

# PRINCIPLES OF ARTIFICIAL INTELLIGENCE

## LABORATORY PROGRAMS

### DEPTH FIRST SEARCH:

#### SOURCE CODE:

```
import networkx as nx

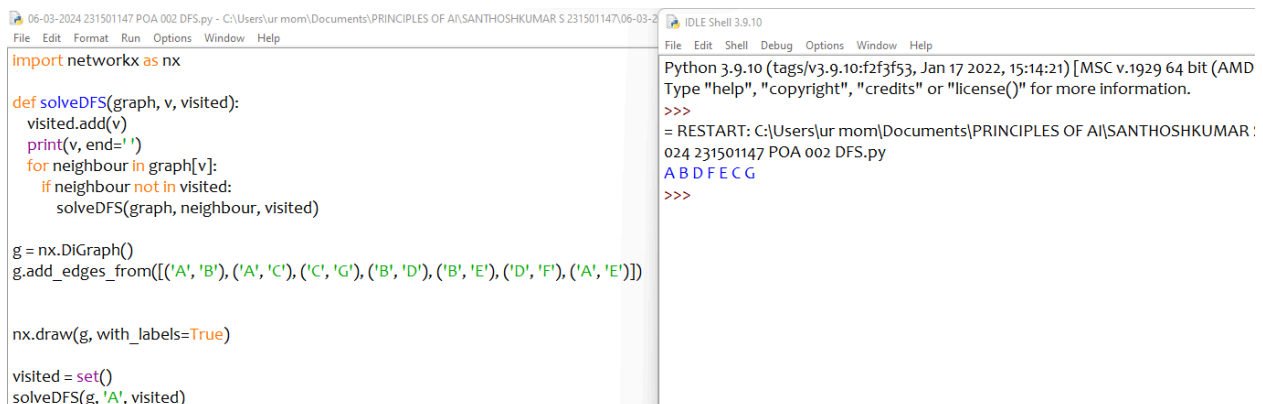
def solveDFS(graph, v, visited):
    visited.add(v)
    print(v, end=' ')
    for neighbour in graph[v]:
        if neighbour not in visited:
            solveDFS(graph, neighbour, visited)

g = nx.DiGraph()
g.add_edges_from([('A', 'B'), ('A', 'C'), ('C', 'G'), ('B', 'D'), ('B', 'E'), ('D', 'F'), ('A', 'E')])

nx.draw(g, with_labels=True)

visited = set()
solveDFS(g, 'A', visited)
```

#### OUTPUT:



The screenshot displays a Python IDE with two panes. The left pane shows the source code for a Depth First Search (DFS) algorithm using the networkx library. The code defines a function solveDFS, creates a directed graph g with nodes A, B, C, D, E, F, and G, and then calls solveDFS starting from node A. The right pane shows the output of the program, which is the sequence of nodes visited in order: A B D F E C G.

```
06-03-2024 231501147 POA 002 DFS.py - C:\Users\ur mom\Documents\PRINCIPLES OF AI\SANTHOSHKUMAR S 231501147\06-03-24
File Edit Format Run Options Window Help

import networkx as nx

def solveDFS(graph, v, visited):
    visited.add(v)
    print(v, end=' ')
    for neighbour in graph[v]:
        if neighbour not in visited:
            solveDFS(graph, neighbour, visited)

g = nx.DiGraph()
g.add_edges_from([('A', 'B'), ('A', 'C'), ('C', 'G'), ('B', 'D'), ('B', 'E'), ('D', 'F'), ('A', 'E')])

nx.draw(g, with_labels=True)

visited = set()
solveDFS(g, 'A', visited)
```

```
IDLE Shell 3.9.10
Python 3.9.10 (tags/v3.9.10:f2f3f53, Jan 17 2022, 15:14:21) [MSC v.1929 64 bit (AMD
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\ur mom\Documents\PRINCIPLES OF AI\SANTHOSHKUMAR :
024 231501147 POA 002 DFS.py
A B D F E C G
>>>
```