*IDDFS*

def dls(graph, current, goal, depth, path, visited, traversal):

traversal.append(current)

if depth == 0 and current == goal:

return path + [current]

if depth > 0:

visited.add(current)

for neighbor in graph.get(current, []):

if neighbor not in visited:

result = dls(graph, neighbor, goal, depth - 1, path + [current], visited, traversal)

if result is not None:

return result

visited.remove(current)

return None

def iddfs(graph, start, goal, max\_depth):

for depth in range(max\_depth + 1):

visited = set()

traversal = []

print(f"\nDepth Level {depth}:")

path = dls(graph, start, goal, depth, [], visited, traversal)

print("Traversal Order:", " -> ".join(traversal))

if path:

print("Goal found at this level!")

return path

return None

def main():

print("Enter the number of nodes:")

n = int(input())

graph = {}

print("Enter the nodes:")

nodes = input().split()

for node in nodes:

graph[node] = []

print("Enter the edges in the format 'A B' meaning edge from A to B (type 'done' to finish):")

while True:

edge = input()

if edge.lower() == 'done':

break

u, v = edge.split()

if u in graph:

graph[u].append(v)

else:

graph[u] = [v]

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

goal\_node = input("Enter the goal node: ")

max\_depth = int(input("Enter the maximum depth limit: "))

result = iddfs(graph, start\_node, goal\_node, max\_depth)

if result:

print("\nGoal Path:", " -> ".join(result))

else:

print("\nGoal not found within depth limit.")

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

main()

