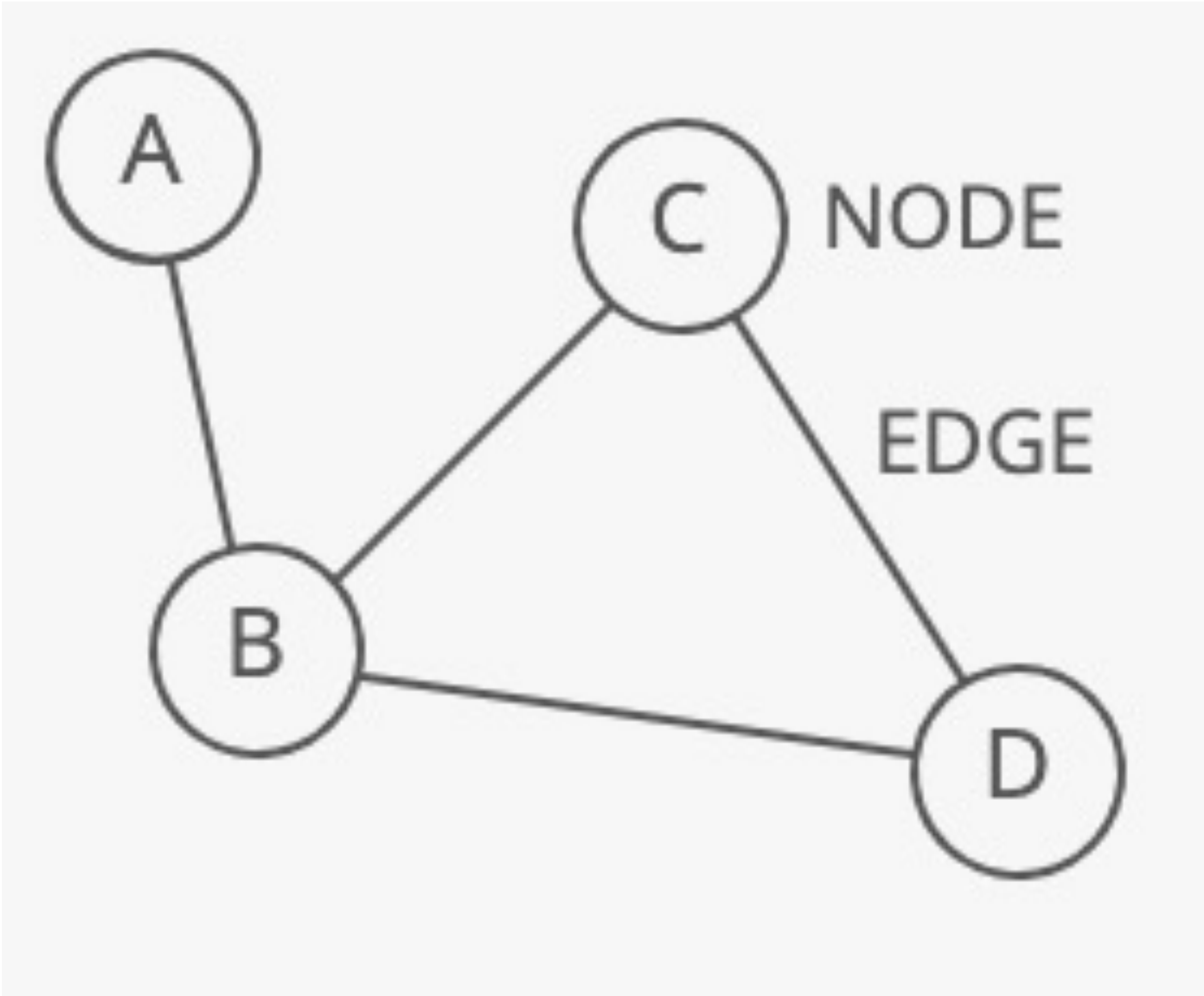
Algorithms and Data Structures

Graph

Part I: Graph Representations

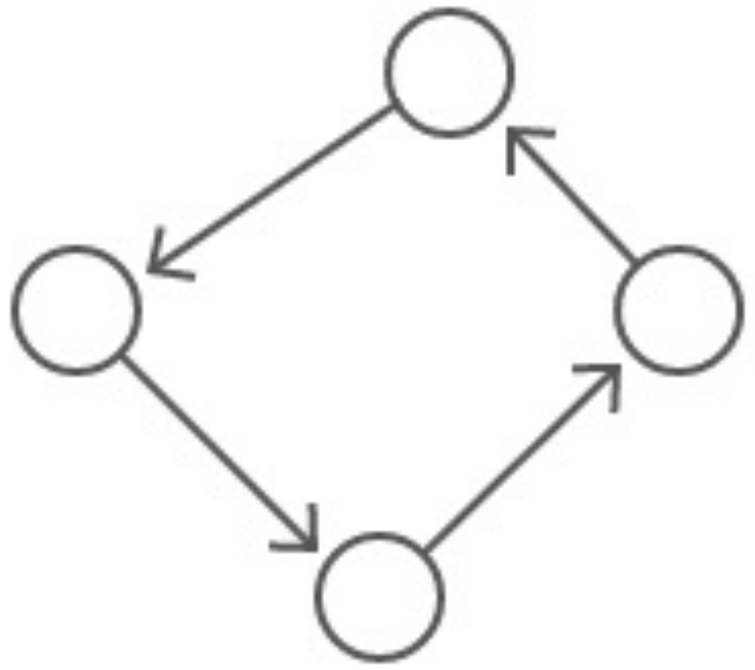
What is a Graph

- A **graph** organizes items in an interconnected network.
- Each item is a **node** (or **vertex**).
- Nodes are connected by **edges.**

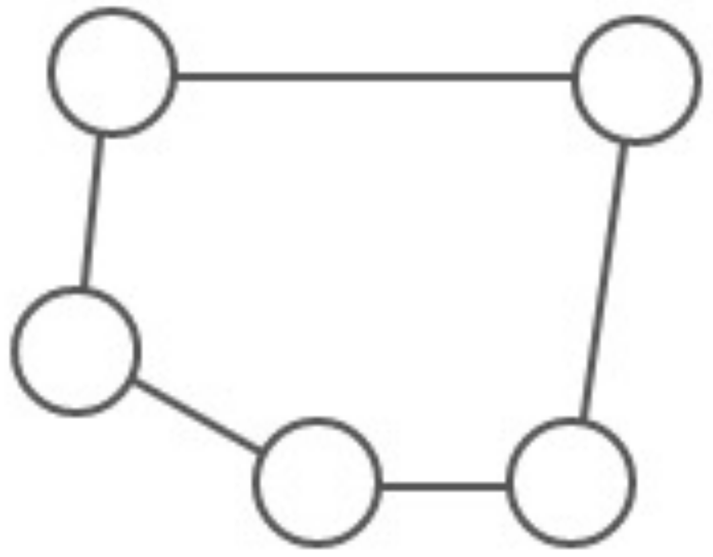


Directed?

- In **directed** graphs, edges point from the node at one end to the node at the other end. In **undirected** graphs, the edges simply connect the nodes at each end.



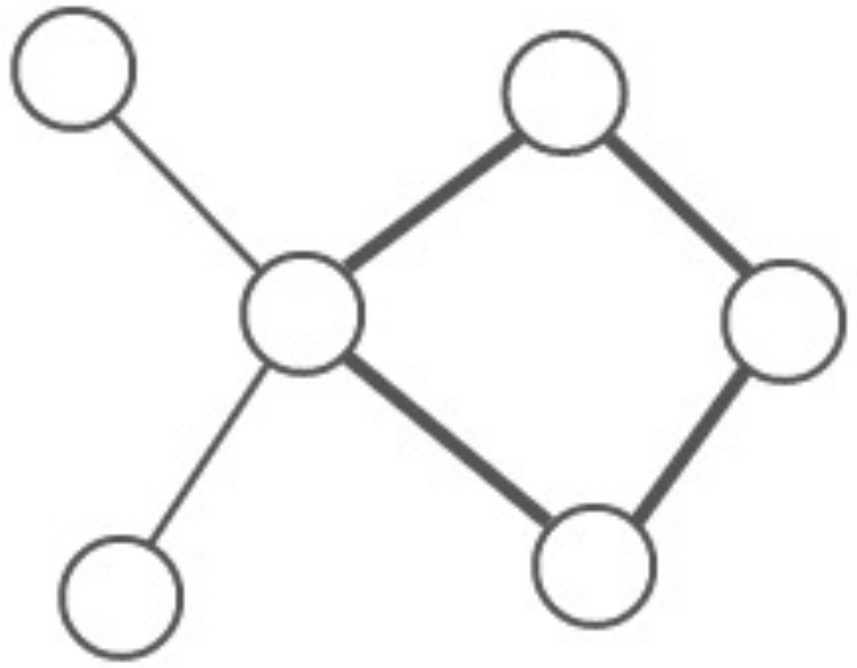
DIRECTED GRAPH



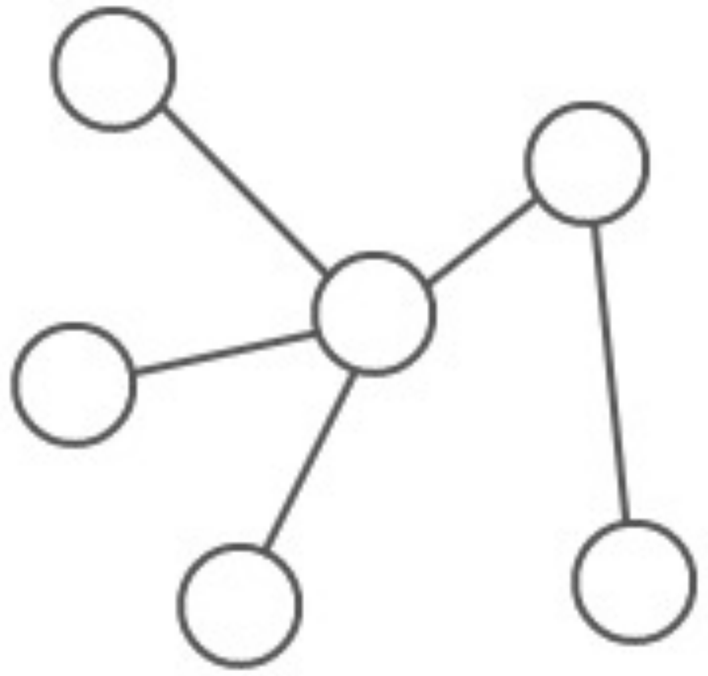
UNDIRECTED GRAPH

Cyclic?

