 # Payal Chavan
 # Homework-4
 # Summer 2024
 # CS 5800 Algorithms (Seattle)
 # Date: 06/14/2024

Question 1: Implementation of DFS including pre-post values.

Output:

```
/usr/bin/python3
/Users/payalchavan/Documents/Algorithms/Assignment4/graph_algorithms.py
Graph:
0 (100, 100) inf -> -1 [1 (200, 20) inf -> -1, 6 (200, 350) inf -> -1, ]
1 (200, 20) inf -> -1 [0 (100, 100) inf -> -1, 2 (750, 100) inf -> -1, 4 (400, 250) inf -> -1, ]
2 (750, 100) inf -> -1 [1 (200, 20) inf -> -1, 5 (850, 400) inf -> -1, 3 (550, 150) inf -> -1, ]
3 (550, 150) inf -> -1 [2 (750, 100) inf -> -1, 5 (850, 400) inf -> -1, 4 (400, 250) inf -> -1,
7 (550, 450) inf -> -1, ]
4 (400, 250) inf -> -1 [1 (200, 20) inf -> -1, 3 (550, 150) inf -> -1, 6 (200, 350) inf -> -1, ]
5 (850, 400) inf -> -1 [2 (750, 100) inf -> -1, 3 (550, 150) inf -> -1, ]
6 (200, 350) inf -> -1 [0 (100, 100) inf -> -1, 4 (400, 250) inf -> -1, 7 (550, 450) inf -> -1,
8 (350, 550) inf -> -1, 9 (120, 600) inf -> -1, ]
7 (550, 450) inf -> -1 [3 (550, 150) inf -> -1, 6 (200, 350) inf -> -1, 8 (350, 550) inf -> -1, ]
8 (350, 550) inf -> -1 [6 (200, 350) inf -> -1, 7 (550, 450) inf -> -1, 9 (120, 600) inf -> -1, ]
9 (120, 600) inf -> -1 [6 (200, 350) inf -> -1, 8 (350, 550) inf -> -1, ]

DFS Results
Node 9 (pre, post): (9, 10)  parent 8
Node 8 (pre, post): (8, 11)  parent 7
Node 7 (pre, post): (7, 12)  parent 6
Node 6 (pre, post): (6, 13)  parent 4
Node 4 (pre, post): (5, 14)  parent 3
Node 3 (pre, post): (4, 15)  parent 5
Node 5 (pre, post): (3, 16)  parent 2
Node 2 (pre, post): (2, 17)  parent 1
Node 1 (pre, post): (1, 18)  parent 0
Node 0 (pre, post): (0, 19)  parent -1
```

Question 2: Implementation of Dijkstra's algorithm.

Output:

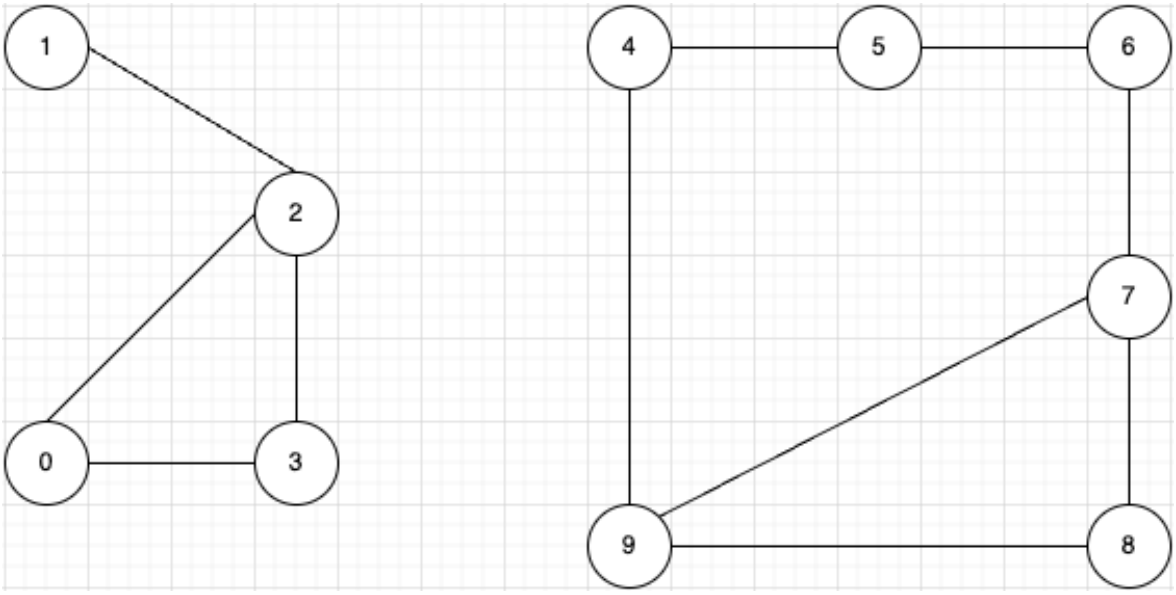
```
Dijkstra's Results:
0 (100, 100) 0.0 -> -1
1 (200, 20) 128.1 -> 0
2 (750, 100) 683.9 -> 1
3 (550, 150) 613.1 -> 4
4 (400, 250) 432.9 -> 1
5 (850, 400) 1000.1 -> 2
6 (200, 350) 269.3 -> 0
7 (550, 450) 633.3 -> 6
8 (350, 550) 519.3 -> 6
9 (120, 600) 531.7 -> 6
(base)
payalchavan@Payals-MacBook-Air
/Users/payalchavan/Documents/Algorithms/Assignment4
⚡
```

