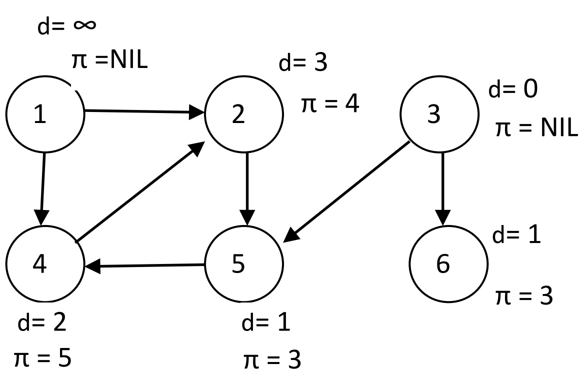
COMP 3270 Assignment 4 (100 points)
Due by 11:59PM on Monday, July 23, 2018

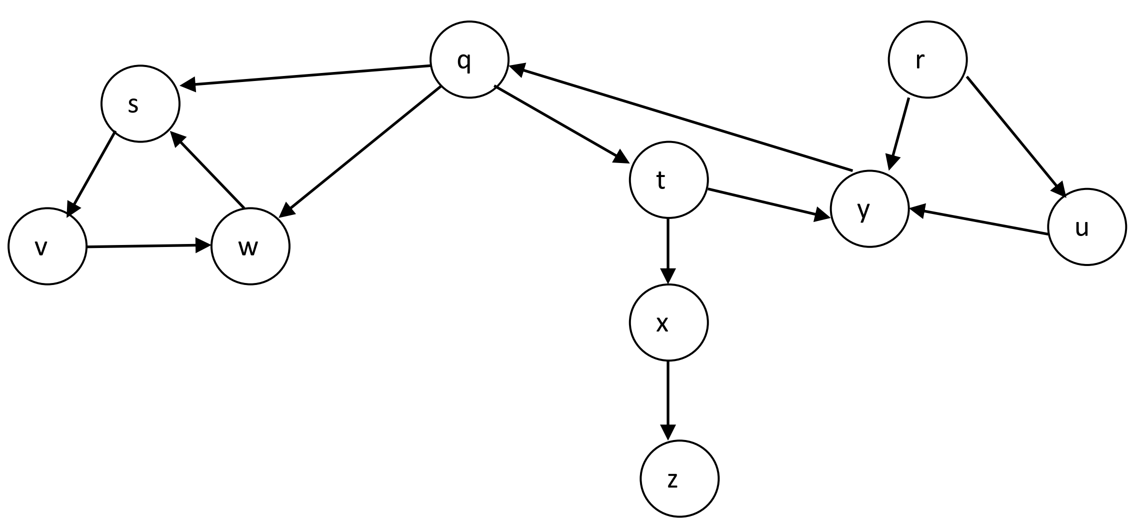
Instructions:

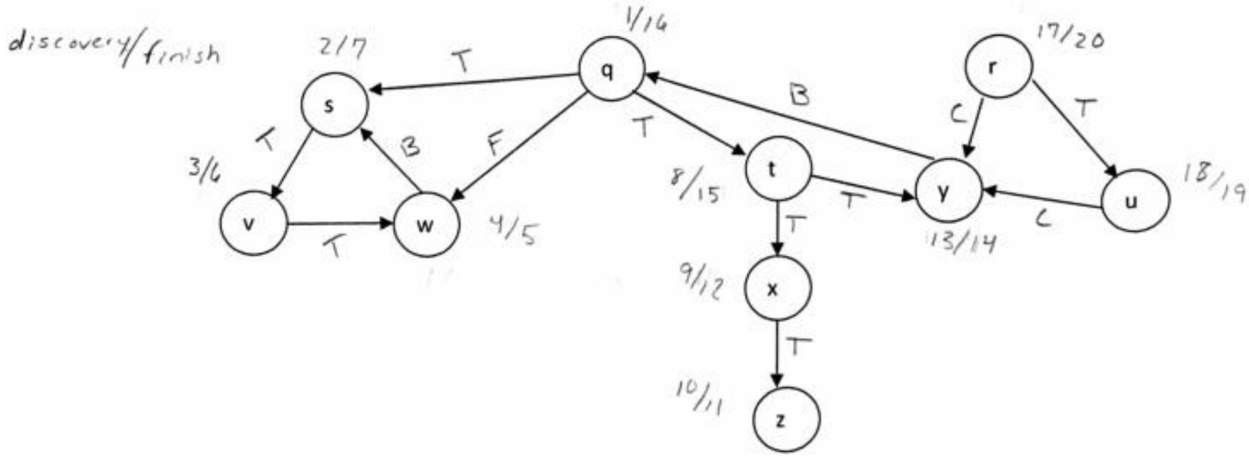
1. This is an individual assignment. You should do your own work. Any evidence of copying will result in a zero grade and additional penalties/actions.
2. Late submissions **will not** be accepted unless prior permission has been granted or there is a valid and verifiable excuse.
3. Think carefully; formulate your answers, and then write them out concisely using English, logic, mathematics and pseudocode (<u>no programming language syntax</u>).
4. Type your final answers in this Word document.
5. Don't turn in handwritten answers with scribbling, cross-outs, erasures, etc. If an answer is unreadable, it will earn zero points. **Neatly and cleanly handwritten submissions are acceptable**.