- A graph is **cyclic** if it has a cycle—an unbroken series of nodes with no repeating nodes or edges that connects back to itself. Graphs without cycles are **acyclic**



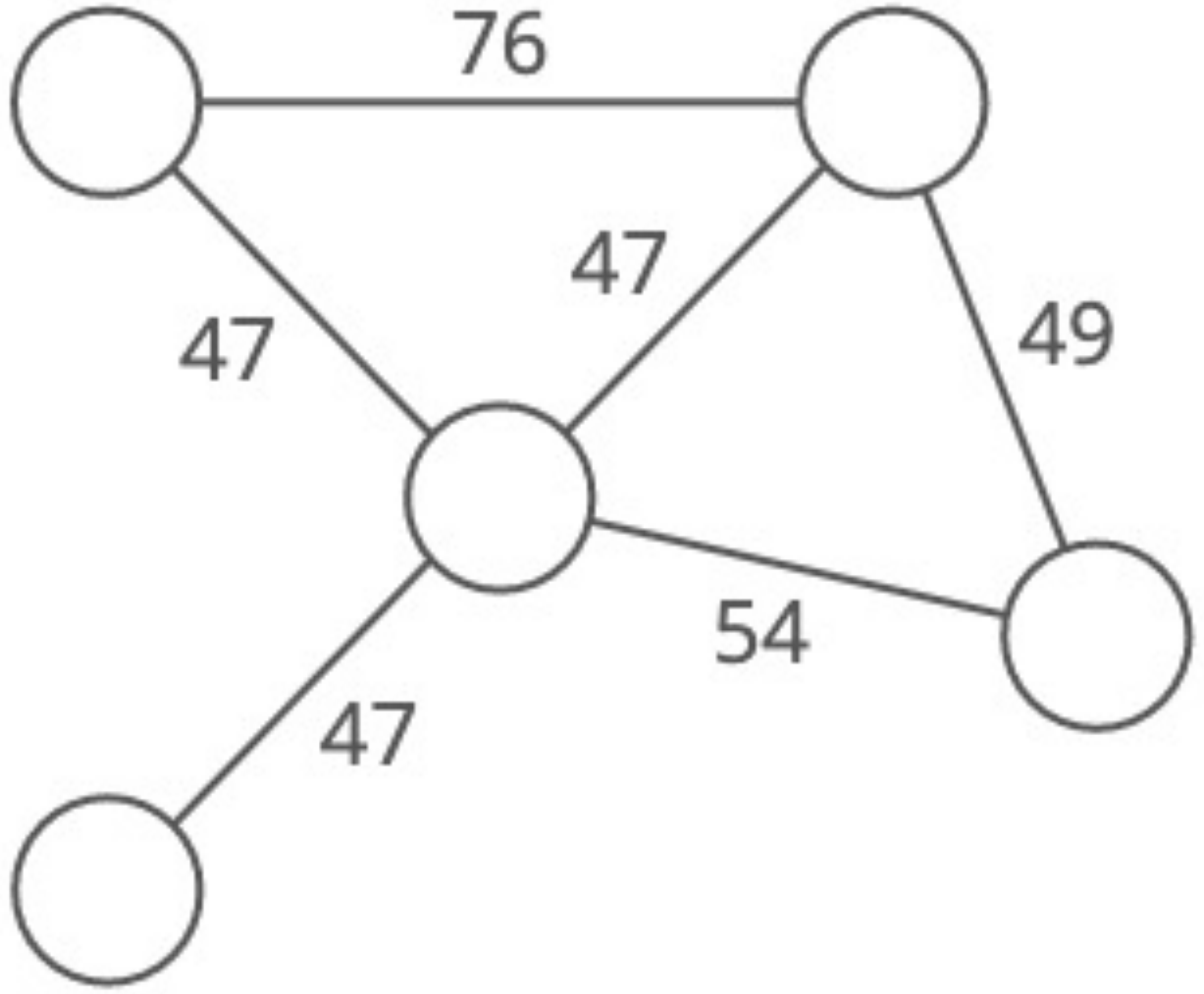
CYCLIC GRAPH



ACYCLIC GRAPH

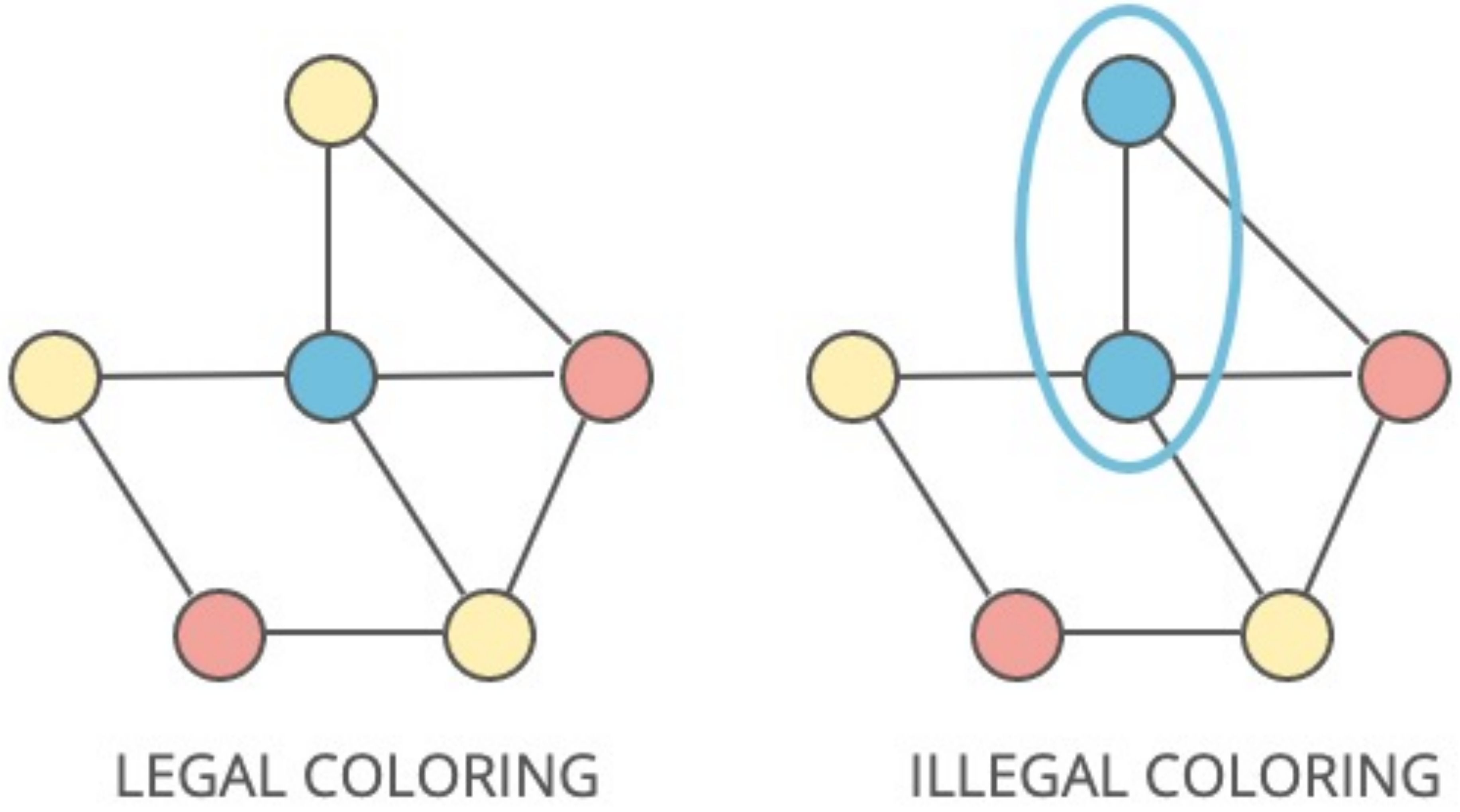
Weighted?

- If a graph is **weighted**, each edge has a "weight." The weight could, for example, represent the distance between two locations, or the cost or time it takes to travel between the locations.



Legal colored?

- A **graph coloring** is when you assign colors to each node in a graph. A **legal coloring** means no adjacent nodes have the same color.

