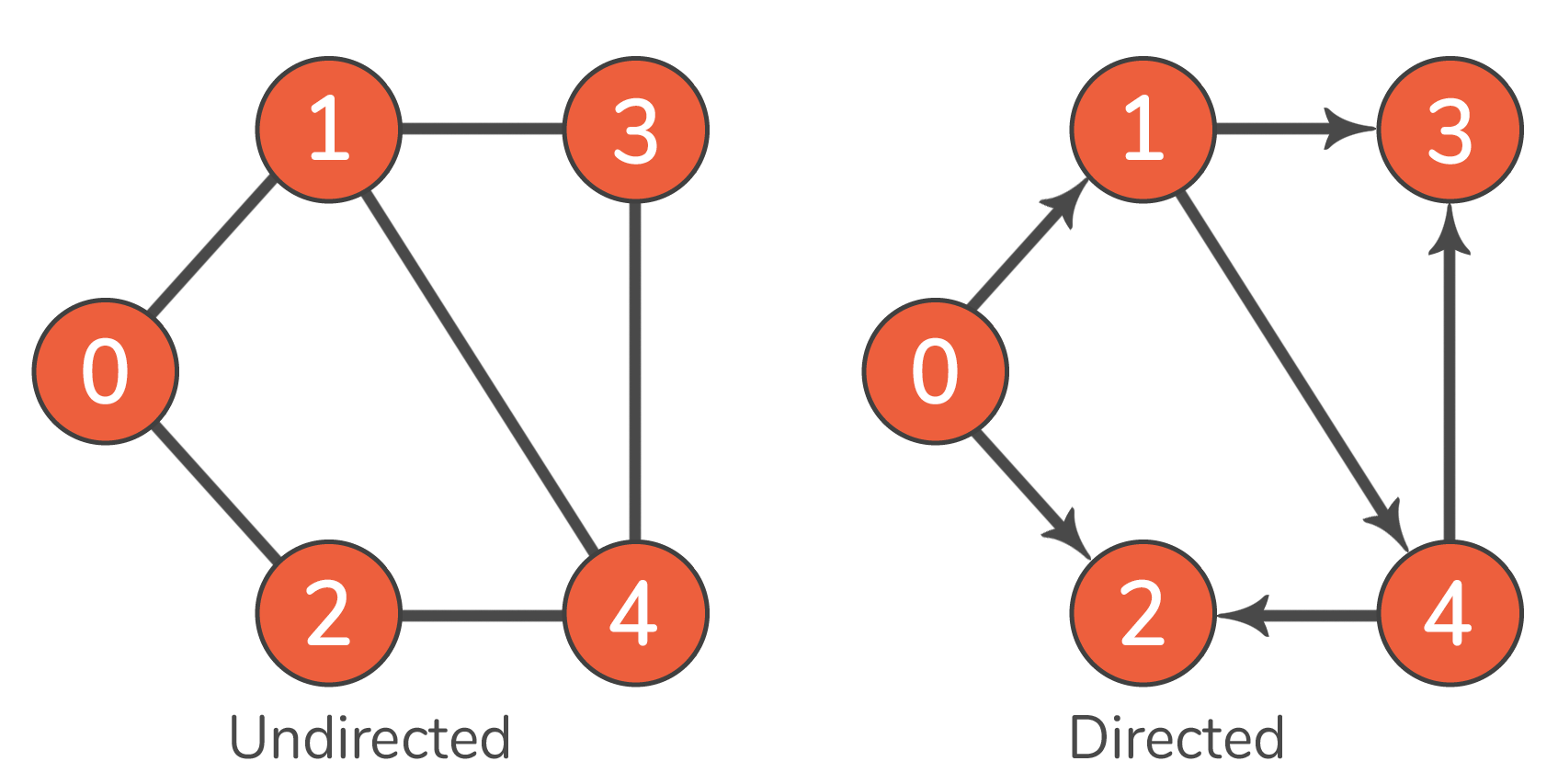
Graph Representation using an Adjacency List [CodeStudio](https://www.codingninjas.com/studio/problems/create-a-graph-and-print-it_1214551?leftPanelTabValue=PROBLEM)

This program demonstrates graph representation using an adjacency list. The **Graph** class encapsulates the logic for initializing the number of nodes, adding edges, and printing the adjacency list. The adjacency list offers a flexible way to store and traverse graph connections.

Example:



Output:

Undirected Graph

0: {1, 2}

1: {0, 3, 4}

2: {0, 4}

3: {1, 4}

4: {2, 3}

Directed Graph

0: {1, 2}

1: {3, 4}

4: {2, 3}

**Graph Class**

Members:

* **int nodes**: Number of nodes in the graph.
* **unordered\_map<int, list<int>> adjacencyList**: Adjacency list representation.

Methods:

1. **Graph(int nodes)**
   * **Purpose:** Initializes the graph with a specified number of nodes.
   * **Complexities:**
     + **Time: O(1)**
     + **Space: O(1)**
2. **void addEdge(vector<vector<int>> &edges, bool isDirected)**
   * **Purpose:** Adds edges to the graph.
   * **Complexities:**
     + **Time: O(E), where E is the number of edges.**
     + **Space: O(E)**
3. **void printGraph()**
   * **Purpose:** Prints the adjacency list of the graph.
   * **Complexities:**
     + **Time: O(V + E), where V is the number of vertices and E is the number of edges.**
     + **Space: O(1)**