#BFS 1

tree = {

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

}

start = input("Enter the start state: ").strip().upper()

def bfs\_traversal(tree):

Open = [start]

close = []

while Open:

node = Open.pop(0) #pop the first element

if node not in close:

close.append(node) #adding the popped node

neighbour = tree[node]

for i in neighbour:

Open.append(i)

return close

print ("Traversal is",bfs\_traversal(tree))

OUTPUT:

|  |
| --- |
|  |

|  |
| --- |
|  |

#2

print("Breadth First Search")

tree = {'A':['B','C'],'B':['D','E'],'C':['F','G'],'D':[],'E':[],'F':[],'G':[]}

start = input("Enter the start state: ").strip().upper()

goal = input("Enter the goal state: ").strip().upper()

def bfs\_goal(tree):

Open = [start]

close = []

if start == goal:

print("start state itself is the goal state")

return close

close.append(start)

while Open:

node = Open.pop(0)

neighbour = tree[node]

for i in neighbour:

if i not in close:

close.append(i)

Open.append(i)

if i == goal:

return close

print("goal node not found")

print("Traversal is",bfs\_goal(tree))

OUTPUT:

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |