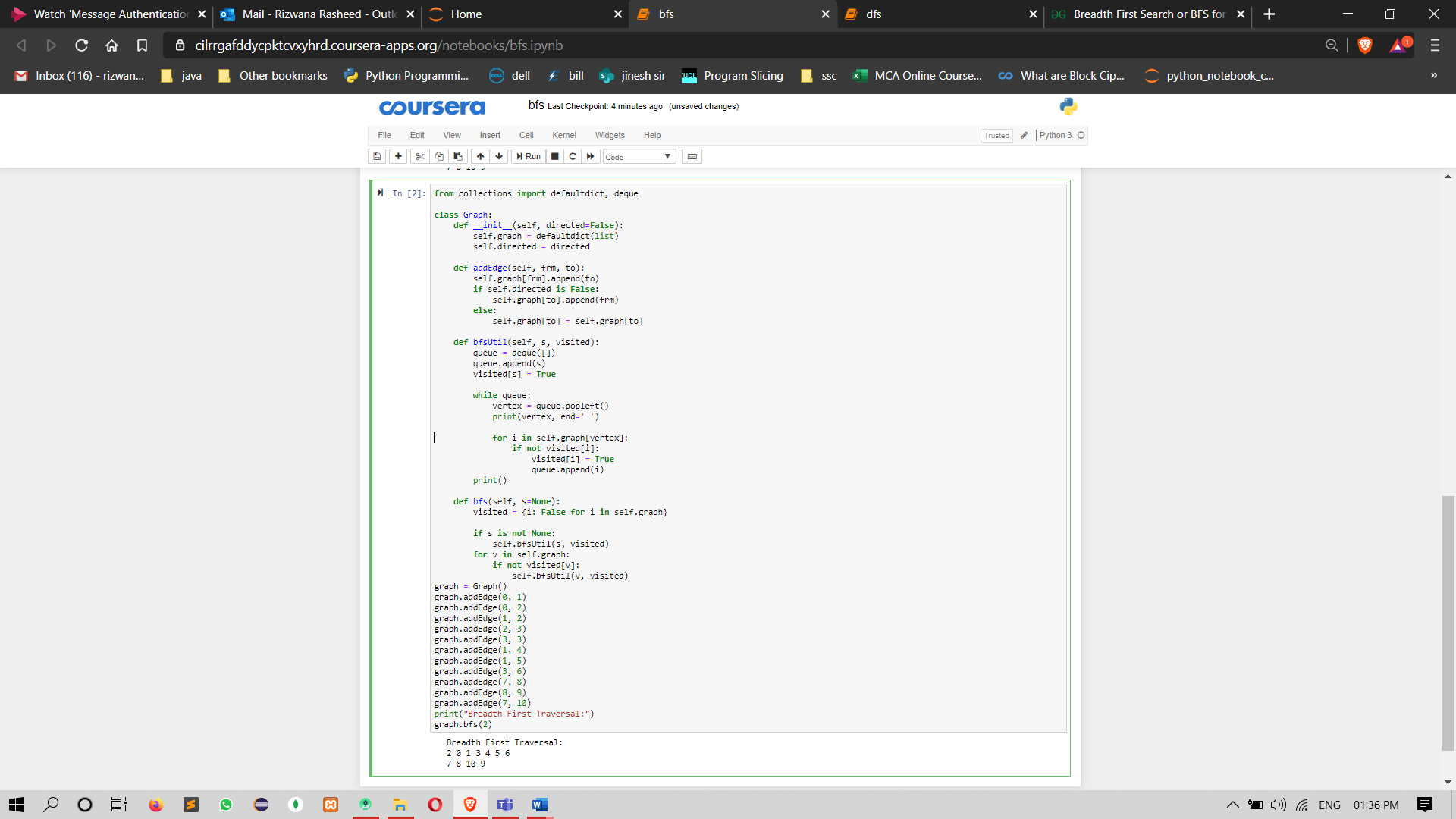
1. BFS **(19/Apr/2020)**
   1. Print the nodes in the order they are visited



from collections import defaultdict, deque

class Graph:

def \_\_init\_\_(self, directed=False):

self.graph = defaultdict(list)

self.directed = directed

def addEdge(self, frm, to):

self.graph[frm].append(to)

if self.directed is False:

self.graph[to].append(frm)

else:

self.graph[to] = self.graph[to]

def bfsUtil(self, s, visited):

queue = deque([])

queue.append(s)

visited[s] = True

while queue:

vertex = queue.popleft()

print(vertex, end=' ')

for i in self.graph[vertex]:

if not visited[i]:

visited[i] = True

queue.append(i)

print()

def bfs(self, s=None):

visited = {i: False for i in self.graph}

if s is not None:

self.bfsUtil(s, visited)

for v in self.graph:

if not visited[v]:

self.bfsUtil(v, visited)

graph = Graph()

graph.addEdge(0, 1)

graph.addEdge(0, 2)

graph.addEdge(1, 2)

graph.addEdge(2, 3)

graph.addEdge(3, 3)

graph.addEdge(1, 4)

graph.addEdge(1, 5)

graph.addEdge(3, 6)

graph.addEdge(7, 8)

graph.addEdge(8, 9)

graph.addEdge(7, 10)

print("Breadth First Traversal:")

graph.bfs(2)