1. (15 points) Show d and π values that result from running Breadth First Search on the directed graph below using vertex 3 as the start node.

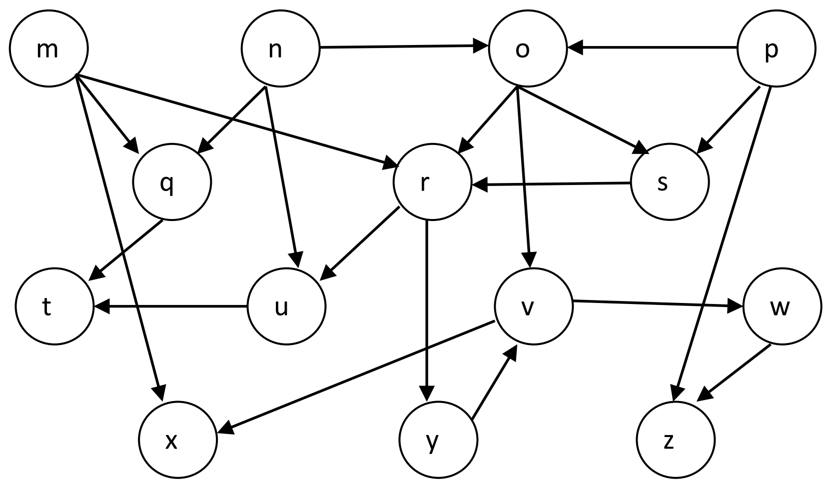


2. (10 points) Show how Depth First Search works on the graph below by marking on the graph the discovery and finishing times (d and f) for each vertex and the classification of each edge. Assume that the for loops in DFS and DFS-VISIT consider vertices alphabetically.





3. (15 points) List the vertices of the graph below in Topological Order, as produced by the Topological Sort algorithm. Assume that the for loops in DFS and DFS-VISIT consider vertices alphabetically.



p	n	o	s	m	r	y	v	x	w	z	u	q	t
27/28	21/26	22/25	23/24	1/20	6/19	9/18	10/17	15/16	11/14	12/13	7/8	2/5	3/4

4. (15 points) Do Problem 24.1-1 (p. 654) (you do not have to do the last part, i.e., running the algorithm again after changing an edge weight).

1st Iteration

	z	s	t	x	y
d:	0	$\cancel{\infty}$ 2	$\cancel{\infty}$ 8	$\cancel{\infty}$ 7	$\cancel{\infty}$ 9
π:	NIL	~~NIL~~ z	~~NIL~~ s	~~NIL~~ z	~~NIL~~ s