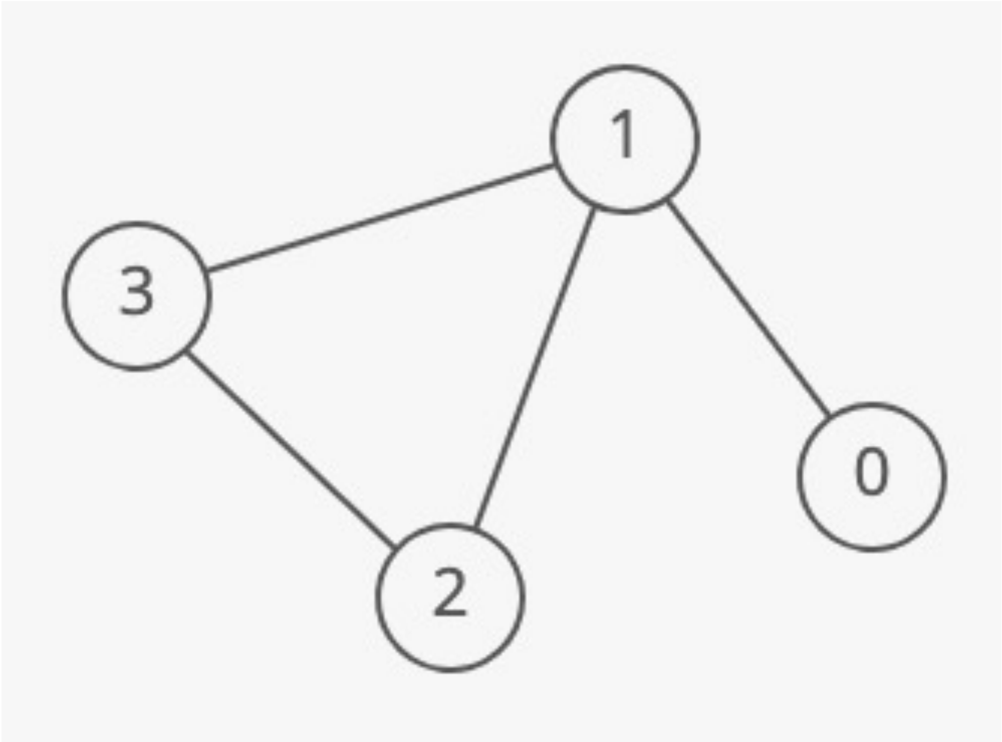


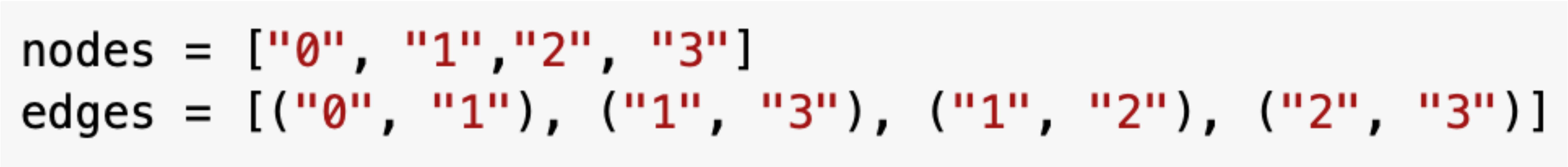
Graph Representation

- A graph can be represented in various approaches:
  - Edge list (and node list): A list of all the edges in the graph (and a list of all the nodes too since some nodes might not be connecting with other nodes)
  - Adjacency list (A list where the index represents the node and the value at that index is a list of the node's neighbors)
  - Adjacency matrix (A matrix of `0`s and `1`s indicating whether node `x` connects to node `y` (`0` means no, `1` means yes).
  - Incidence matrix and list

Edge List

- Edge list (and node list): A list of all the edges in the graph (and a list of all the nodes too since some nodes might not be connecting with other nodes)

