[8:45 pm, 29/11/2022] G 1: from collections import defaultdict

class Graph:

def \_init\_ (self):

self.graph = defaultdict(list)

def addEdge(self,u,v):

self.graph[u].append(v)

def BFS(self,s):

visited = [False]\*(len(self.graph))

queue=[]

queue.append(s)

visited[s]=True

while queue:

s=queue.pop(0)

print(s,end=" ")

for i in self.graph[s]:

if visited[i]==False:

queue.append(i)

visited[i]=True

#driver code or object creation for the class Graph

g = Graph()

g.addEdge(0,1)

g.addEdge(0,2)

g.addEdge(1,2)

g.addEdge(2,0)

g.addEdge(2,3)

g.addEdge(3,3)

print("Following is Breadth First Traversal""(starting from vertex 2)")

g.BFS(2)

[9:29 pm, 29/11/2022] G 1: from collections import defaultdict

class Graph:

def \_init\_(self):

self.graph = defaultdict(list)

def addEdge(self,u,v):

self.graph[u].append(v)

def DFSUtil(self,v,visited):

visited.add(v)

print(v,end=' ')

for neighbour in self.graph[v]:

if neighbour not in visited:

self.DFSUtil(neighbour,visited)

def DFS(self,v):

visited=set()

self.DFSUtil(v,visited)

g=Graph()

g.addEdge(0,1)

g.addEdge(0,2)

g.addEdge(1,2)

g.addEdge(2,0)

g.addEdge(2,3)

g.addEdge(3,3)

g.DFS(2)