2nd Iteration

	z	s	t	x	y
d:	0	2	$\cancel{8}$ 5	$\cancel{7}$ 6	9
π:	NIL	z	~~s~~ x	~~z~~ y	s

3rd Iteration

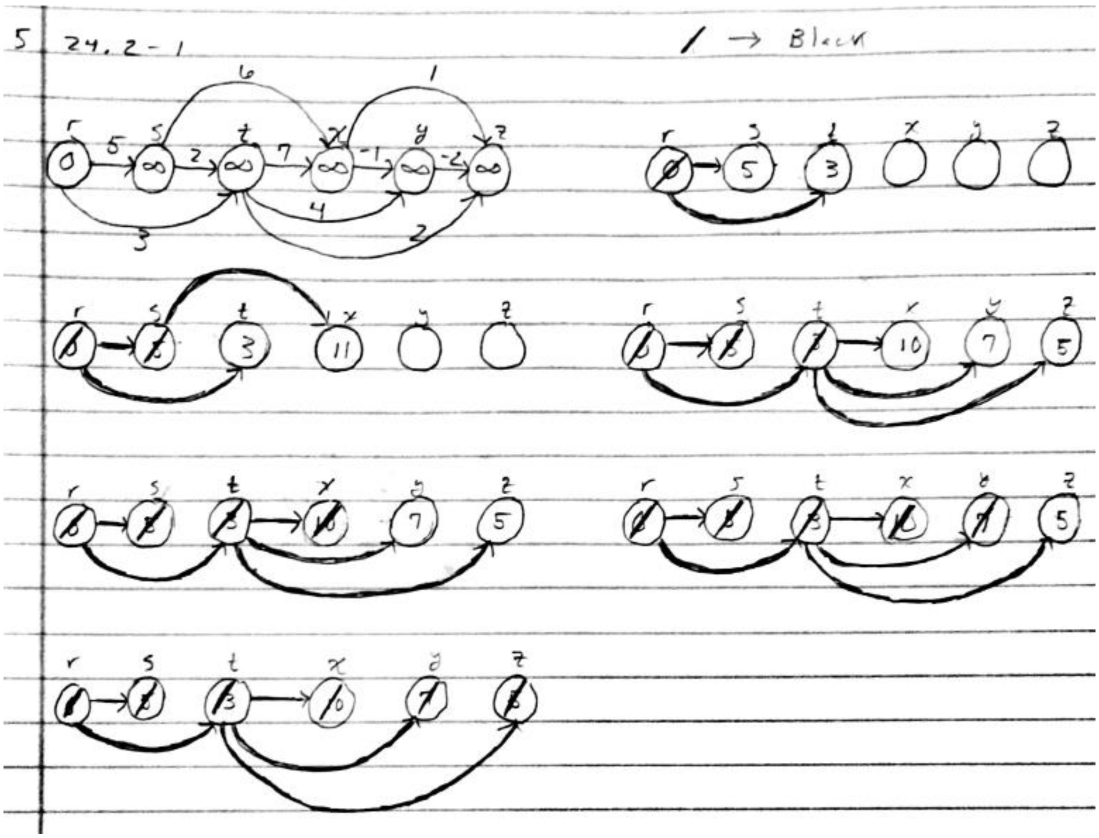
	z	s	t	x	y
d:	0	2	$\cancel{5}$ 4	6	9
π:	NIL	z	x	y	s

4th Iteration

	z	s	t	x	y
d:	0	2	4	6	9
π:	NIL	z	x	y	s

5. **(15 points)** Do Problem 24.2-1 (p. 657). Show the results similar to Fig. 24.5.

Note*: A vertex being slashed means it is now black.

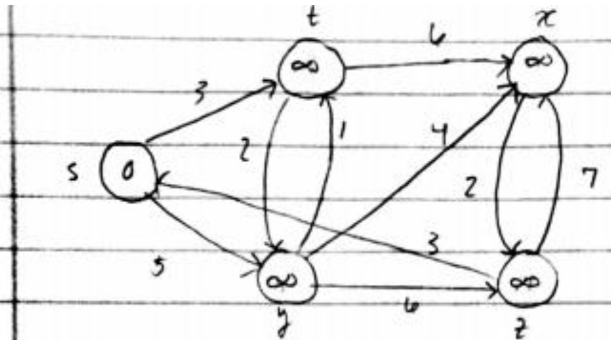


6. **(20 points)** Do Problem 24.3-1 (p. 662).

Note*: To be more precise, the set S would be updated before the distance and pi values are changed. Also, a vertex being slashed means it is now black.

