1. Acknowledge all sources and collaborations. If you do not give an acknowledgement statement, your assignment may not be graded.

2. On the graph G, for source D, for the first 3 iterations of the Bellman-Ford loop, for each vertex v: show v's distance from D after the iteration.

```
G = {'A': [['C',2],['D', 7],['F',1]],
     'B': [['D',1],['E',-2],['F',3]],
     'C': [['A',2],['E', 1],['F',4]],
     'D': [['A',7],['B',-1],['F',3]],
     'E': [['B',3],['C', 1],['F',6]],
     'F': [['A',1],['B', 3],['C',4],['D',-2],['E',6]]}
```

distance from D	A	B	C	D	E	F
after it'n 0	---	---	---	0	---	---
after it'n 1						
after it'n 2						
after it'n 3						

3. Consider the Bellman-Ford algorithm from source vertex s on a weighted directed graph G. Let v be a vertex in G, and assume that (s, q, a, v) is a shortest path from s to v in G, with distance 19. Assume that `update(a,v)` has been called (and finished), and at some point previous to that `update(q,a)` was called, and at some point previous to that `update(s,q)` was called. What can you say about the current value of `dist[v]`? Prove your answer.

4. Modify the webnotes version of the Bellman-Ford algorithm so that it detects the presence of a negative weight cycle. You should only have to change a few lines of code. The output should be exactly as in the webnotes version, except that once a negative weight cycle is detected, a message should be printed and execution should halt.

   (i) Give the lines of your program that are different from that of the webnotes.

   (ii) Give the output of your program from this graph:

   `G ={'S': [['A',1]], 'A': [['B',1]], 'B': [['S',-3]]}`

5. A graph has preorder sequence `FJDEKBLGNIHAMC` and postorder sequence `JDFLBNGKEHCMAI`.

   (i) From the preorder sequence, how do you know that the root of the first tree in the dfs recursion forest is `F`?

   (ii) From (i) and the postorder sequence, how do you know that the vertices of the first tree in the dfs recursion forest are `FJD`?

   (iii) Draw the complete dfs recursion forest. The first tree should be on the left, the next tree to its right, etc. Also, for each node, its first child is drawn leftmost, the next child is drawn to the first child's right, etc.

6. A topological sorting of an acyclic directed graph is an ordering of the vertices, such that all arcs in the digraph go forward in the ordering.

   (i) Explain why the first vertex in a topological sorting must be a source in the digraph.

   (ii) For the digraph with vertices A,...,J and arcs (A,C), (B,D), (B,G), (B,J), (D,C), (D,F), (E,C), (E,F), (G,F), (H,B), (H,J), (I,A), (I,B), (I,J), give a topological sorting.

(iii) Give an efficient algorithm to find a topological sorting. Assume that the input digraph is acyclic. Assuming that the digraph is implemented as an adjacency matrix, give your algorithm's runtime.

7. For the undirected complete graph whose edge weights are shown below, (i) trace Kruskal's MST algorithm: show the order in which edges are picked, (ii) trace Prim's MST algorithm: show the order in which edges are picked.

	B	C	D	E	F	G	H	I	J	K
A	3	9	1	8	6	7	5	4	3	9
B		10	5	9	4	7	6	6	2	7
C			8	7	1	9	4	2	6	12
D				9	8	7	11	13	9	8
E					12	15	17	21	9	9
F						8	8	7	16	11
G							6	9	8	15
H								5	9	7
I									13	11
J										6