|  |  |
| --- | --- |
| **EX.N0:2** | **DEPTH FIRST SEARCH (DFS)** |
| **DATE: 06/03/2024** |

**AIM :**

To implement a depth first search problem using python.

**ALGORITHM:**

Step 1: Start.

Step 2: Start with root node G

Step 3: Search deep.if goal node has no children visit different node

Step 4: If state is reach goal node.

**PROGRAM**

def dfs(node, graph, visited, component):

    component.append(node)  # Store answer

    visited[node] = True  # Mark visited

    # Traverse to each adjacent node of a node

    for child in graph[node]:

        if not visited[child]:  # Check whether the node is visited or not

            dfs(child, graph, visited, component)  # Call the dfs recursively

if \_\_name\_\_ == "\_\_main\_\_":

    # Graph of nodes

    graph = {

        0: [2],

        1: [2, 3],

        2: [0, 1, 4],

        3: [1, 4],

        4: [2, 3]

    }

    node = 0  # Starting node

    visited = [False]\*len(graph)  # Make all nodes to False initially

    component = []

    dfs(node, graph, visited, component)  # Traverse to each node of a graph

    print( f" Following is the Depth-first search: {component}")

**Output:**

Following is the Depth-first search: [0, 2, 1, 3, 4]

**RESULT:**

Thus the experiment to do Depth first search by using python has been executed and verified Successfull