1. (10 Points) Given adjacency list representation of an undirected graph below and source vertex as 0, use algorithms provided in the class slides

a) Draw path tree using Depth First Search

b) Draw path tree using Breadth First Search

0: 1 9 8
1: 3 4 6 0
2: 4 8 6
3: 6 5 9 1
4: 7 9 6 2 1
5: 6 3
6: 9 7 5 4 3 2 1
7: 9 8 6 4
8: 7 2 0
9: 7 6 4 3 0

Answer: The path tree from vertex 0 is shown in figure 1.a and figure 1.b.

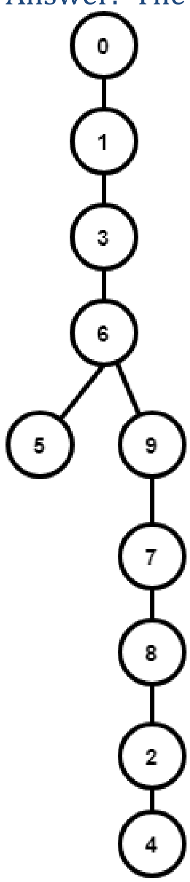


Figure 1.a

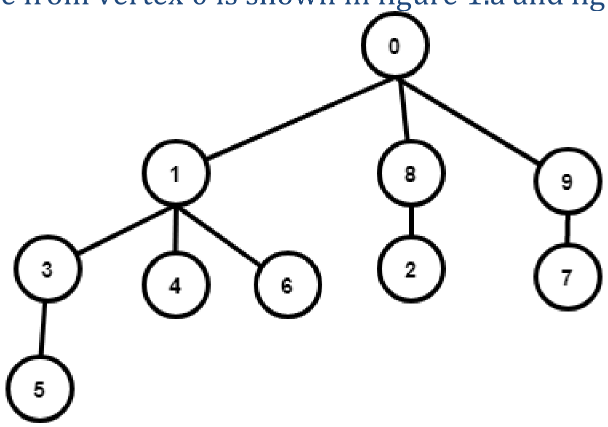


Figure 1.b

The array contents for DFS (a) are the following:

	0	1	2	3	4	5	6	7	8	9
marked	true	true	true	true	true	true	true	true	true	true
edgeTo	0	0	8	1	2	6	3	9	7	6

The array contents for BFS (b) are the following:

	0	1	2	3	4	5	6	7	8	9
marked	true	true	true	true	true	true	true	true	true	true
edgeTo	0	0	8	1	1	3	1	9	0	0

2. (5 Points) Given adjacency list representation of a digraph below, does it have cycles? If so, provide a cycle.

```
0: 1 2 3 4 8
1: 2 3 6 8
2: 3 4 8
3: 1 6 8 9
4: 5 7 9
5: 4 6 9
6: 1 5 8
7: 8 9
8: 0 9
9: 3 6 7
```

Answer: Yes, the digraph has directed cycles. By following DirectedCycle code on slide 28 from module 10, this cycle is found: 3, 1, 2, 3.

3. (5 Points) Given adjacency list representation of a digraph below, does it have a topological order? If so, provide one. Otherwise, explain why.

```
0: 6 1
1: 3 4 6
2:
3: 2
4: 3
5: 6 2 4 0
6: 2 4
7: 3 0 9 8
8: 2 4 3 6
9: 4 3 6 2
```

