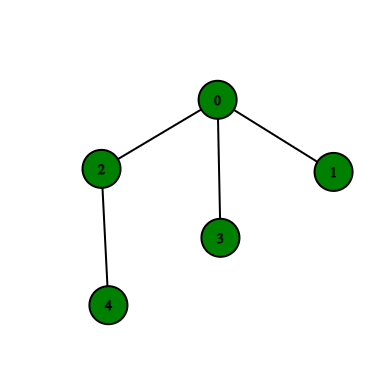
Depth First Search Traversal of Graph [CodeStudio](https://www.codingninjas.com/studio/problems/dfs-traversal_630462?leftPanelTabValue=PROBLEM)

You are given a undirected graph. Perform a Depth First Traversal of the graph.

Example:

 Output: {0, 2, 4, 3, 1}

**Explanation**:

**addEdge Function:**

* **Purpose:**
  + Populates the graph's adjacency list based on the provided edge list.
* **Explanation:**
  + Iterates through each edge in the **edges** vector.
  + For each edge, extracts the source vertex **u** and iterates over the connected vertices.
  + Adds an edge from **u** to **v** in the adjacency list.
  + If the graph is undirected, adds an edge from **v** to **u** as well.
* **Time Complexity:**
  + **O(E), where E is the number of edges in the input vector.**
* **Space Complexity:**
  + **O(E), where E is the number of edges. Each edge results in the creation of one or two entries in the adjacency list.**

**Approach 1: Function to perform Depth-First Search (DFS) on the entire graph**

* **Purpose:**
  + Performs DFS traversal on the entire graph, handling disconnected components.
* **Explanation:**
  + Stores the DFS traversal result.
  + Marks nodes as visited.
  + Iterates through each node in the graph, starting DFS from unvisited nodes.
  + Constructs DFS traversal components and adds them to the overall result.
* **Time Complexity:**
  + **O(V + E), where V is the number of vertices and E is the number of edges. Accounts for the traversal of all vertices and edges, even in the presence of disconnected components.**
* **Space Complexity:**
  + **O(V + E), where V is the number of vertices and E is the number of edges. This includes space for the visited map, DFS traversal result vector, and recursion stack.**

**Overall Time and Space Complexity:**

* **Overall Time Complexity:**
  + **Using addEdge: O(V + E) for both connected graphs and graphs with disconnected components.**
  + **Without addEdge (only DFS): O(V + E) for the DFS traversal alone, considering all vertices and edges.**
* **Overall Space Complexity:**
  + **O(V + E), considering the adjacency list, visited map, and auxiliary data structures in the dfsOfGraph function.**

**Conclusion:**

* The program efficiently performs DFS traversal on both connected graphs and graphs with disconnected components.
* The time complexity remains O(V + E) even when dealing with disconnected components, primarily due to the **addEdge** function. If DFS alone is considered, the time complexity is O(V + E).