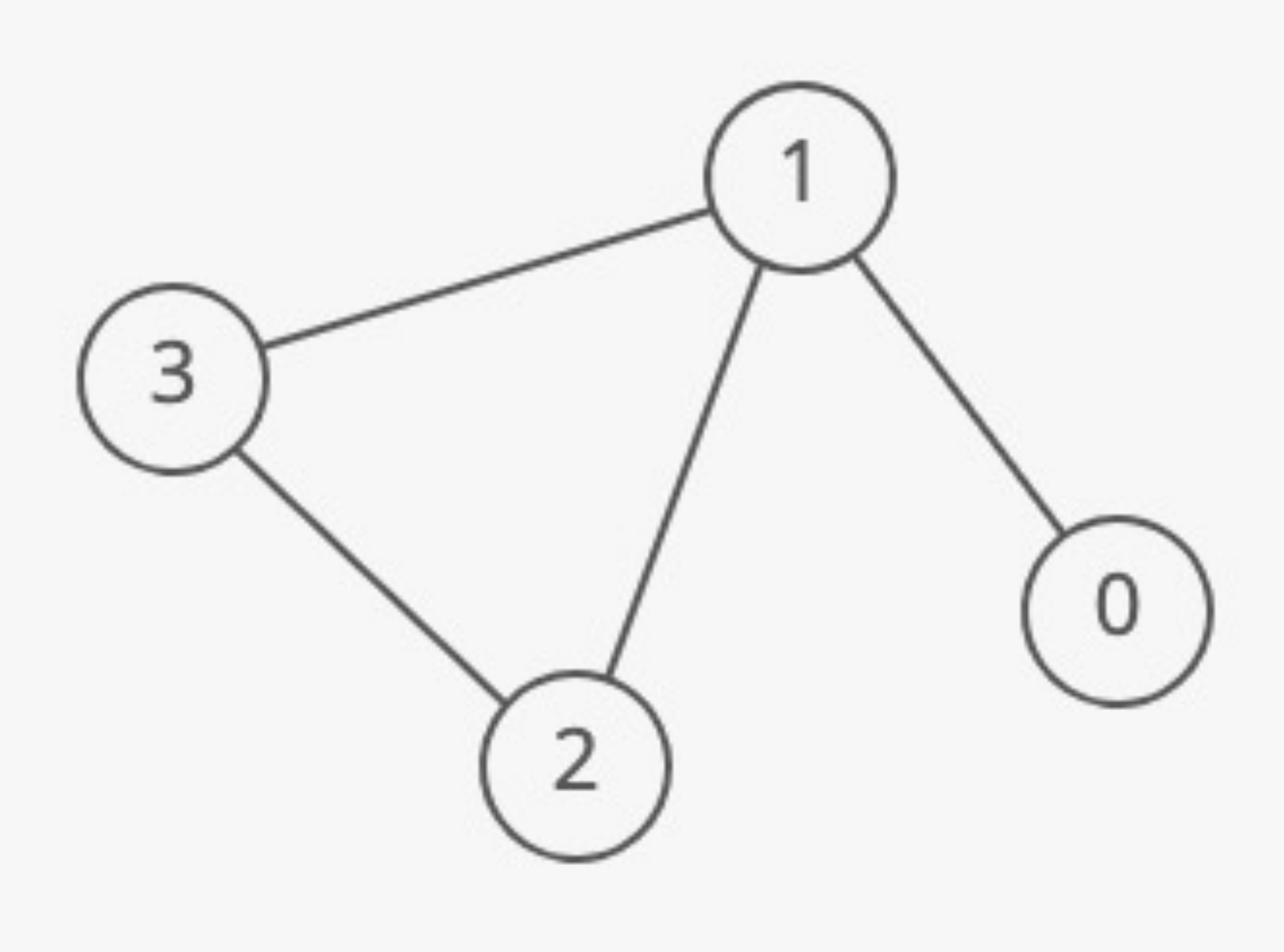


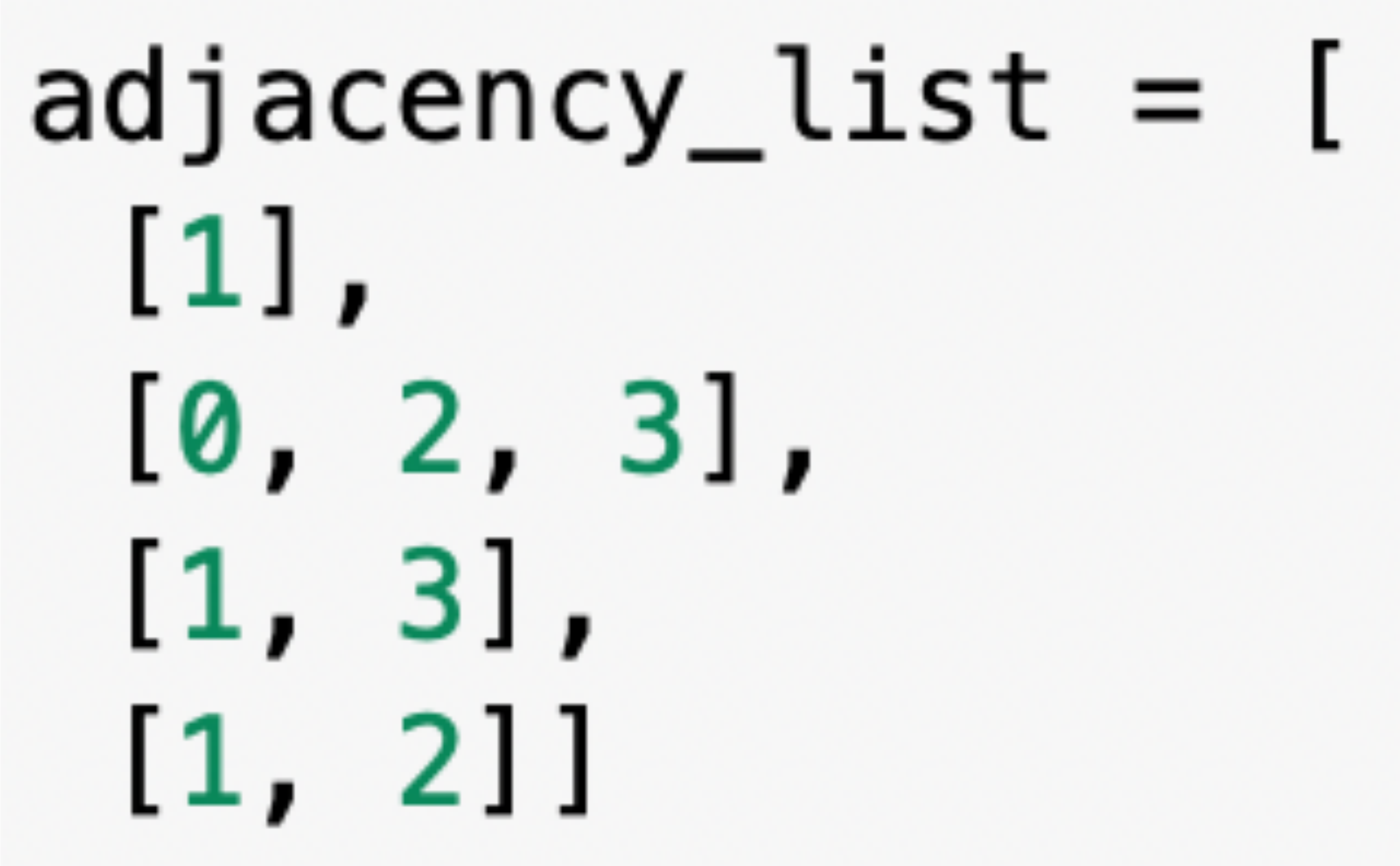


```
nodes = ["0", "1","2", "3"]
edges = [("0", "1"), ("1", "3"), ("1", "2"), ("2", "3")]
```

Adjacency List

- Adjacency list (A list where the index represents the node and the value at that index is a list of the node's neighbors)

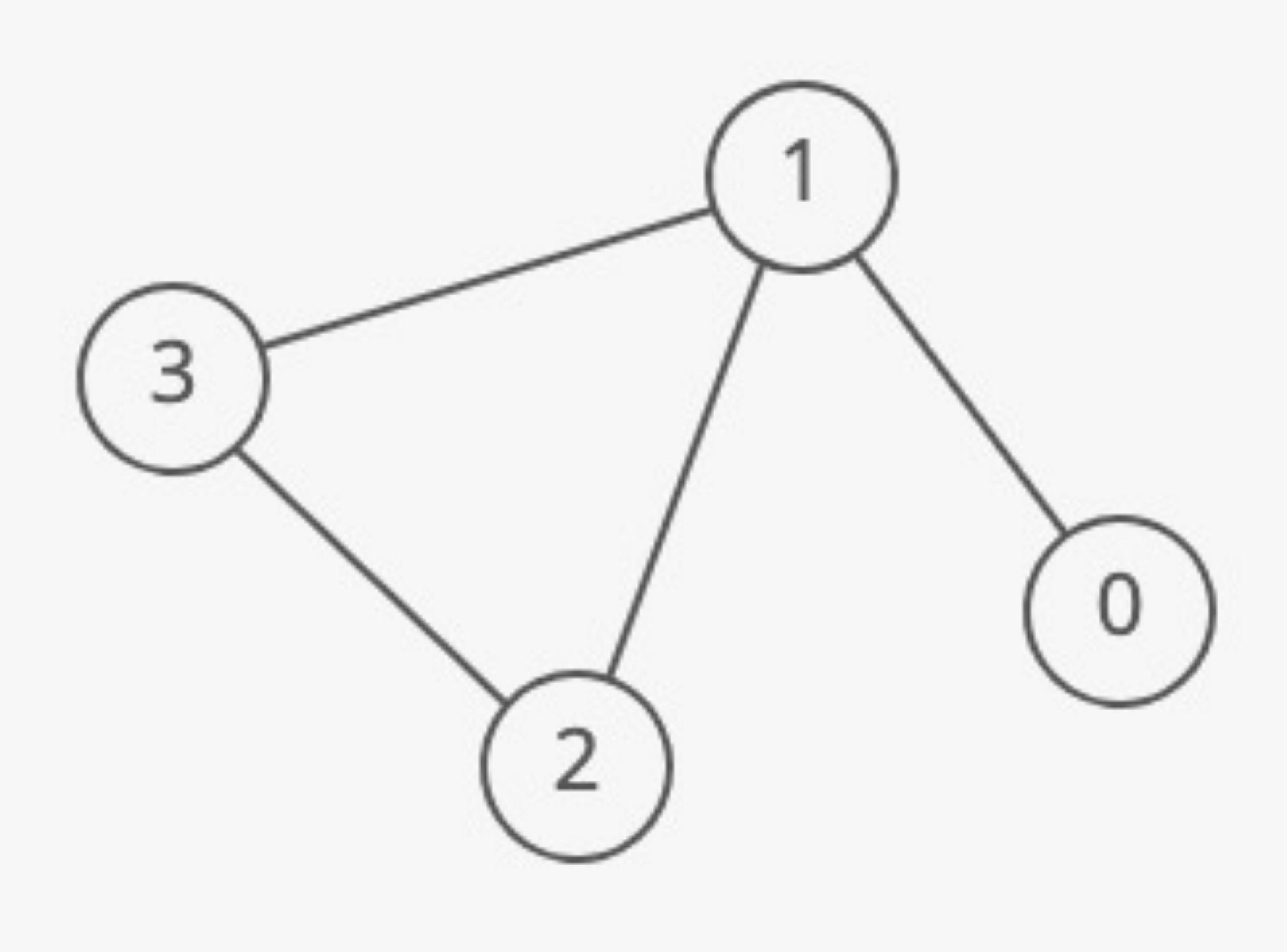


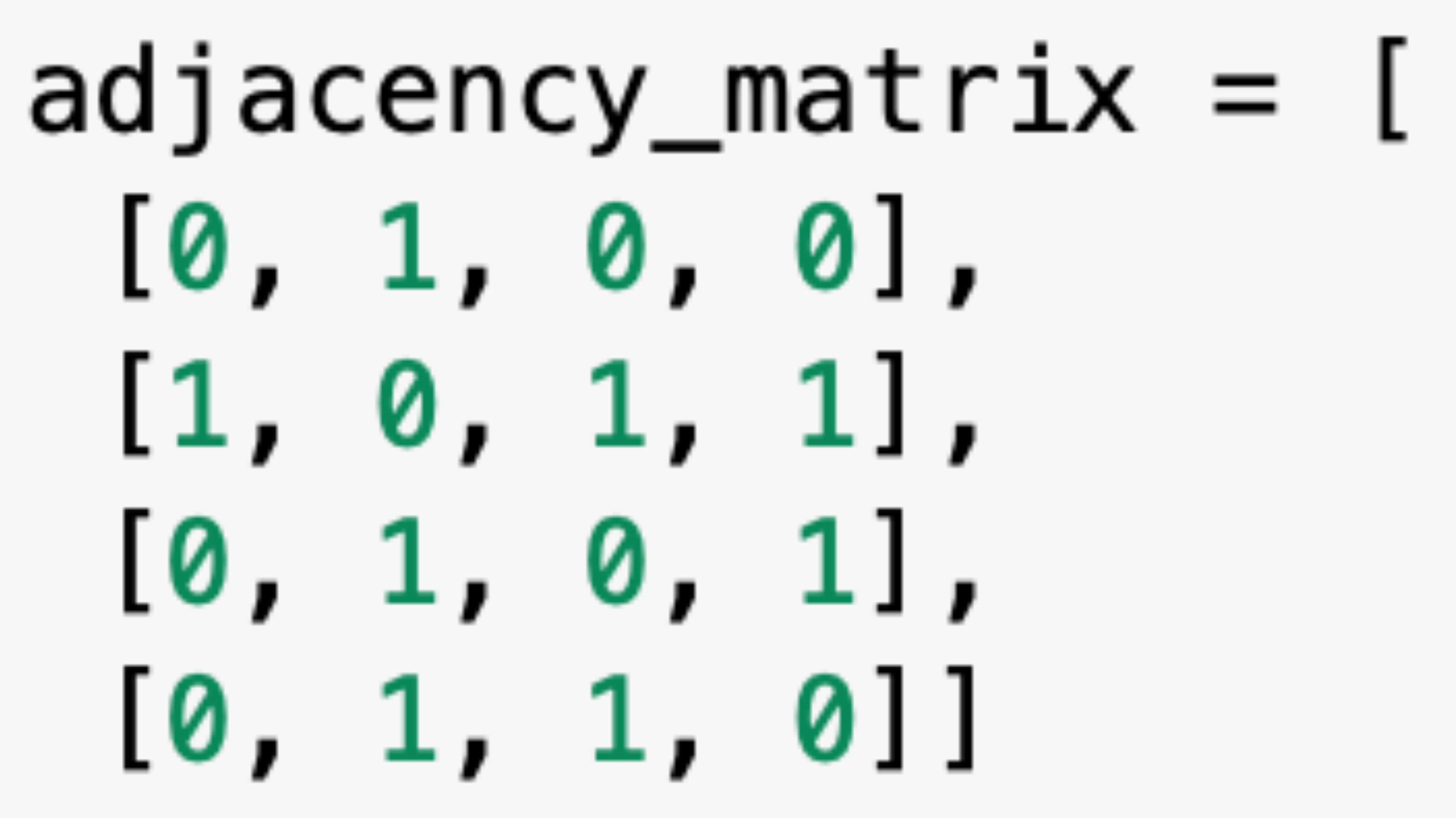


```
adjacency_list = [
  [1],
  [0, 2, 3],
  [1, 3],
  [1, 2]]
```

Adjacency Matrix

- Adjacency matrix (A matrix of `0`s and `1`s indicating whether node `x` connects to node `y` (`0` means no, `1` means yes).