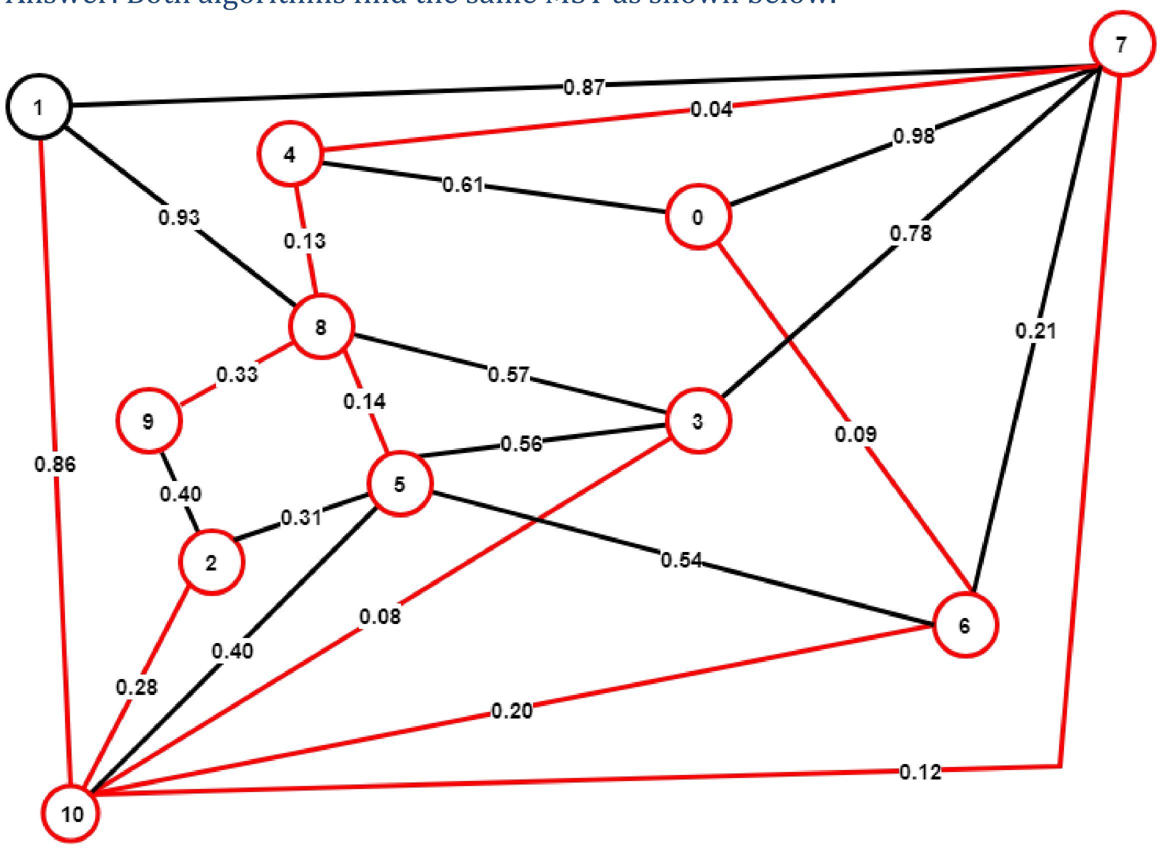
Answer: Yes the digraph has topological order. By following DepthFirstOrder code on slide 29 from module 10, this is a topological order
7, 8, 9, 5, 0, 1, 6, 4, 3, 2

4. (10 Points) Given the following edge weighted graph, provide MST edges in the order as they are added to the MST and MST weight, using

a) Lazy Prim's MST algorithm starting from vertex 0

b) Kruskal's MST algorithm

Answer: Both algorithms find the same MST as shown below.

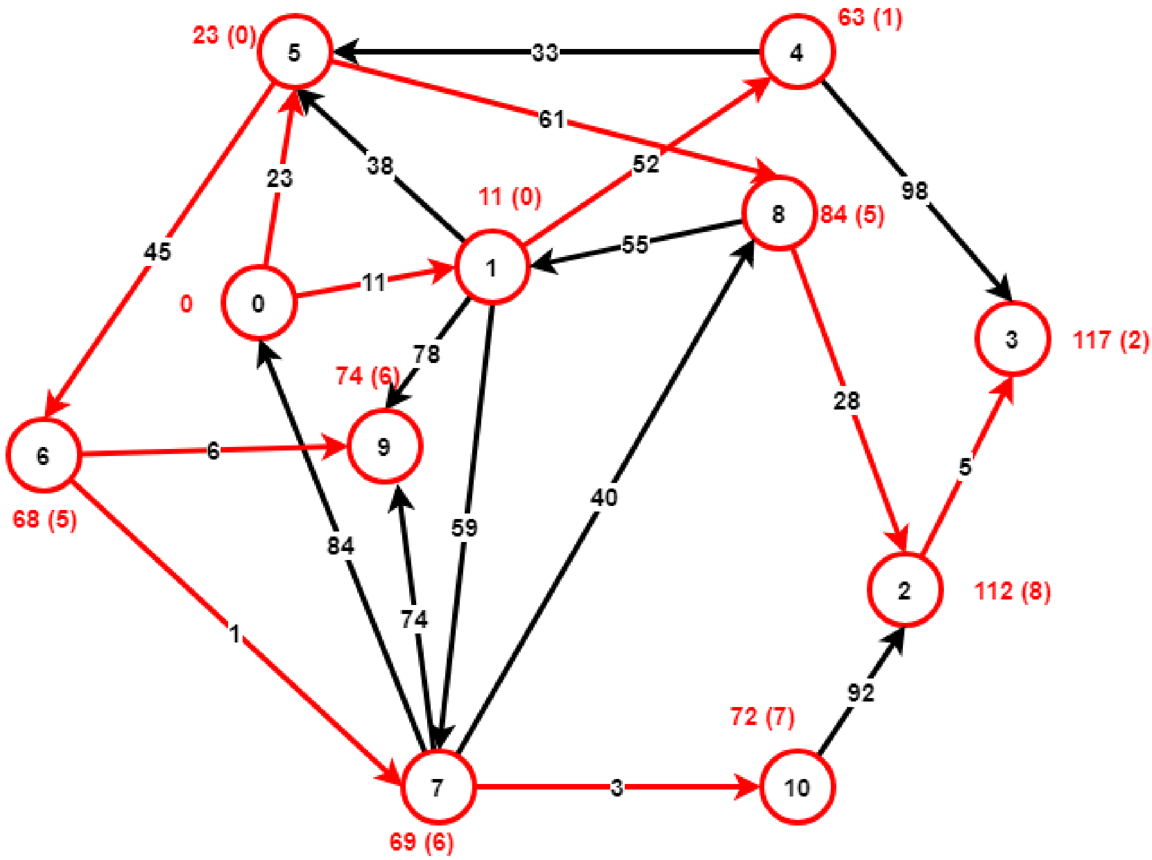


The following table shows edges in the order as they are added to MST and weight of MST.

	Lazy Prim's MST	Kruskal's MST
MST edges	0-6, 0.09 6-10, 0.20 3-10, 0.08 7-10, 0.12 4-7, 0.04 4-8, 0.13 5-8, 0.14 2-10, 0.28 8-9, 0.33 1-10, 0.86	4-7, 0.04 3-10, 0.08 0-6, 0.09 7-10, 0.12 4-8, 0.13 5-8, 0.14 6-10, 0.20 2-10, 0.28 8-9, 0.33 1-10, 0.86
MST weight	2.27	2.27

5. (10 Points) Given the following edge weighted digraph, find the shortest path tree from source vertex 0, including the trace of distTo and edgeTo contents similar to slide#18 from module 11.

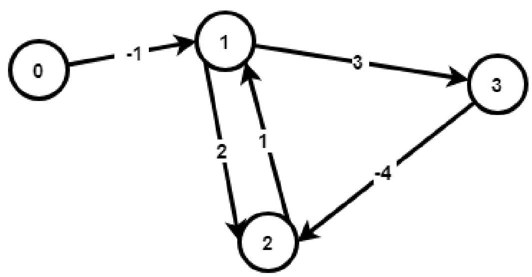
Answer:



