

## Assignment No-01(Group A)

**Title:** - Implement depth first search algorithm and Breadth First Search algorithm

### Objectives:-

1. Understand the implementation of depth first search algorithm
2. Understand the implementation of Breadth First Search algorithm

### Problem Statement:-

Implement depth first search algorithm and Breadth First Search algorithm, Use an undirected graph and develop a recursive algorithm for searching all the vertices of a graph or tree data structure

### Software and Hardware requirements:-

1. **Operating system:** Linux- Ubuntu 16.04 to 17.10, or Windows 7 to 10,
2. **RAM-** 2GB RAM (4GB preferable)
3. You have to install **Python3** or higher version

### Theory-

#### 1. Depth First Search

What do we do once have to solve a maze? We tend to take a route, keep going until we discover a dead end. When touching the dead end, we again come back and keep coming back till we see a path we didn't attempt before. Take that new route. Once more keep going until we discover a dead end. Take a come back again... This is exactly how Depth-First Search works.

The Depth-First Search is a recursive algorithm that uses the concept of backtracking. It involves thorough searches of all the nodes by going ahead if potential, else by backtracking. Here, the word backtrack means once you are moving forward and there are not any more nodes along the present path, you progress backward on an equivalent path to seek out nodes to traverse. All the nodes are progressing to be visited on the current path until all the unvisited nodes are traversed after which subsequent paths are going to be selected.

### DFS Algorithm

A standard Depth-First Search implementation puts every vertex of the graph into one in all 2 categories:

- 1) Visited
- 2) Not Visited.

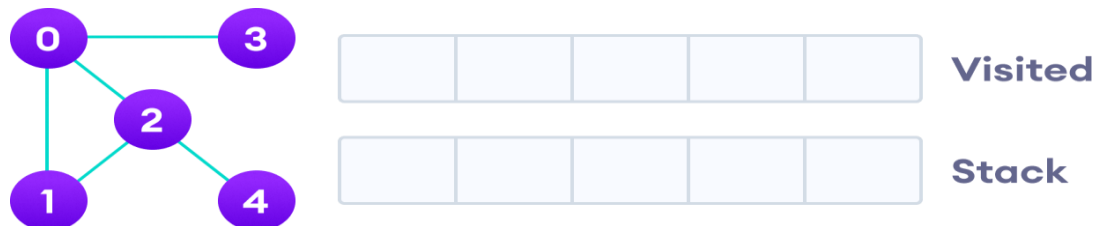
The only purpose of this algorithm is to visit all the vertex of the graph avoiding cycles.

The DSF algorithm follows as:

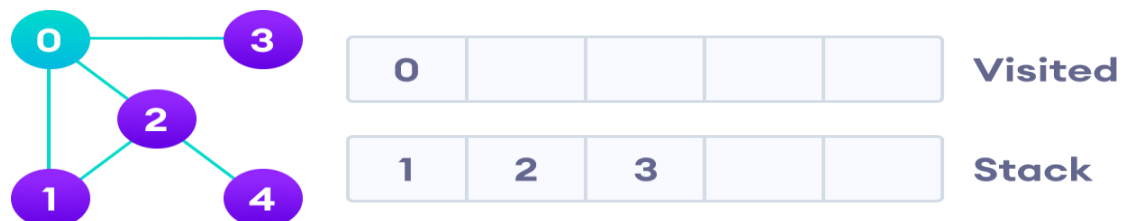
1. We will start by putting any one of the graph's vertex on top of the stack.
2. After that take the top item of the stack and add it to the visited list of the vertex.
3. Next, create a list of that adjacent node of the vertex. Add the ones which aren't in the visited list of vertexes to the top of the stack.
4. Lastly, keep repeating steps 2 and 3 until the stack is empty.

### Depth First Search Example

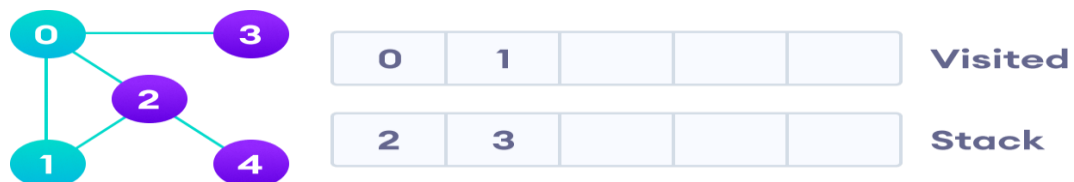
Let's see how the Depth First Search algorithm works with an example. We use an undirected graph with 5 vertices.



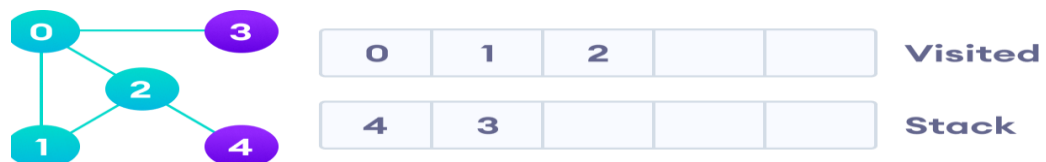
We start from vertex 0, the DFS algorithm starts by putting it in the Visited list and putting all its adjacent vertices in the stack.



