LP2 (AI) Lab Exp No.1

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.

import random

def bfs(graph, start, visited=None):

if visited is None:

visited = set()

queue = [start]

visited.add(start)

print(start)

while queue:

current = queue.pop(0)

for neighbor in graph[current] - visited:

print(neighbor)

queue.append(neighbor)

visited.add(neighbor)

return visited

def dfs(graph, start, visited=None):

if visited is None:

visited = set()

visited.add(start)

print(start)

for next\_node in graph[start] - visited:

dfs(graph, next\_node, visited)

return visited

def create\_graph():

graph = {}

num\_nodes = int(input("Enter the number of nodes: "))

for node in range(num\_nodes):

graph[str(node)] = set()

num\_edges = int(input("Enter the number of edges: "))

for \_ in range(num\_edges):

edge = input("Enter edge (format: node1 node2): ").split()

node1, node2 = edge

graph[node1].add(node2)

graph[node2].add(node1)

return graph

def main():

graph = create\_graph()

while True:

print("\nMenu:")

print("1. Breadth-First Search (BFS)")

print("2. Depth-First Search (DFS)")

print("3. Reset Graph")

print("4. Exit")

choice = input("Enter your choice: ")

if choice == '1':

start\_node = input("Enter the start node for BFS: ")

print("BFS Traversal:")

bfs(graph, start\_node)

elif choice == '2':

start\_node = input("Enter the start node for DFS: ")

print("DFS Traversal:")

dfs(graph, start\_node)

elif choice == '3':

print("Resetting graph.")

graph = create\_graph()

elif choice == '4':

print("Exiting program.")

break

else:

print("Invalid choice. Please enter a valid option.")

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

main()

OUTPUT:-