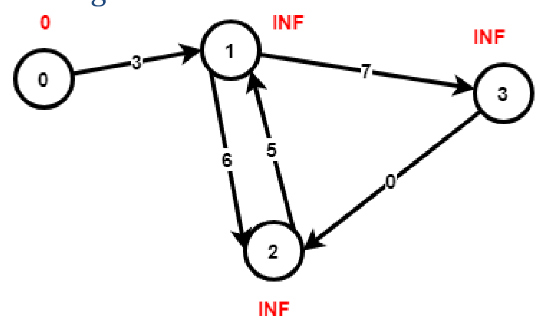
	distTo											edgeTo										
	0	1	2	3	4	5	6	7	8	9	10	0	1	2	3	4	5	6	7	8	9	10
Initial	0	INF	INF	INF	INF	INF	INF	INF	INF	INF	INF											
Relax 0	~~0~~	11	INF	INF	INF	23	INF	INF	INF	INF	INF		0->1				0->5					
Relax 1	~~0~~	~~11~~	INF	INF	63	23	INF	70	INF	89	INF		0->1			1->4	0->5		1->7		1->9	
Relax 5	~~0~~	~~11~~	INF	INF	63	~~23~~	68	70	84	89	INF		0->1			1->4	0->5	5->6	1->7	5->8	1->9	
Relax 4	~~0~~	~~11~~	INF	161	~~63~~	~~23~~	68	70	84	89	INF		0->1		4->3	1->4	0->5	5->6	1->7	5->8	1->9	
Relax 6	~~0~~	~~11~~	INF	161	~~63~~	~~23~~	~~68~~	69	84	74	INF		0->1		4->3	1->4	0->5	5->6	6->7	5->8	6->9	
Relax 7	~~0~~	~~11~~	INF	161	~~63~~	~~23~~	~~68~~	~~69~~	84	74	72		0->1		4->3	1->4	0->5	5->6	6->7	5->8	6->9	7->10
Relax 10	~~0~~	~~11~~	164	161	~~63~~	~~23~~	~~68~~	~~69~~	84	74	~~72~~		0->1	10->2	4->3	1->4	0->5	5->6	6->7	5->8	6->9	7->10
Relax 9	~~0~~	~~11~~	164	161	~~63~~	~~23~~	~~68~~	~~69~~	84	~~74~~	~~72~~		0->1	10->2	4->3	1->4	0->5	5->6	6->7	5->8	6->9	7->10
Relax 8	~~0~~	~~11~~	112	161	~~63~~	~~23~~	~~68~~	~~69~~	~~84~~	~~74~~	~~72~~		0->1	8->2	4->3	1->4	0->5	5->6	6->7	5->8	6->9	7->10
Relax 2	~~0~~	~~11~~	~~112~~	117	~~63~~	~~23~~	~~68~~	~~69~~	~~84~~	~~74~~	~~72~~		0->1	8->2	2->3	1->4	0->5	5->6	6->7	5->8	6->9	7->10
Relax 3	~~0~~	~~11~~	~~112~~	~~117~~	~~63~~	~~23~~	~~68~~	~~69~~	~~84~~	~~74~~	~~72~~		0->1	8->2	2->3	1->4	0->5	5->6	6->7	5->8	6->9	7->10

6. (5 Points) To find shortest paths for an edge weighted graph with negative edge weights, 1) find the minimum edge weight, 2) add the absolute value of the minimum edge weight to every edge, 3) run Dijkstra's SPT algorithm to find the shortest path from the modified graph.
Would the above algorithm work? If so, explain why. If not, provide a counter example.

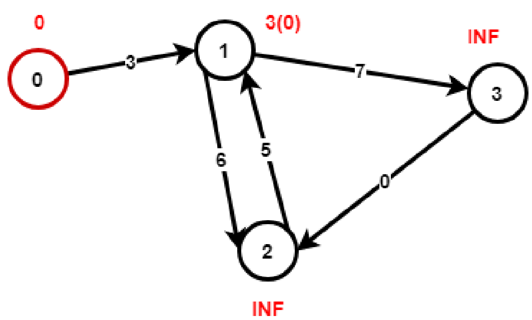
Answer: The above algorithm does not always work. Here is a counter example. By following the algorithm, the "shortest path" from 0 to 2 is 0->1->2, with path weight 1, while a shorter path 0->1->3->2 provide path weight -2 from 0 to 2.



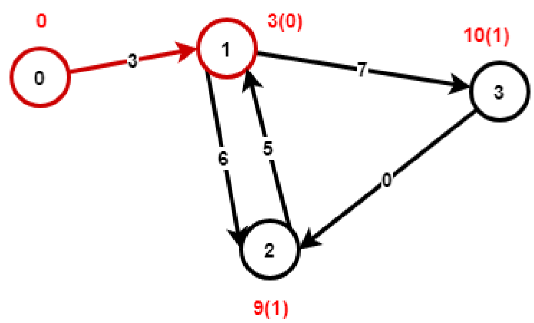
Original Graph



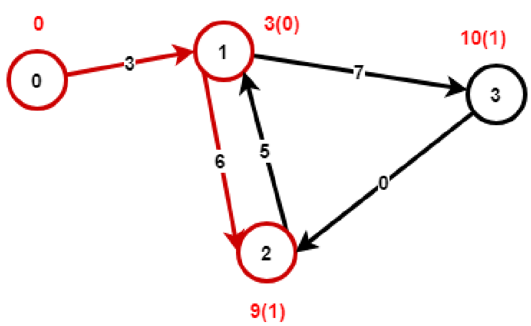
Added 4 to each Edge



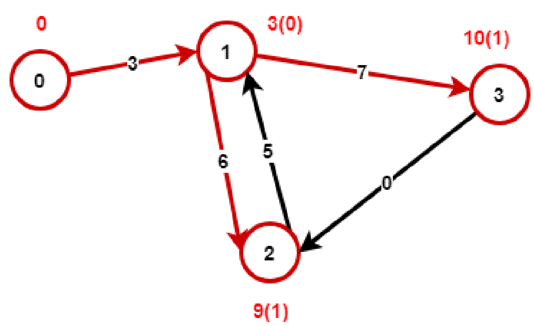
Add 0 to SPT, Relax 0



Add 1 to SPT, Relax 1



Add 2 to SPT, Relax 2



Add 3 to SPT

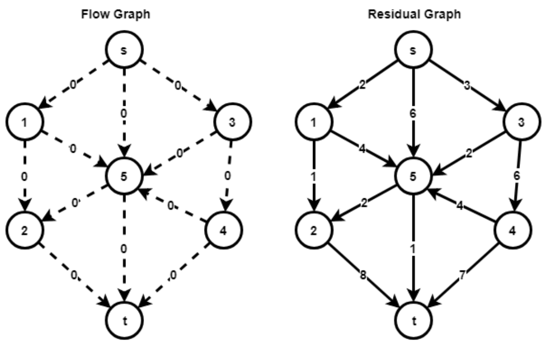
7. (5 Points) Given the network flow digraph below,

a) What is the maximum flow?

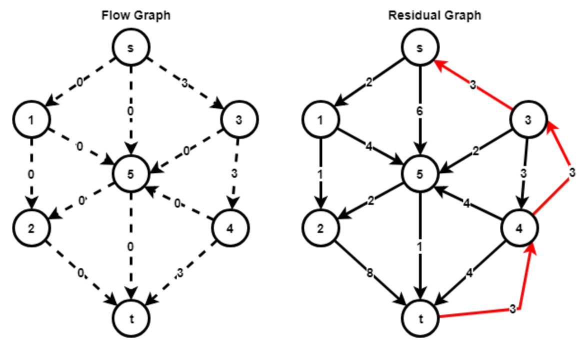
b) Provide the flow graph that shows the maximum flow paths.

Answer:

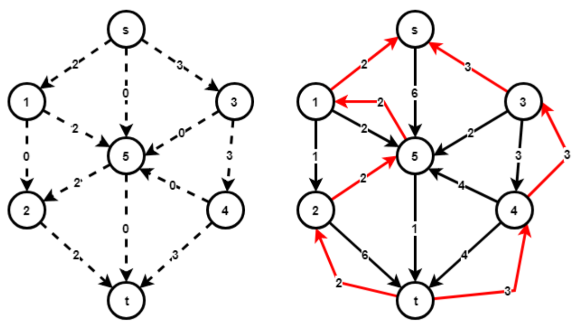
a) The maximum flow is 7. B) There are multiple ways to reach the maximum flow, the following flow graph shows one.



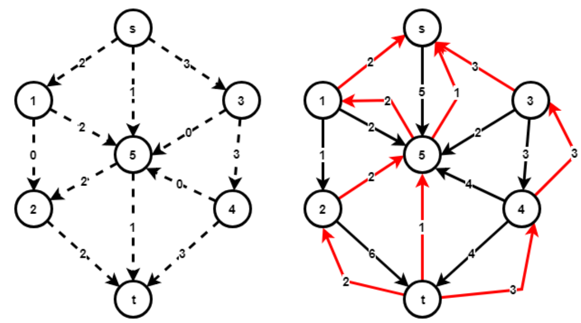
(0) Initial



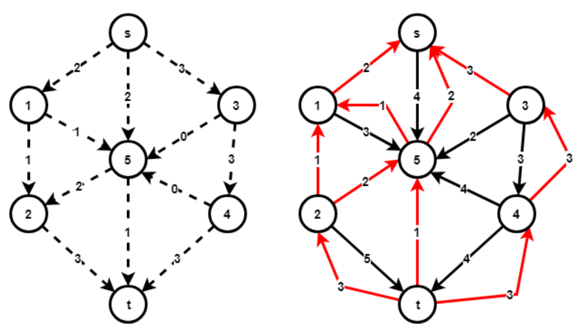
(1) After 3 units flow added along s, 3, 4, t



(2) After 2 units flow added along s, 1, 5, 2, t



(3) After 1 unit flow added along s, 5, t



(4) After 1 unit flow added along s, 5, 1, 2, t

8. (5 Points) Indicate true or false for each of the following statements regarding complexity classes.

a) There are some problems in P but not in NP
b) There is no overlap between NP and undecidable problems
c) NP-Complete problems are harder than P problems
d) It has been proven that P!=NP
e) NP stands for Non-Polynomial

Answer:
a) False b) True c) True d) False e) False

9. (5 Points) For each of the algorithms listed in the table, indicate which of the following algorithm design techniques is used: Greedy algorithm, divide and conquer, dynamic programming, randomized algorithms, and backtracking.