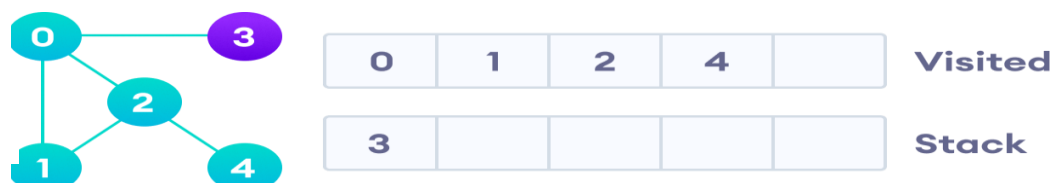
Next, we visit the element at the top of stack i.e. 1 and go to its adjacent nodes. Since 0 has already been visited, we visit 2 instead



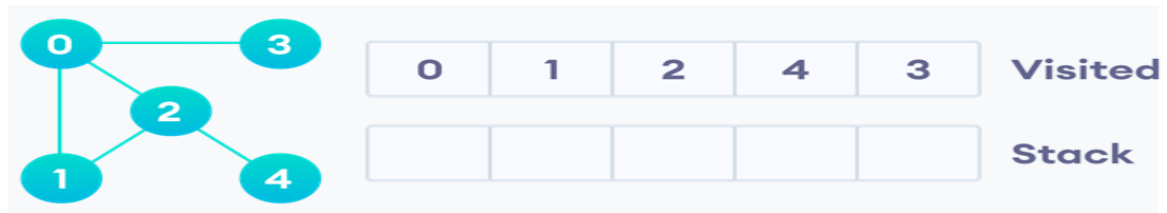
Vertex 2 has an unvisited adjacent vertex in 4, so we add that to the top of the stack and visit it.



Vertex 4 has an unvisited adjacent vertex in 3, so we add that to the top of the stack and visit it.



After we visit the last element 3, it doesn't have any unvisited adjacent nodes, so we have completed the Depth First Traversal of the graph.



After we visit the last element 3, it doesn't have any unvisited adjacent nodes, so we have completed the Depth First Traversal of the graph.

### Application of DFS Algorithm

1. For finding the path
2. To test if the graph is bipartite
3. For finding the strongly connected components of a graph
4. For detecting cycles in a graph.

### Breadth-First Search

Breadth-First Search (BFS) is an algorithm used for traversing graphs or trees.

Traversing means visiting each node of the graph. Breadth-First Search is a recursive algorithm to search all the vertices of a graph or a tree. BFS in python can be implemented by using data structures like a dictionary and lists. Breadth-First Search in tree and graph is almost the same. The only difference is that the graph may contain cycles, so we may traverse to the same node again.

### BFS Algorithm

Breadth-first search is the process of traversing each node of the graph, a standard BFS algorithm traverses each vertex of the graph into two parts:

- 1) Visited
- 2) Not Visited.

So, the purpose of the algorithm is to visit all the vertex while avoiding cycles.

BFS starts from a node, then it checks all the nodes at distance one from the beginning node, then it checks all the nodes at distance two, and so on. So as to recollect the nodes to be visited, BFS uses a queue.

### The steps of the algorithm work as follow:

1. Start by putting any one of the graph's vertices at the back of the queue.
2. Now take the front item of the queue and add it to the visited list.
3. Create a list of that vertex's adjacent nodes. Add those which are not within the visited list to the rear of the queue.
4. Keep continuing steps two and three till the queue is empty.

Many times, a graph may contain two different disconnected parts and therefore to make sure that we have visited every vertex, we can also run the BFS algorithm at every node.

**Explanation:**

1. Create a graph.
2. Initialize a starting node.
3. Send the graph and initial node as parameters to the bfs function.
4. Mark the initial node as visited and push it into the queue.
5. Explore the initial node and add its neighbours to the queue and remove the initial node from the queue.
6. Check if the neighbours node of a neighbouring node is already visited.
7. If not, visit the neighbouring node neighbours and mark them as visited.
8. Repeat this process until all the nodes in a graph are visited and the queue becomes empty.

**Advantages of BFS**

1. It can be useful in order to find whether the graph has connected components or not.
2. It always finds or returns the shortest path if there is more than one path between two vertices.

**Disadvantages of BFS**

1. The execution time of this algorithm is very slow because the time complexity of this algorithm is exponential.
2. This algorithm is not useful when large graphs are used.

**Conclusion**

Depth-First Search and Breadth-First Search (BFS) are used to traverse the graph or tree. We implemented Depth-First Search and Breadth-First Search (BFS) in python for searching all the vertices of a graph or tree data structure.