LE - 8
Traversing Graphs

Introduction:

LE - 8 is a hands-on component for the concepts covered in the lecture on graph traversal.

Expected Functionality:

Building on top of your work from LE - 7, you will be writing traversal functions to operate on your graph. The following are the expectations associated with LE - 8:

1. Implement the **`dijkstra(graph, src, dest = None)`** function to add a find the shortest distance between to all nodes from a source node. The function should return a dictionary with the key as the destination node and the value as the distance to that node from the source node. This takes in the following inputs:
   - **`graph`** - the graph created using the graph class from LE7 to be traversed.
   - **`src`** - the node to start traversing from, return none if not a valid node.
   - **`dest`** - the destination node to find the shortest distance to. Default value None; returns the entire dictionary. If dest is set to a node value; return only the distance to that destination node.
2. Implement the **`primMST(graph,  src)`** to find the Minimal Spanning Tree using Prim's algorithm from a source node to traverse the whole graph. The function should return the cost of the MST by summing up the selected edge weights during traversal. This takes in the following inputs:
   - **`graph`** - the graph created using the graph class from LE7 to be traversed.
   - **`src`** - the node to start traversing from, return none if not a valid node.

Starter Code:

You are provided with the *LE8_Graph_Traversal.ipynb* file. The notebook contains the following helper functions to aid you in testing:

- `printArr()` - prints the distance dictionary received from `dijkstra()`.
- `visualizeGraph()` - visualization function for the graph using the NetworkX library.
- `read_data_from_txt()` - reads and creates a graph from the .txt file passed.

Some .txt file links are also shared in the notebook for you to test your code. These should be removed while submitting the code and only the `dijkstra` and `primMST` functions should be part of the submission code.

You can explore these links for a helpful visualization of the Dijkstra and Prim's MST algorithms.

Rubric:

Your code will be tested with provided test cases.

Location of the code:

The code would be provided at:
https://colab.research.google.com/drive/1fy4dgVxu5h_ieur81Y5q8olFMJdCaLZq?usp=sharing

What to do when done:

Once you have completed the exercise, you should upload it to the codePost (will be visible soon).
Please ensure the following while submitting:

1. Once satisfied with your code, you should download the file as a python script (.py file),
2. by going to **File > Download > Download .py**
3. The name of the file should be LE8.py
4. You can run the test cases on your script up to a limit of 50 times.
5. Once satisfied with the test runs, complete your submission.