Answer:

Algorithms	**Algorithm Design Technique Used**
Merge Sort	Divide and Conquer Algorithm
Prim's MST	Greedy Algorithm
Bellman-Ford SPT	Dynamic Programming
Skip Lists	Randomized Algorithm
Tic-tac-toe game using minimax strategy with α-β pruning	Backtracking

Extra Credit Questions

10. (4 Points) Consider the graphs defined by the following
four sets of edges:

a) 0-2 0-3 1-3 0-3 2-5 5-6 3-6 4-7 4-8 5-8 5-9 6-7 6-9 8-8
b) 0-2 0-3 1-3 1-4 2-5 2-9 3-6 4-7 4-8 5-8 5-9 6-7 6-9 7-8
c) 1-2 1-3 0-3 0-4 2-5 2-9 3-6 4-7 4-8 5-8 5-9 6-7 6-9 7-8
d) 4-1 7-9 6-2 7-3 5-0 0-2 0-8 1-6 3-9 6-3 2-8 1-5 9-8 4-5 4-7

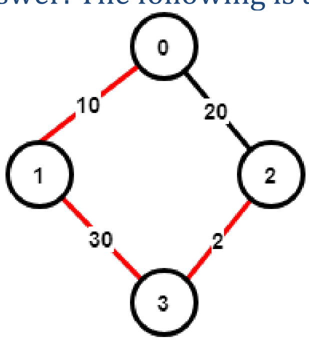
Which of these graphs have Euler cycles (cycles that visit each edge exactly once)? Which of them have Hamilton cycles (cycles that visit each vertex exactly once except the starting and ending vertex)?

Answer:

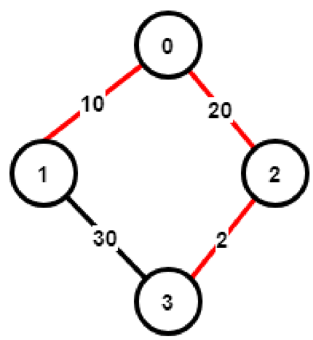
None of the graphs has Euler cycles. Graph b, c, and d have Hamilton cycles.

11. (3 Points) Give a counterexample that shows why the following strategy does not necessarily find the MST: 'Start with any vertex as a single-vertex MST, then add V-1 edges to it, always taking next a min-weight edge incident to the vertex most recently added to the MST.'

Answer: The following is a counter example



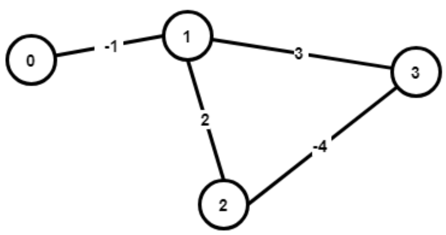
Incorrect MST with weight: 42
Start from 0, add 0-1, 1-3, 3-2 to MST



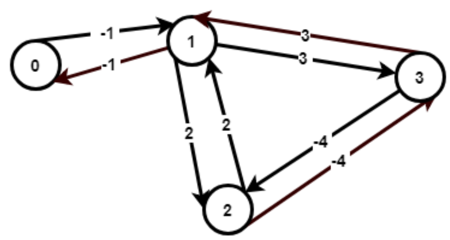
Correct MST with weight: 32

12. (3 Points) Suppose that we convert an EdgeWeightedGraph into an EdgeWeightedDigraph by creating two DirectedEdge objects in the EdgeWeightedDigraph (one in each direction) for each Edge in the EdgeWeightedGraph with the same edge weight for each direction and then use the Bellman-Ford algorithm to find a SPT. Explain why this approach fails.

Answer: This algorithm does not work because when there are negative edge weights, this approach can create negative cycles in converted digraph. If negatives cycles are reachable from the source vertex, such as 0 in the following graph, SPT does not exist.



Original Undirected Graph



Converted Digraph

Submission Note

1) For written part of the questions:
   a) Write your answers inside a text document (in plain text, MS Word, or PDF format)
   b) Name the file as firstname.lastname.assignment5.txt(doc, docx, or pdf) with proper file extension
3) Submit both of your text document via Canvas course web site.
4) Due Dec 8, 11:59 PM