Question 3: Implementation of DFS on unweighted undirected graph.

Output:

```
⚡ /usr/bin/python3
/Users/payalchavan/Documents/Algorithms/Assignment4/graphA.py

DFS Results:
Node 1 (pre, post): (2, 3)  parent 2
Node 3 (pre, post): (4, 5)  parent 2
Node 2 (pre, post): (1, 6)  parent 0
Node 0 (pre, post): (0, 7)  parent -1
Node 9 (pre, post): (13, 14)  parent 8
Node 8 (pre, post): (12, 15)  parent 7
Node 7 (pre, post): (11, 16)  parent 6
Node 6 (pre, post): (10, 17)  parent 5
Node 5 (pre, post): (9, 18)  parent 4
Node 4 (pre, post): (8, 19)  parent -1
(base)
payalchavan@Payals-MacBook-Air
/Users/payalchavan/Documents/Algorithms/Assignment4
⚡
```



From the console output we can see that Node 0 has parent -1 which means this node acts as Source Node and based on pre/post values (0, 7). We confirm that nodes processes from this node and ends at this node. So, whatever the values for pre/post of other nodes are in between 0 & 7 are in the same DFS tree starting from node 0.
We can see those nodes are 0, 1, 2, and 3 which has values between 0 and 7.

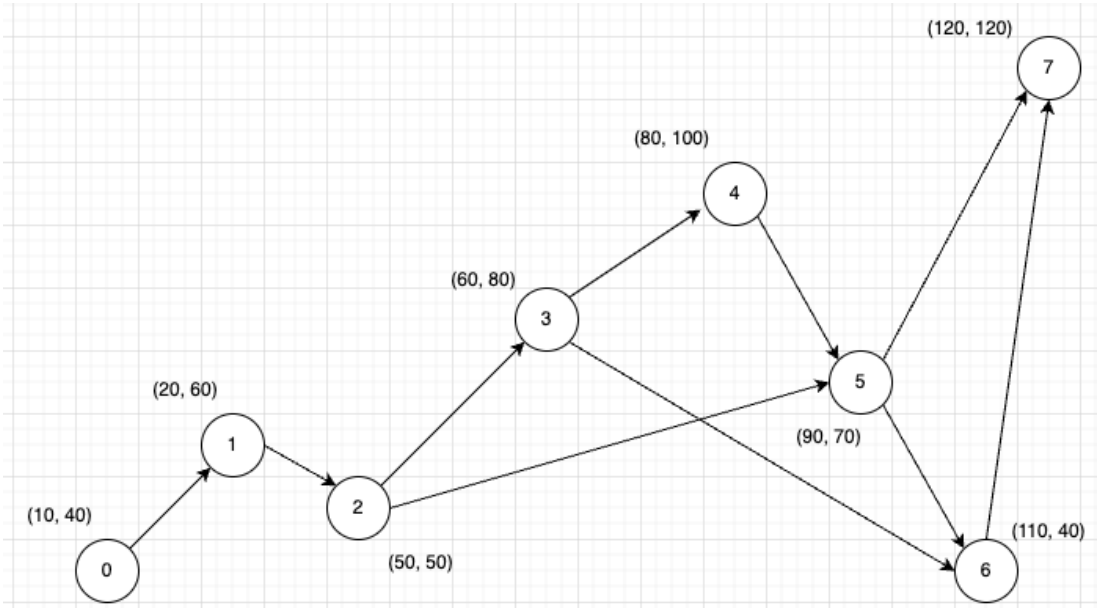
Similarly, Node 4 acts as Source Node because parent value is -1. And based on pre/post values (8, 17), all the nodes which has those values in between can be reached from Node 4 in DFS tree. These nodes are 4, 5, 6, 7, 8, and 9.
From the graph diagram we can see its indeed part of same graph.

