds the shortest path from a single source node to all other nodes in a weighted graph.
- Mechanism: Uses a priority queue to select the vertex with the minimum distance.
- Application: Widely used in network routing protocols, mapping services, and as a subroutine in other graph algorithms.

Consider the start vertex, A
Distance to A from A = 0
Distances to all other vertices from A are unknown, therefore ∞ (infinity)



Vertex	Shortest distance from A	Previous vertex
A		
B		
C		
D		
E		

Visited = [ ] Unvisited = [A, B, C, D, E]

Figure: Dijkstra

Questions & Answers

Thank you for your attention!

Feel free to ask any questions or share your thoughts.