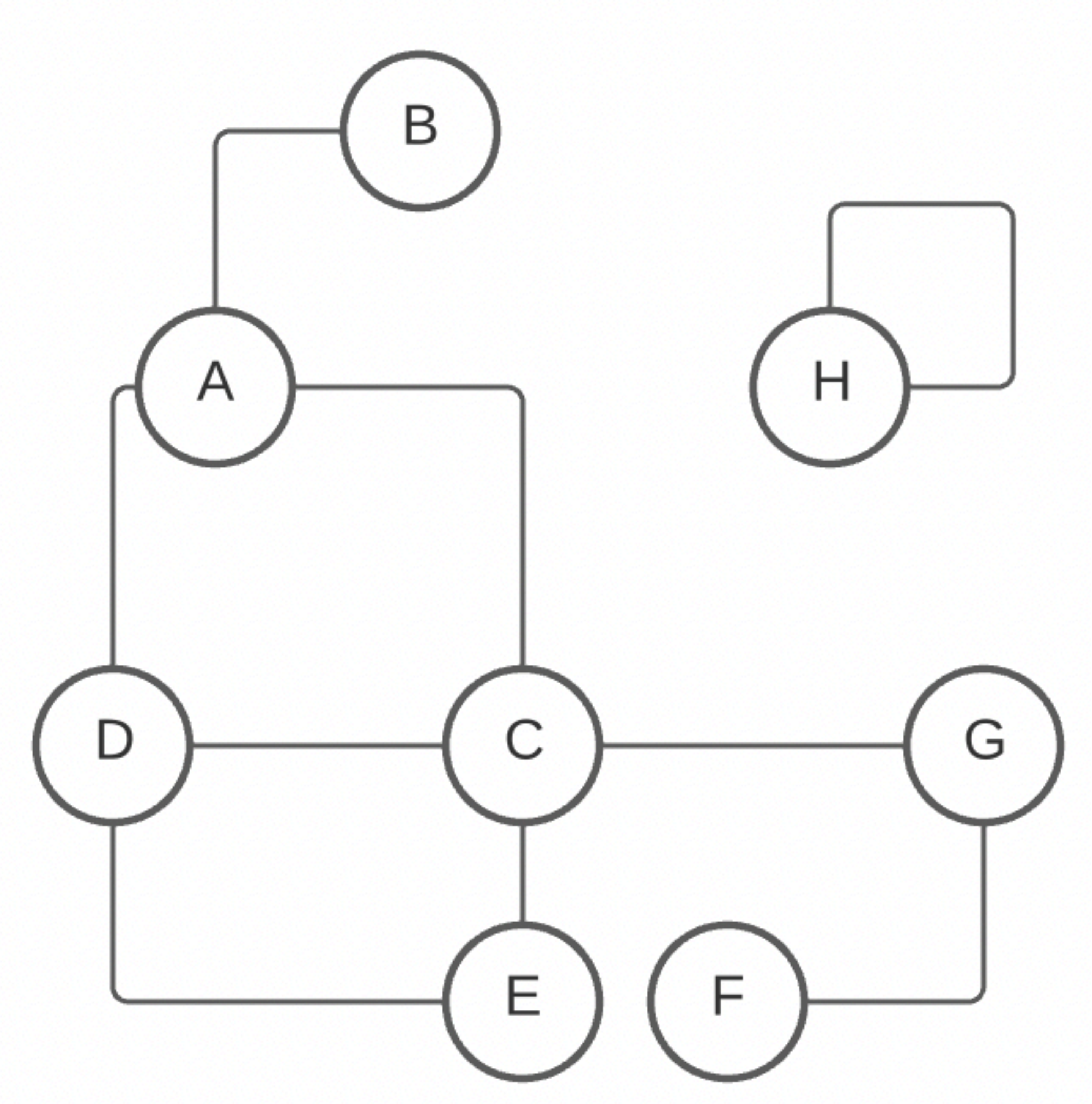




```
adjacency_matrix = [
 [0, 1, 0, 0],
 [1, 0, 1, 1],
 [0, 1, 0, 1],
 [0, 1, 1, 0]]
```

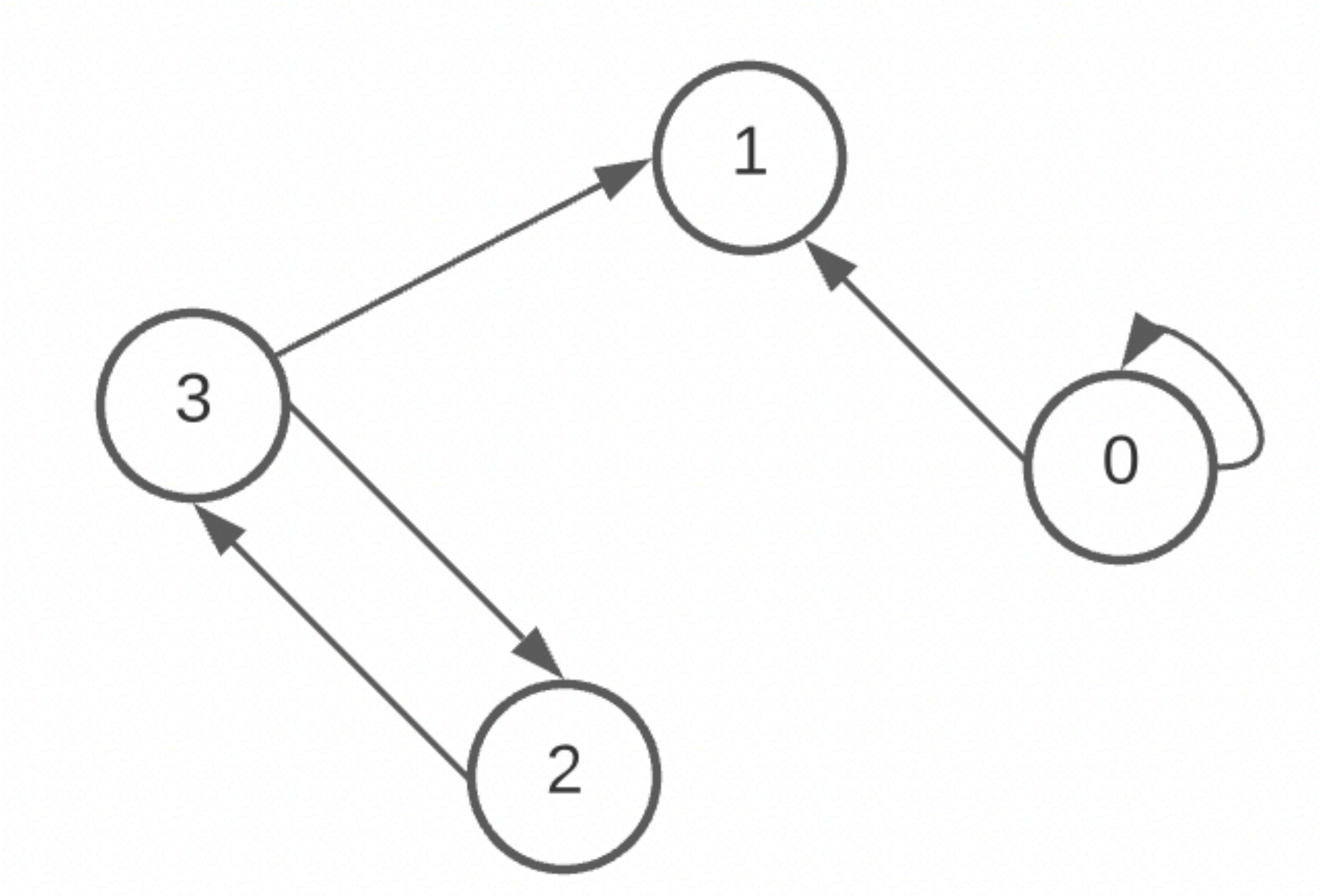
Exercise

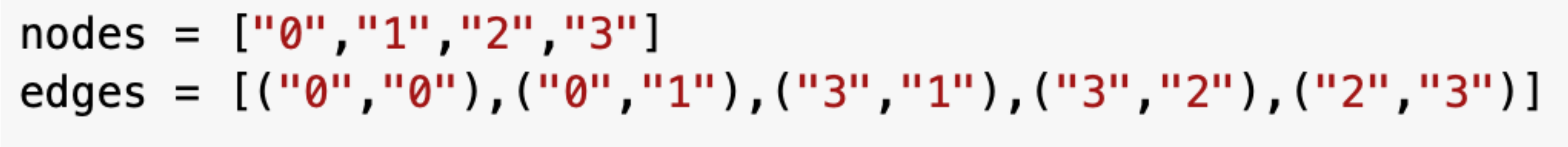
- Given the graph below, use the Edge/Node list, adjacency list, and adjacency matrix to represent it.