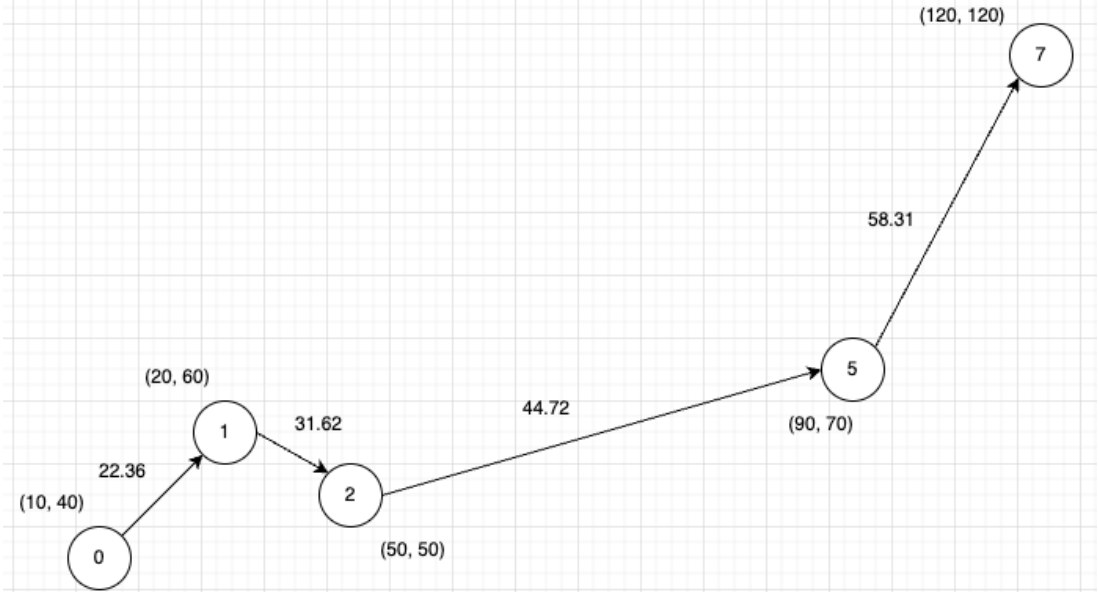
Thus, we can say nodes starting from 0 and 4 creates 2 Strongly Connected Components.

Question 4: Implementation of Dijkstra's algorithm on a directed graph.
Output:

```
⚡ /usr/bin/python3
/Users/payalchavan/Documents/Algorithms/Assignment4/graphB.py

Dijkstra's Results:
0 (10, 40) 0.0 -> -1
1 (20, 60) 22.4 -> 0
2 (50, 50) 54.0 -> 1
3 (60, 80) 85.6 -> 2
4 (80, 100) 113.9 -> 3
5 (90, 70) 98.7 -> 2
6 (110, 40) 134.8 -> 5
7 (120, 120) 157.0 -> 5
```





When we apply Dijkstra's Algorithm on the above Directed Connected Graph, we get the minimum cost between each node as we see in the result.
From the node 7, we get the parent node from which it has the minimum cost which is Node 5. Similarly, through parent node we reach to Node 0 (source node).
From this, we can get the resulting minimum spanning tree. As we can see from the above graph and results, that the minimum distance from 0 to 7 is 157.0.

Reflection:

From this coding exercise, I learnt how to implement DFS and Dijkstra's Algorithm on any given graph. These algorithms provided valuable insights into graph traversal and path-finding techniques.

Acknowledgements:

I would like to express my sincere gratitude to the following individuals and resources for their invaluable contributions
to this assignment:
1) Prof. Bruce Maxwell: Thank you, Professor Bruce Maxwell for your guidance in this assignment. Your expertise in
algorithms greatly influenced my work.
2) Classmates and TA's: I appreciate the discussions of my classmates, and TA's who clarified my doubts.
3) DPV Algorithms Textbook: This textbook was a useful resource for me to understand the basics of algorithms.
4) https://www.geeksforgeeks.org/depth-first-search-or-dfs-for-a-graph/: This website provided clear explanations for
understanding DFS.
5) https://www.programiz.com/dsa/dijkstra-algorithm: This was a useful resource to understand Dijkstra's Algorithm.