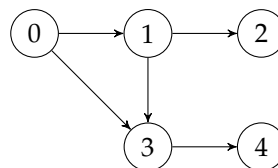


1. Tracing DFS

The following code implements depth-first search:

```
def dfs(G, node, visited):  
    visited[node] = True  
    print(node)  
    for neighbor in G[node]:  
        if not visited[neighbor]:  
            dfs(G, neighbor, visited)
```

Consider the graph below:



The neighbors of each node are explored in the order shown below:

0: [1, 3], 1: [2, 3], 2: [], 3: [4], 4: []

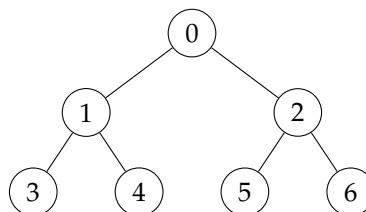
a. When DFS reaches node 1, which neighbor does it explore first? _____

b. Write the order in which the nodes are printed using the function above given that we start our dfs at node 0.

[DFS order: _____ → _____ → _____ → _____ → _____]

2. DFS on a Tree

Consider the following tree:



Suppose we run DFS starting at node 0. When a node has multiple children, we always explore the **left child first**.

a. Which node does DFS visit immediately after node 0? _____

b. Write the order in which DFS visits the nodes.

[DFS order: _____ → _____ → _____ → _____ → _____ → _____ → _____]

3. Tracing DFS Without a visited Array

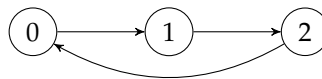
The following code implements depth-first search without keeping track of visited nodes:

```
def dfs(G, node):  
    print(node)  
    for neighbor in G[node]:  
        dfs(G, neighbor)
```

Consider the graph represent by this adjacency list:

0: [1], 1: [2], 2: [0]

The graph is drawn on the next page.



a. Write the first several nodes that are printed given that we start our dfs at node 0 in the function above.

[DFS order: _____ → _____ → _____ → _____ → _____ → ...]

b. What happens after DFS reaches node 2?

- (A) DFS stops.
- (B) DFS returns to node 1.
- (C) DFS goes back to node 0 and continues.
- (D) DFS skips node 0 because it was already visited.

c. The DFS never finishes. Why does this happen?