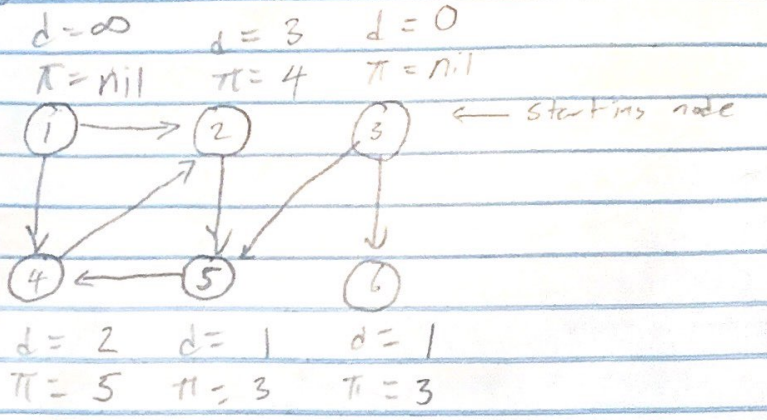
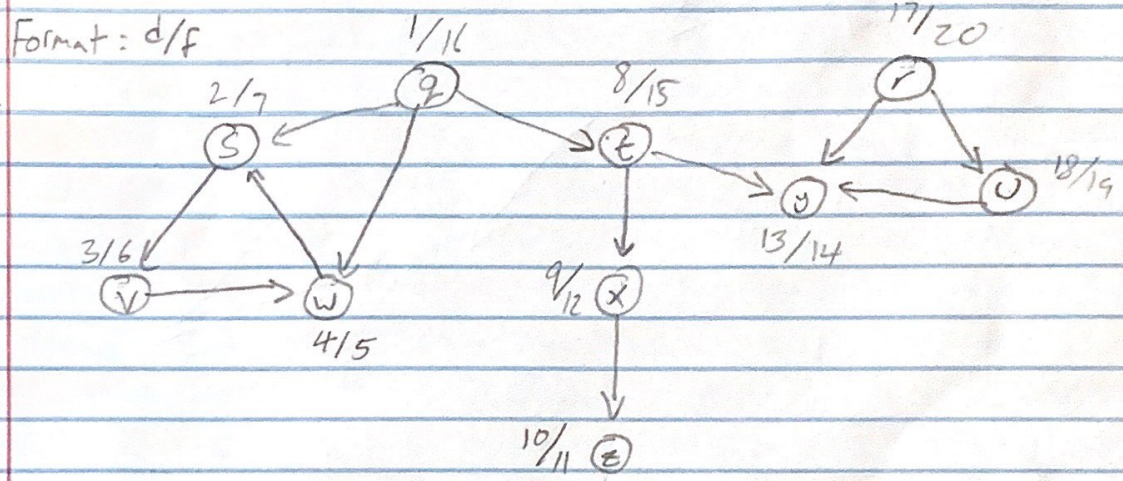
Algorithms HW 4 — Tae M

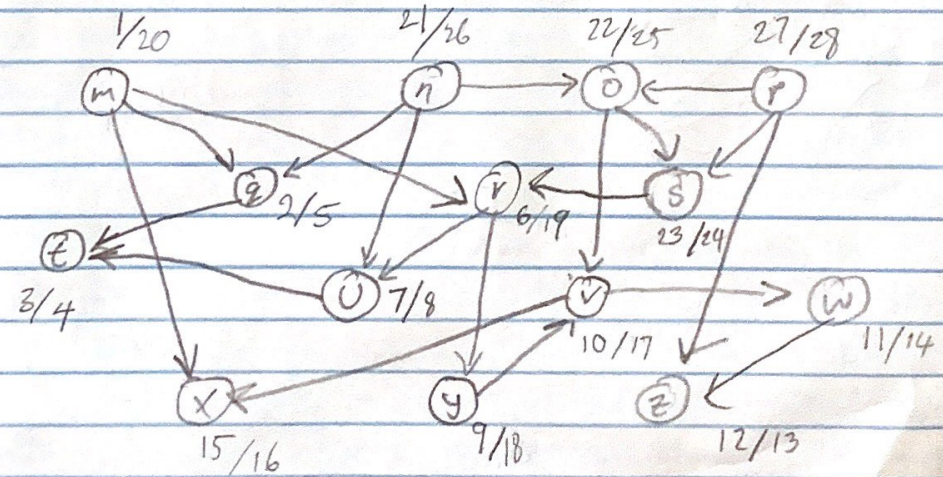
1.



2.