Edge List

- Edge list (and node list): A list of all the edges in the graph (and a list of all the nodes too since some nodes might not be connecting with other nodes)

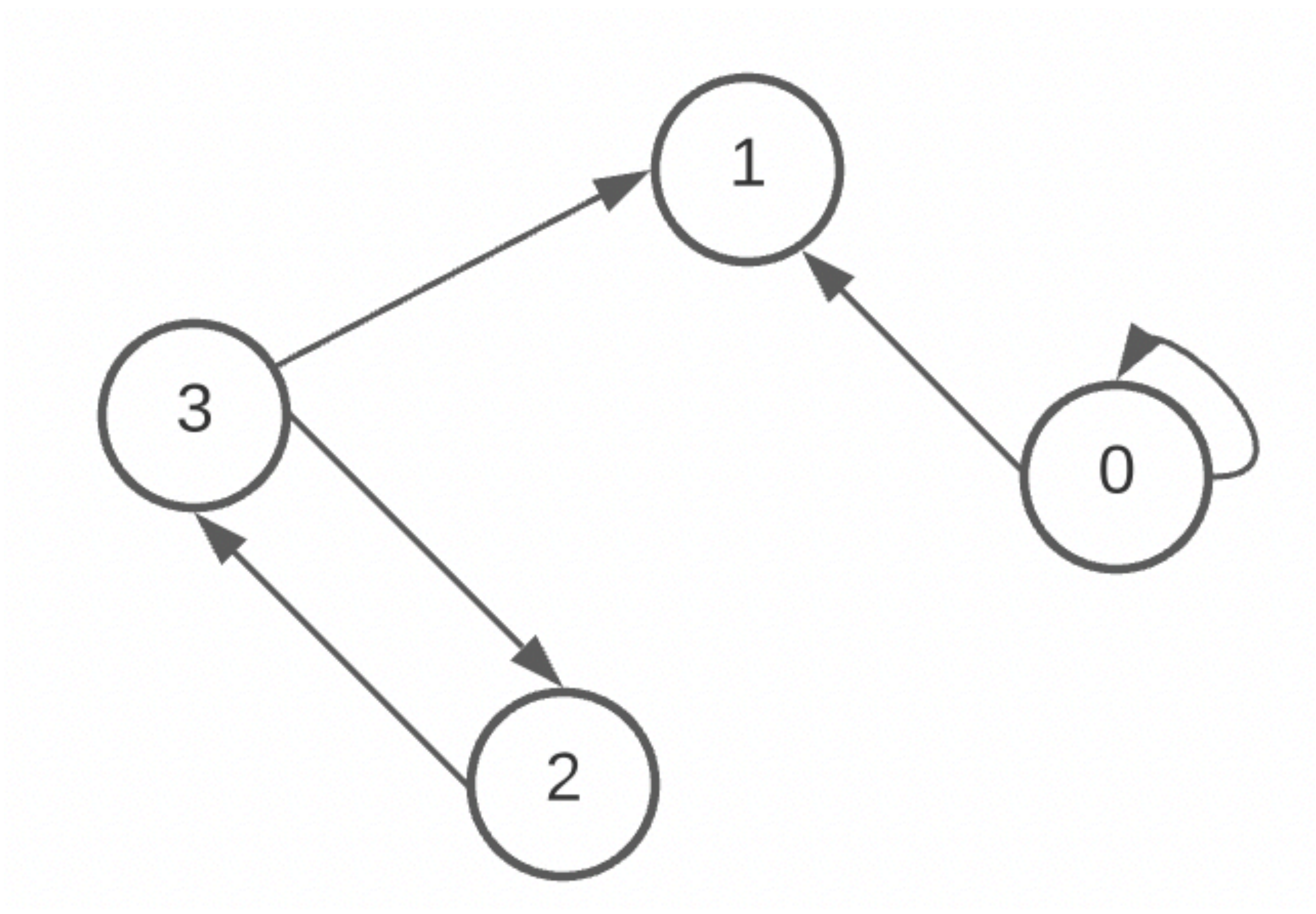


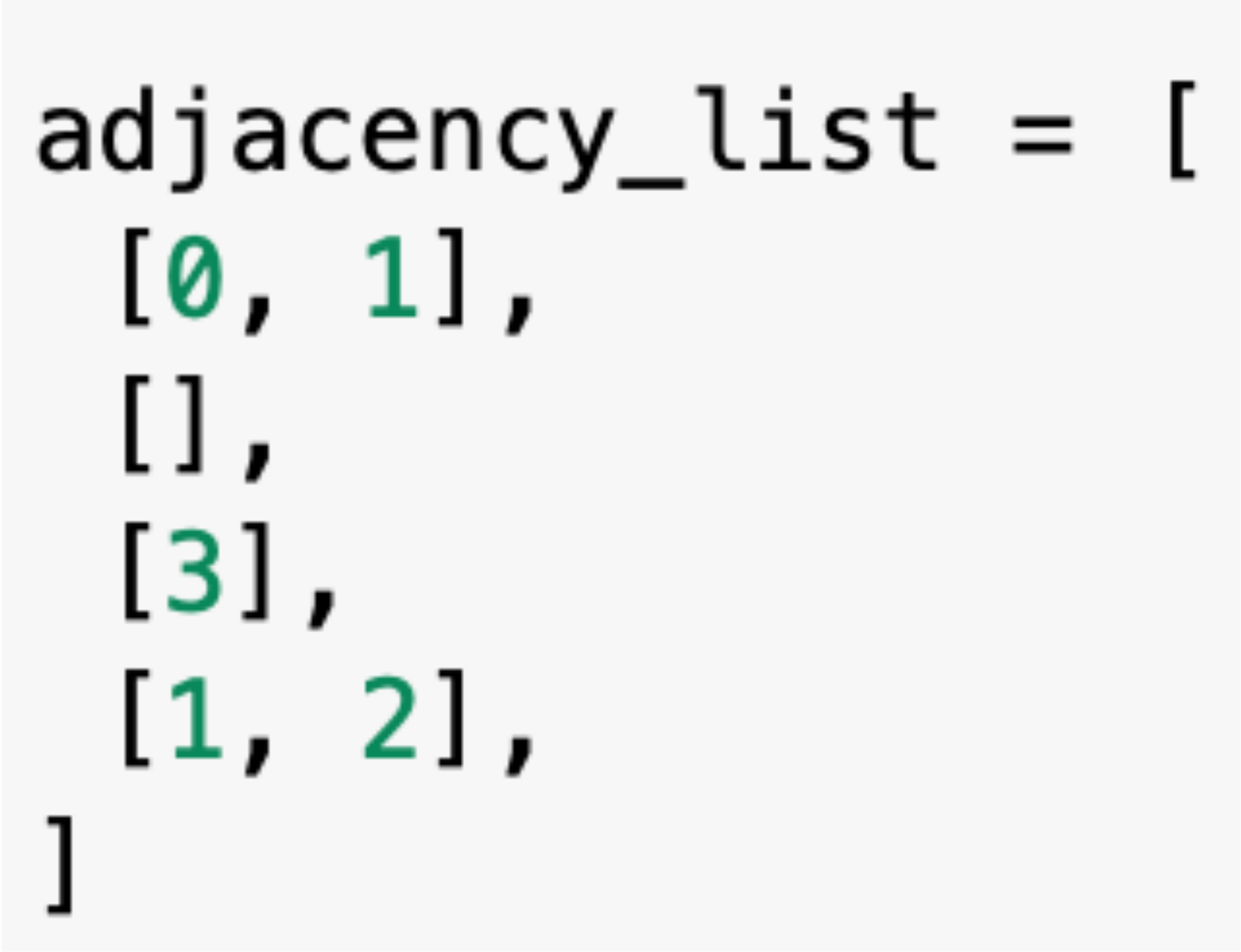


```
nodes = ["0","1","2","3"]
edges = [("0","0"),("0","1"),("3","1"),("3","2"),("2","3")]
```

Adjacency List

- Adjacency list (A list where the index represents the node and the value at that index is a list of the node's neighbors)

