LAB 10: Graph Traversal Techniques

CS211 – Data Structures and Algorithms
Usman Institute of Technology
Fall 2019

- **How to submit:**
  - Create an account on http://www.turnitin.com/ as a Student (if you don't have already)
  - Use following information at time of sign-up

    CS Section A
    - Class ID: 22664649
    - Enrollment Key: DSFALL19CSA

    CS Section B
    - Class ID: 22664651
    - Enrollment Key: DSFALL19CSB

A. Create a class Graph and implement the Graph operations in the following order.

1. Add a constructor of the class that takes one argument vertex in order to set the number of vertices. The constructor should also initialize a 2d array for adjacency matrix

```
class Graph:
    def __init__(self,vertex):
        // your code goes here
```

2. Add a function **AddEdge()** that takes two arguments source and destination and inserts 1 to the matrix whose vertices are connected

```
    def AddEdge(self,src,dest):
        // your code goes here
```

```
Example:
g = Graph(5)
g.AddEdge(0,1)
g.AddEdge(1,2)
```

3. Add a function **PrintMatrix()** that prints the adjacency matrix.

```
    def PrintMatrix(self):
        // your code goes here
```

```
Example:
g = Graph(5)
g.AddEdge(0,1)
g.AddEdge(0,2)
g.AddEdge(1,3)
g.AddEdge(3,0)
g.PrintMatrix()
```

4. Add a function **GetNeighbours()** which takes a vertex as a parameter and returns all the neighbours of that vertex.

```
    def GetNeighbours(self, vertex):
        // your code goes here
```

```
Example:
g = Graph(5)
g.AddEdge(0,1)
g.AddEdge(0,2)
g.GetNeighbours(0)

#The function should return
1,2
```

5. Add a function **BFS()** that takes a parameter source and performs breadth first search in the graph starting from the source.

```
    def BFS(self,source):
        // your code goes here
```

```
Example:
g = Graph(5)
g.AddEdge(0,1)
g.AddEdge(0,2)
g.AddEdge(1,3)
g.AddEdge(3,0)

g.GetNeighbours(0)
g.BFS(0)

#The function should return
0,2,1,3
```

6. Add a function **DFS()** that takes a parameter source and performs Depth first search in the graph starting from the source.

```
    def DFS(self,source):
        // your code goes here
```

```
Example:
g = Graph(5)
g.AddEdge(0,1)
g.AddEdge(0,2)
g.AddEdge(1,3)
g.AddEdge(3,0)

g.GetNeighbours(0)
g.DFS(0)

#The function should return
0,3,2,1
```