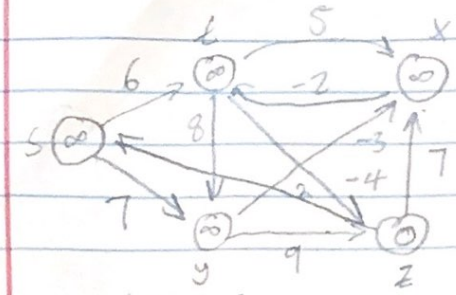
3.



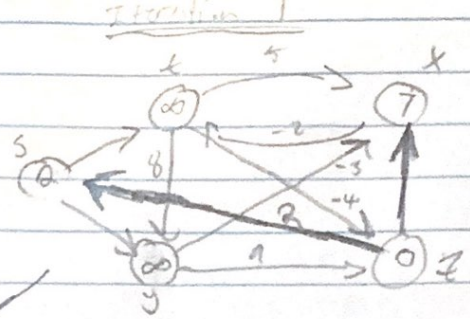
Topological order: P, N, O, S, M, r, y, V, X, W, z, U, q, t

|V| = 5 − 1 = 4 iterations

4.

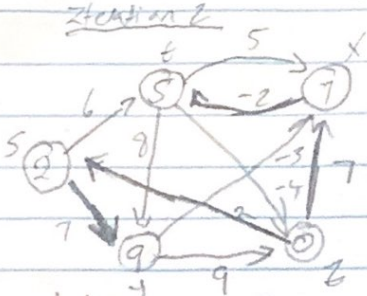


Iteration 1



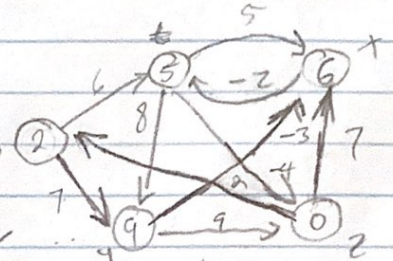
	d	π
s	2	z
t	∞	nil
x	7	z
y	∞	nil
z	0	nil

Iteration 2



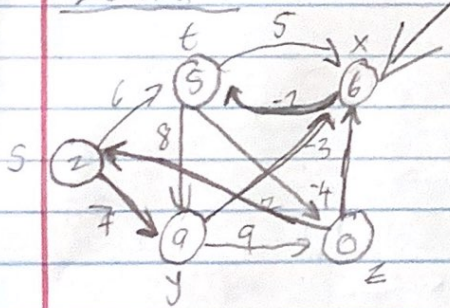
	d	π
s	2	z
t	5	x
x	7	z
y	9	s
z	0	nil

Iteration 3



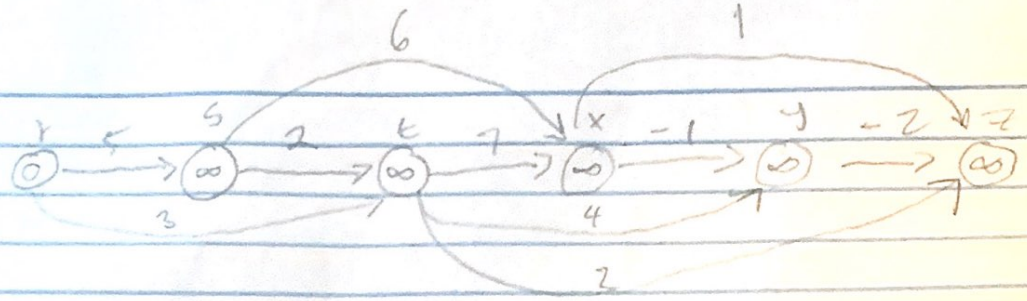
	d	π
s	2	z
t	5	x
x	6	z
y	9	s
z	0	nil

Iteration 4

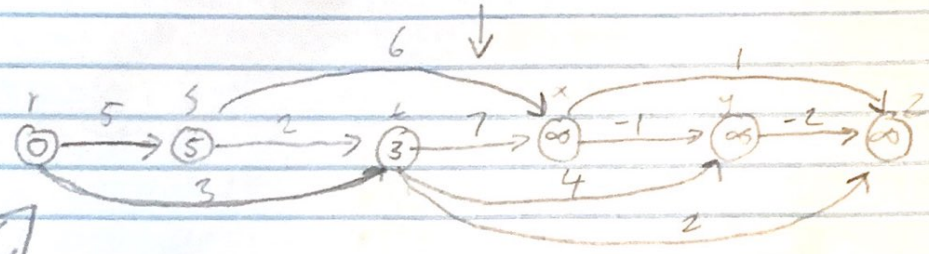


	d	π
s	2	z
t	4	x
x	6	y
y	9	s
z	0	nil

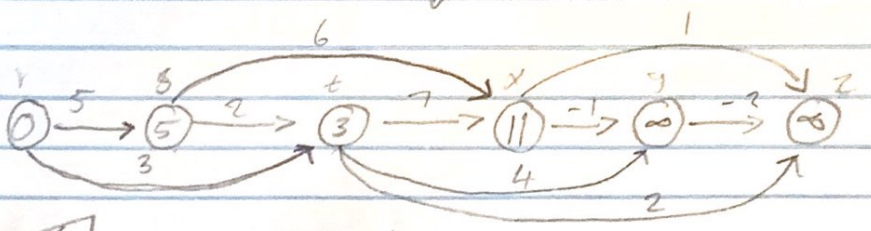
5.