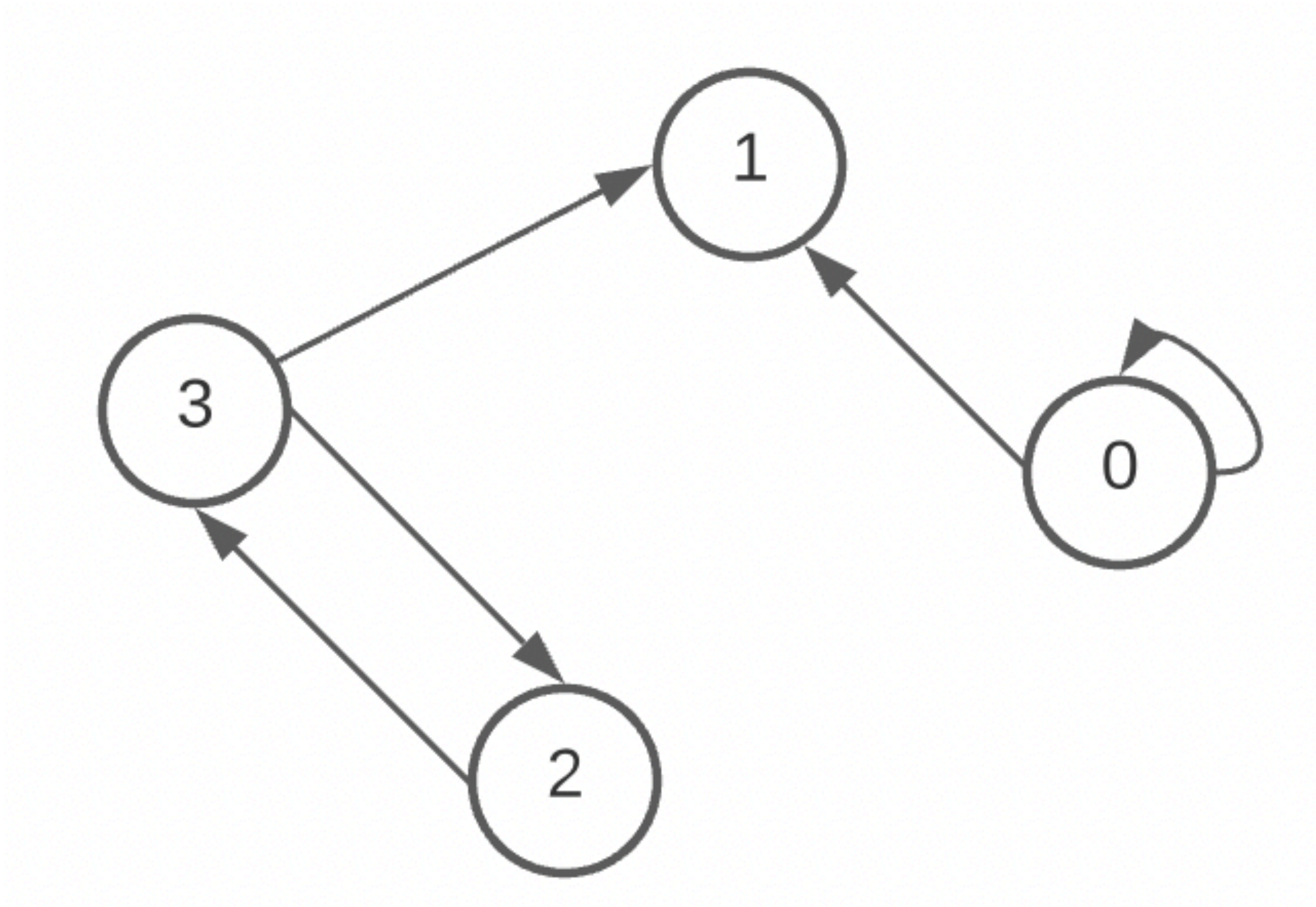


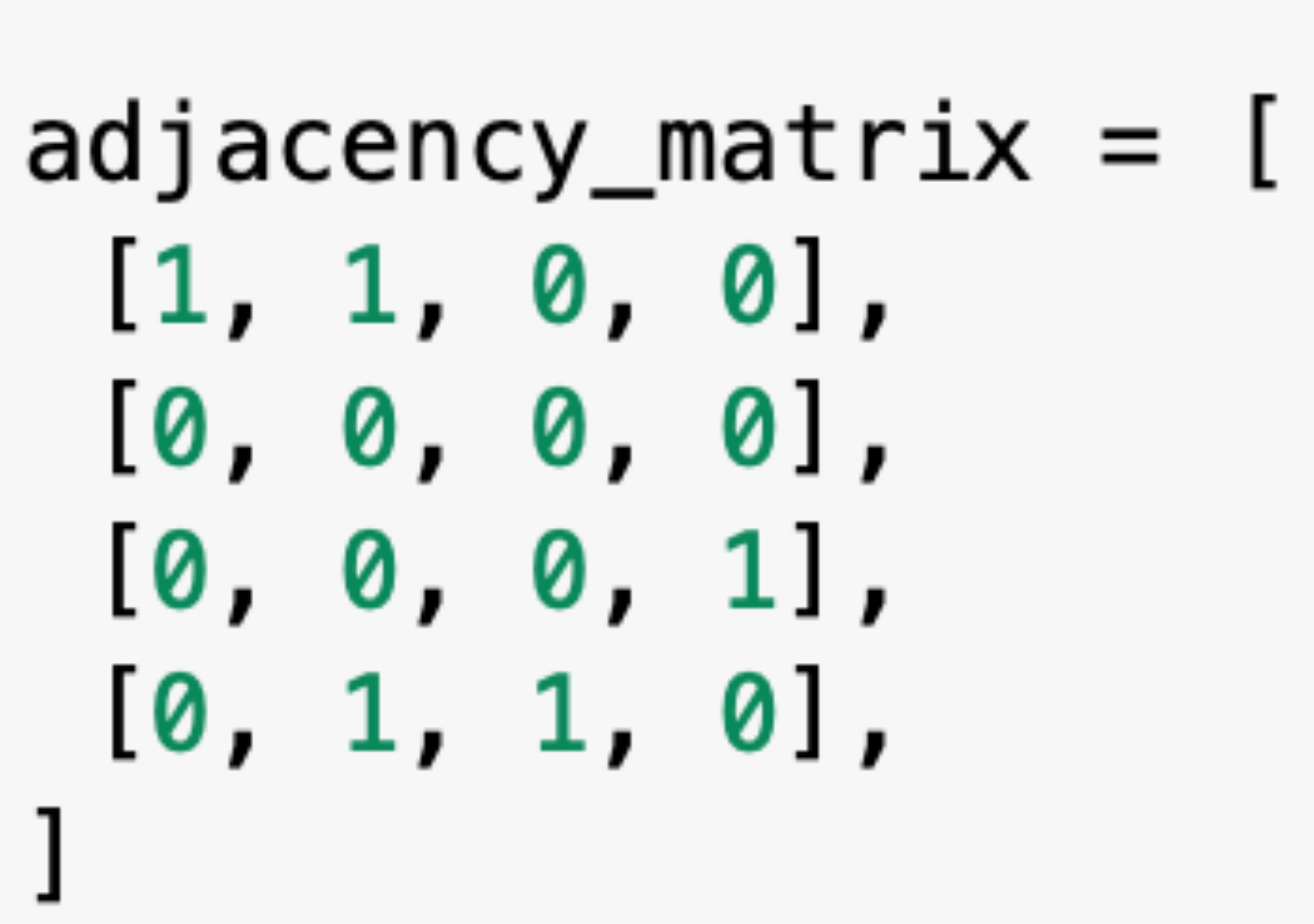


```
adjacency_list = [
 [0, 1],
 [],
 [3],
 [1, 2],
]
```

Adjacency Matrix

- Adjacency matrix (A matrix of `0`s and `1`s indicating whether node `x` connects to node `y` (`0` means no, `1` means yes).

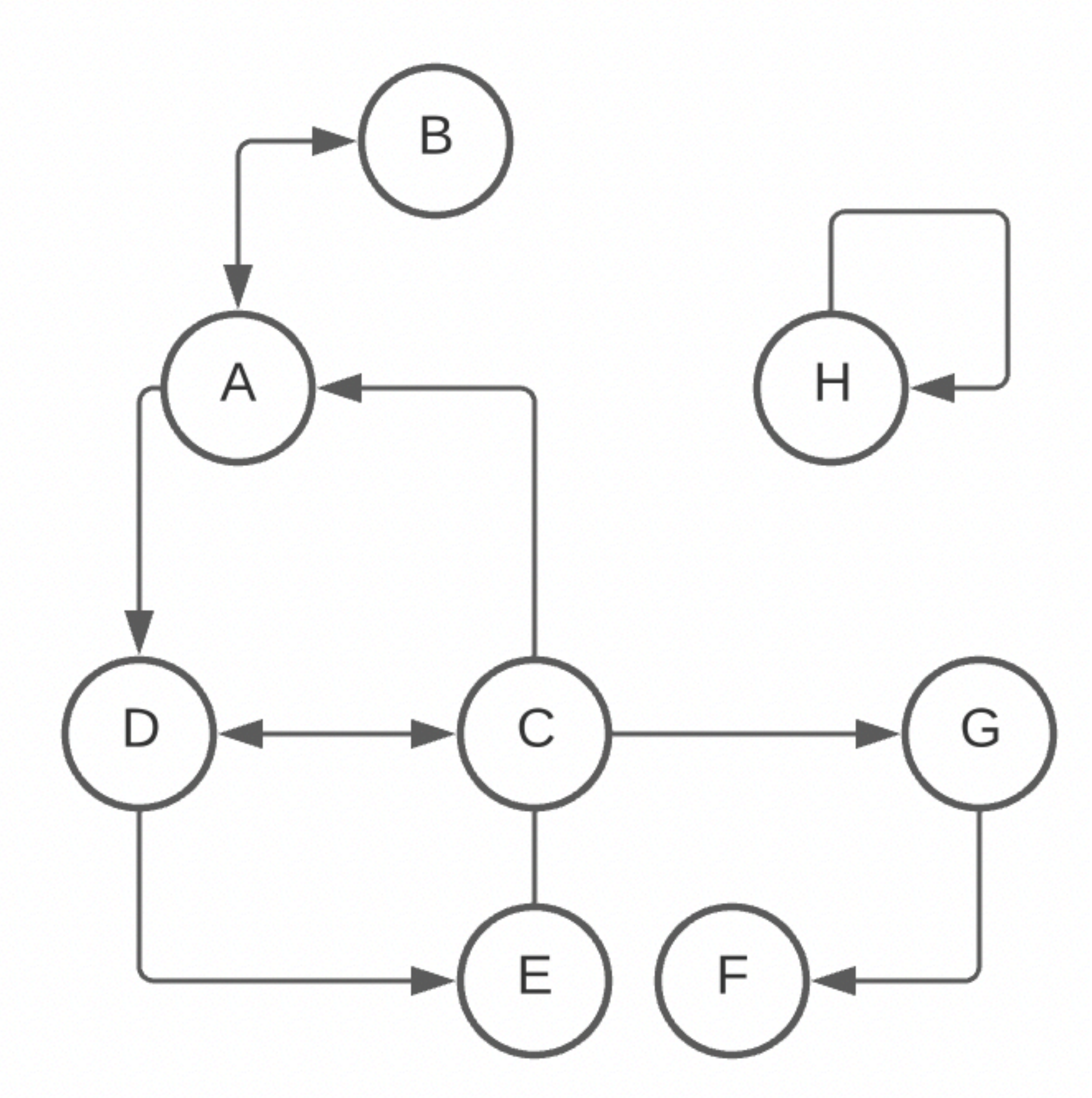




```
adjacency_matrix = [
 [1, 1, 0, 0],
 [0, 0, 0, 0],
 [0, 0, 0, 1],
 [0, 1, 1, 0],
]
```