With S as source

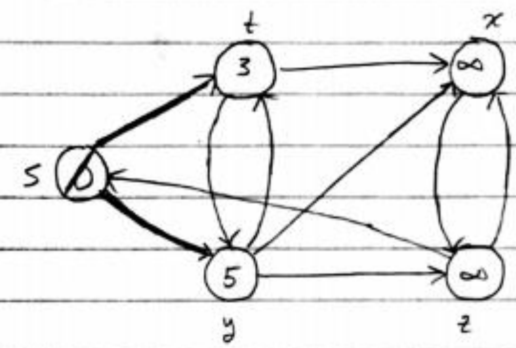
6



Before first iteration

	s	t	x	y	z
d:	0	∞	∞	∞	∞
π:	NIL	NIL	NIL	NIL	NIL

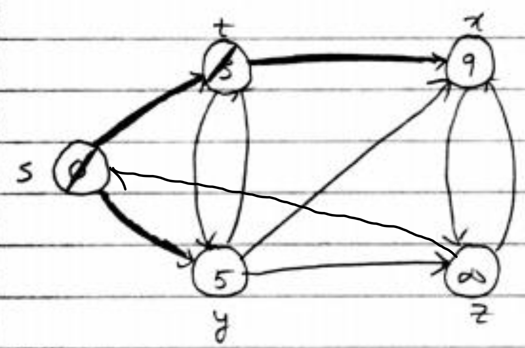
S = { }



First While-Loop Iteration

	s	t	x	y	z
d:	0	3	∞	5	∞
π:	NIL	S	NIL	S	NIL

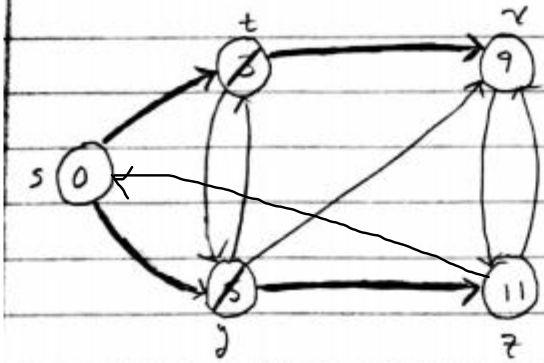
S = {s}



Second While-Loop Iteration

	s	t	x	y	z
d:	0	3	9	5	∞
π:	NIL	S	t	S	NIL

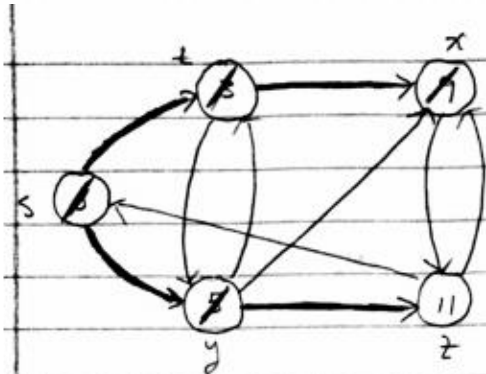
S = {s, t}



Third While-Loop Iteration

	s	t	x	y	z
d:	0	3	9	5	11
π =	NIL	s	t	s	y

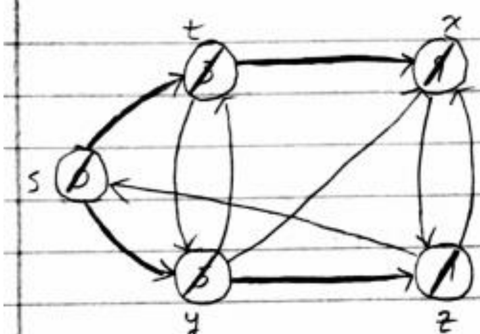
S = {s, t, y}



Fourth While Loop Iteration

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

S = {s, t, y, x}

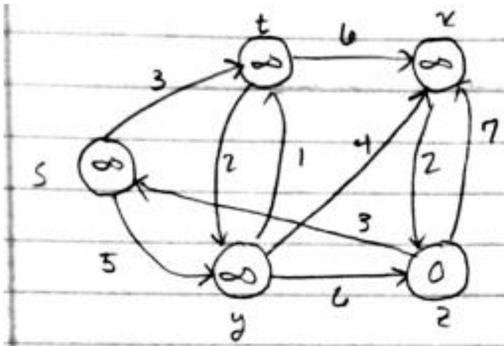


Fifth While Loop Iteration

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

S = {s, t, y, x, z}

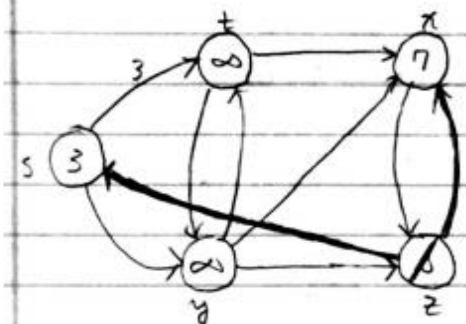
With Z as source



Before First Iteration

d:	∞	∞	∞	∞	0
	s	t	x	y	z
π:	NIL	NIL	NIL	NIL	NIL

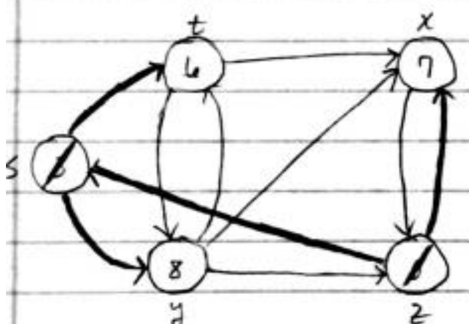
S = { }



First While Loop Iteration

d:	3	∞	7	∞	0
	s	t	x	y	z
π:	z	NIL	z	NIL	NIL

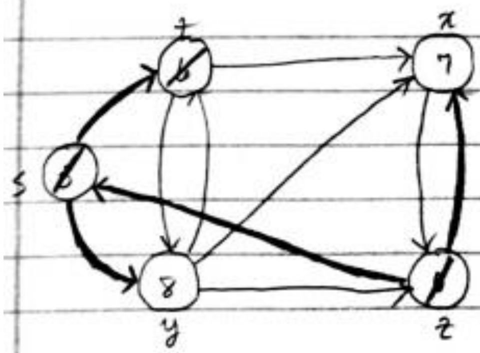
S = {z}



Second While Loop Iteration

d:	3	6	7	8	0
	s	t	x	y	z
π:	z	s	z	s	NIL

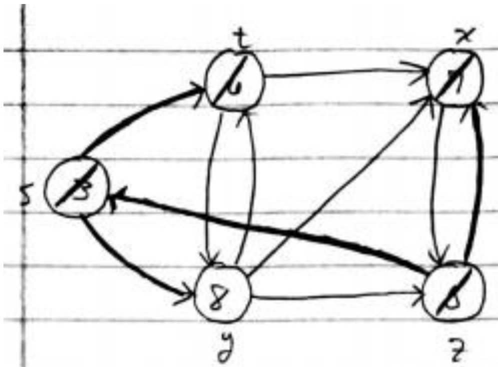
S: {z, s}



Third While Loop Iteration

d:	3	6	7	8	0
	s	t	x	y	z
π:	z	s	z	s	NIL

S = {z, s, t}



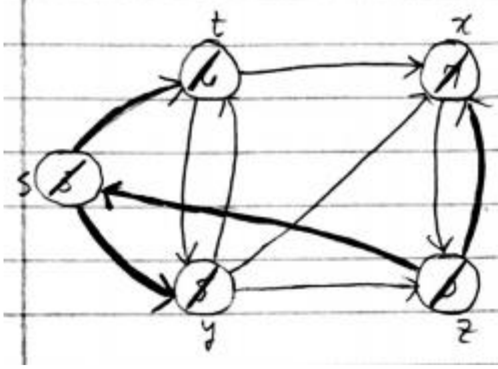
Fourth While Loop Iteration

d: 3 6 7 8 0

s t x y z

π = z s z s NIL

S: {z, s, t, x}



Fifth While Loop Iteration

d = 3 6 7 8 0

s t x y z

π = z s z s NIL

S: {z, s, t, x, y}

(7) (10 points) Suppose that a graph G has a Minimum Spanning Tree (MST) computed. How quickly can we update the MST if we add a new vertex and incident edges to G. Propose and outline a strategy and present an algorithm (you can reuse graph algorithms covered in class as building blocks as part of your solution) and evaluate its asymptotic complexity.

When adding a new vertex, one can use depth first search to discover a cycle containing the new vertex. This would have to be done for every edge connecting the new vertex. At least two of the new edges would have to be considered at a time to induce a cycle back to the new vertex. During the process of inducing a cycle, the algorithm would have to keep track of all the vertices visited. Once a cycle is found, remove the edge with the maximum weight. If multiple new edges induce a cycle that removes an edge with the same max weight, the smallest edge of the two would be selected.

Complexity: $\Theta(V + E)$. This is due to using depth first search to induce a cycle through each edge. The number of times Depth First Search will be ran is the number of new edges from the new vertex minus one. This is because two new edges will be used in the first iteration to induce a cycle.