Depth first search

# Using a Python dictionary to act as an adjacency list

graph = {

'5' : ['3','7'],

'3' : ['2', '4'],

'7' : ['8'],

'2' : [],

'4' : ['8'],

'8' : []

}

visited = set() # Set to keep track of visited nodes of graph.

**def** **dfs**(visited, graph, node): #function for dfs

**if** node **not** **in** visited:

**print** (node)

visited.add(node)

**for** neighbour **in** graph[node]:

dfs(visited, graph, neighbour)

# Driver Code

**print**("Following is the Depth-First Search")

dfs(visited, graph, '5')

breadth first search

graph = {

'A' : ['B','C'],

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

'C' : ['F'],

'D' : [],

'E' : ['F'],

'F' : []

}

visited = [] # List to keep track of visited nodes.

queue = [] #Initialize a queue

def bfs(visited, graph, node):

visited.append(node)

queue.append(node)

while queue:

s = queue.pop(0)

print (s, end = " ")

for neighbour in graph[s]:

if neighbour not in visited:

visited.append(neighbour)

queue.append(neighbour)

# Driver Code

bfs(visited, graph, 'A')