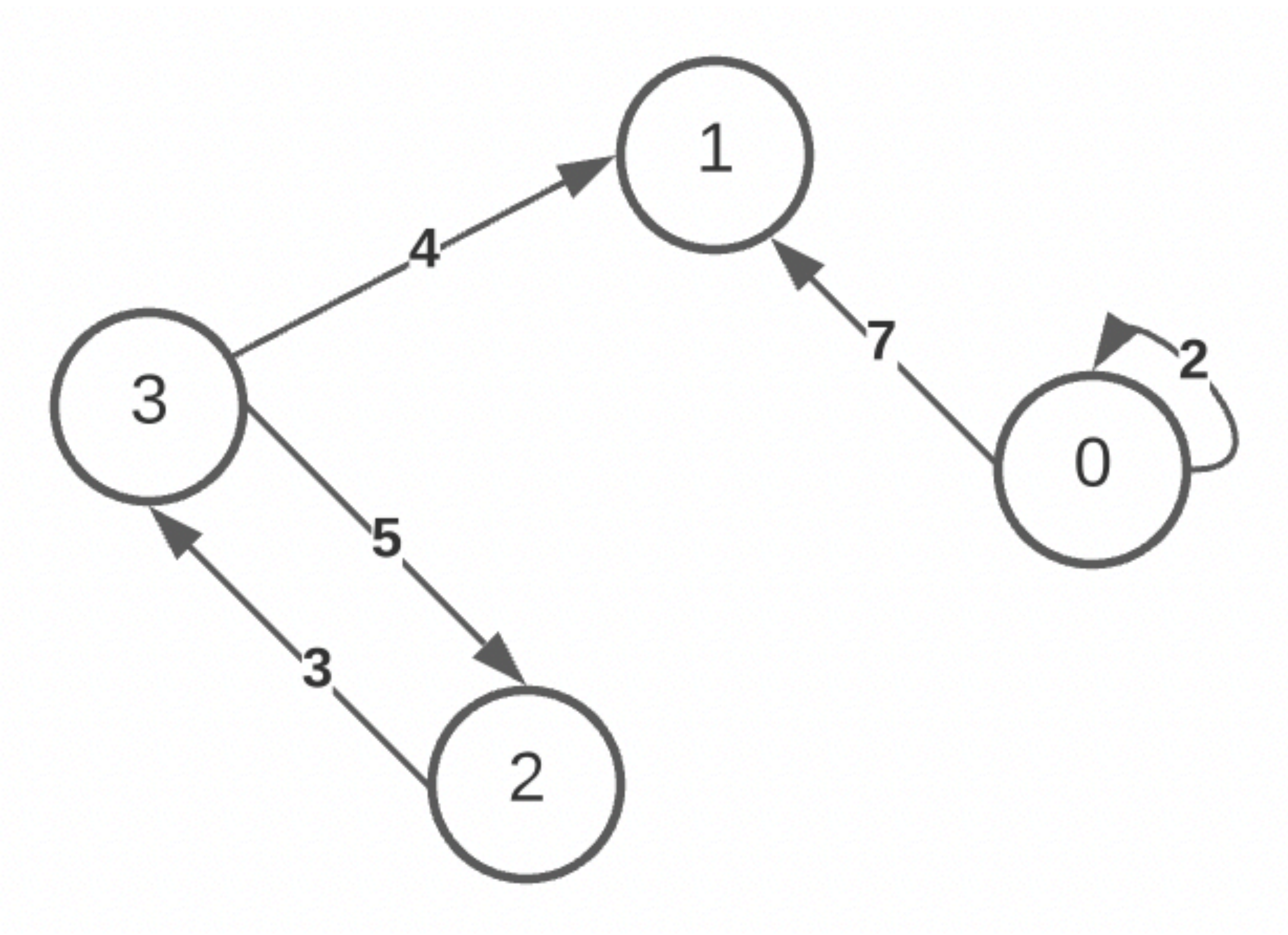
Exercise

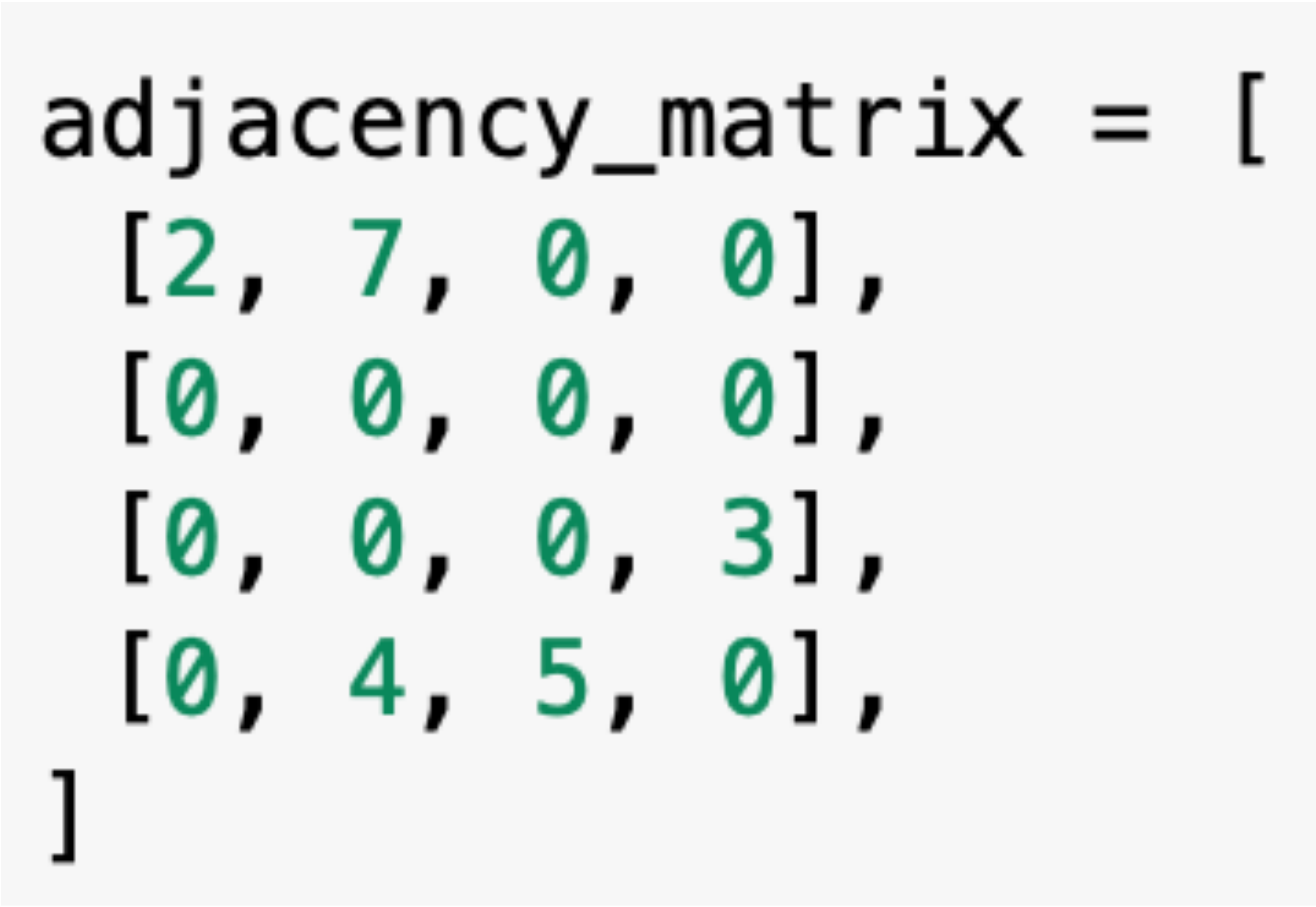
- Given the graph below, use the Edge/Node list, adjacency list, and adjacency matrix to represent it.



Adjacency Matrix (Weighted)

- Adjacency matrix (A matrix of `0`s and `n`s indicating whether node `x` connects to node `y` (`0` means no, `n` means weight).





```
adjacency_matrix = [
  [2, 7, 0, 0],
  [0, 0, 0, 0],
  [0, 0, 0, 3],
  [0, 4, 5, 0],
]
```

Thank you!