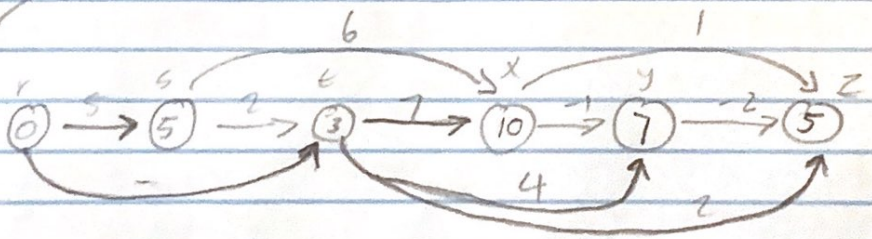
	d	π
r	0	nil
s	5	r
t	3	r
x	∞	nil
y	∞	nil
z	∞	nil



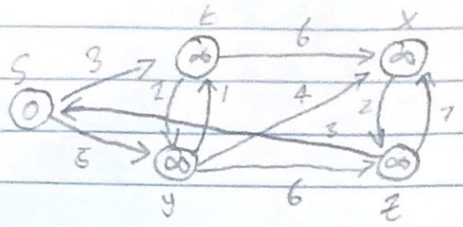
	d	π
r	0	nil
s	5	r
t	3	r
x	11	s
y	∞	nil
z	∞	nil

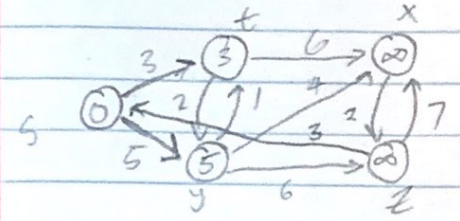


	d	π
r	0	nil
s	5	r
t	3	r
x	10	t
y	7	t
z	5	t

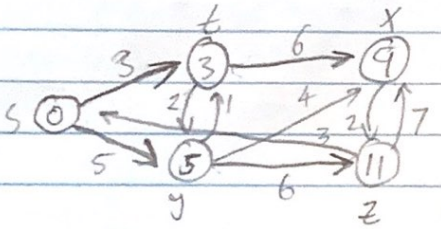


6 Vertex S as source:



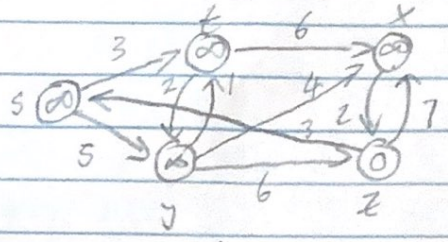


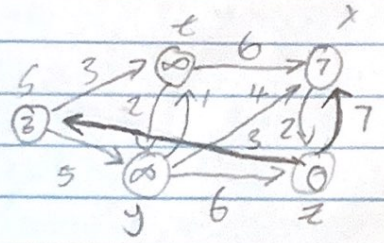
	d	π
s	0	nil
t	3	s
x	5	nil
y	∞	s
z	∞	nil



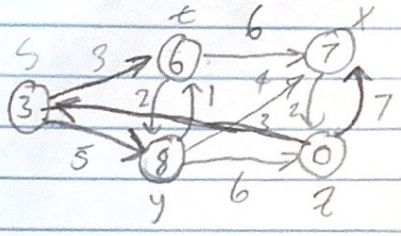
	d	π
s	0	nil
t	3	s
x	5	t
y	9	s
z	11	y

Vertex z as the source:





	d	π
s	3	z
t	∞	nil
x	7	z
y	∞	nil
z	0	nil



	d	π
s	3	z
t	6	s
x	7	z
y	8	s
z	0	nil

7. By adding edges one at a time, cycle is eventually likely to be detected. To deal with this we remove the edge that contains the most-cost edge. Keep iterating this process until all edges have been covered. To detect the most-cost edge we can run through root of the MST to each vertex. Comparing different paths, we are able to find the most-cost edge. Then update MST corresponding to it. The running time is $O(EV)$, since new vertex and incident edge is accounted for each iteration.