**Depth-First Search DFS**

**Iheb Gafsi**\*

INSAT Student

**Iheb.engineer@gmail.com**

**Definition:**

The depth-first search DFS is an algorithm used for traversing or searching a graph data structure, the “depth” in the DFS word refers to the order in which the nodes are explored. The DFS algorithm explores as far as possible along each branch before backtracking until all the nodes that are connected are covered.

**Use cases:**

DFS has several significant use cases in graph-related applications. It is commonly employed for pathfinding between nodes, identifying connected components, and performing topological sorting in directed acyclic graphs. Additionally, DFS is valuable in detecting cycles within graphs, solving puzzles, generating mazes, and aiding in decision-making processes for games and AI algorithms. Its versatility and efficiency make it a fundamental tool for a wide range of graph traversal and exploration tasks.

**Algorithm:**

1. # Variables:

2. n = number of nodes Example:

3. graph = the adjacency list representing our graph

4. visited = [ False, False, …, False]

5.

6. # The DFS algorithm

7. def dfs(i):

8. if visited[i]:

9. return

10. visited[i] = True

11. for n in graph[i]:

12. dfs(n)

13.

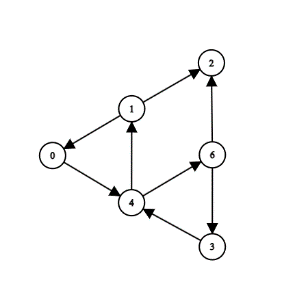
14. # Start from the first node

15. dfs(0)

16.

**Example:**

Here’s a small example illustrating an example of input outputs for the DFS:



We will use the Python code down below to outline the output of the algorithm on this graph:

1. graph = [ [4], [0, 2], [], [4], [1, 6], [], [2, 3] ]

2. n = len(graph)

3. visited = [False]\*n

4.

5. def dfs(i):

6. print(i, end=' | ')

7. if visited[i]:

8. return

9. visited[i] = True

10.

11. for n in graph[i]:

12. dfs(n)

13.

14. dfs(0)

15.

The corresponding output is:

Python >> 0 | 4 | 1 | 0 | 2 | 6 | 2 | 3 | 4 |

