# DFS

def DFS(graph, start, target):

stack=[]

visited = set()

ordered\_traversal = []

stack.append(start)

visited.add(start)

while stack:

node = stack.pop()

print(f'Explroing {node}')

if node == target:

print(f'Node {node} found ...')

break

ordered\_traversal.append(node)

for neighbour in reversed(graph[node]):

if neighbour not in visited:

visited.add(neighbour)

stack.append(neighbour)

print("DFS traversal order:", ordered\_traversal)

graph = { #an adjacency list consisting of letters

'A': ['B', 'F','D','E'],

'B': ['A','K', 'J'],

'C': ['E'],

'D': ['A','G'],

'E': ['A','C','H','I'],

'F': ['A'],

'G': ['D'],

'H': ['E'],

'I': ['L','E'],

'J': ['B'],

'K': ['B','N','M'],

'L': ['I'],

'M': ['K'],

'N': ['K'],

}

start\_node = 'A'

print(f"BFS traversal starting from node '{start\_node}':")

DFS(graph, start\_node,target ='G')

Output:

BFS traversal starting from node 'A':

Explroing A

Explroing B

Explroing K

Explroing N

Explroing M

Explroing J

Explroing F

Explroing D

Explroing G

Node G found ...

DFS traversal order: ['A', 'B', 'K', 'N', 'M', 'J', 'F', 'D']