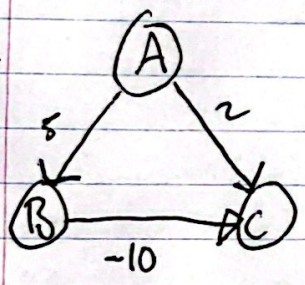
Assignment #6

Memet R
13095155a

Q1.

	*vert	dist	Pred
initialize	S	0	N/A
Step 1	U	3	S
Step 2	V	5	U
Step 3	t	6	V

Q2.

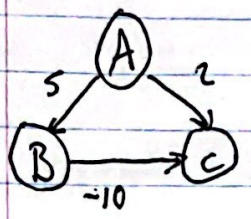


This graph will produce the path A → C → A → B since the Algorithm will traverse the 'cheapest' path first. this will mean A → C is visited first. then C → B is not possible so the Algorithm will back track to A and travel to b (A → B) since c has alredy been visited the Algorithm will finish with result A → C, A → B.

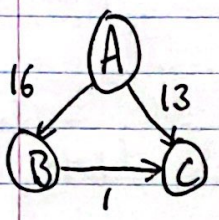
The Algorithm assumes the path it took is lowest cost and wont traverse B → C even though it produces the lowest cost. thus the Algorithm never finds A → B → C

Hilroy

Q3



Using the same graph we increase edge values By constant c where c equals 11



now the updated Algorithm has Dijkstra run on it but produces the same result. A → C is traversed first then A → B, But since C has been visited the Algorithm concludes A → C A → B is the shortest path and does not find path A → B → C

Q4. not, if a spanning tree T is created by vertices $\{V_1, V_2, V_3 \ldots V_n\}$ by adding a new vertex V_{n+1} we Break the minimum spanning tree if V_{n+1} has a lower weight than (V_1, V_2)

Hilroy

Q5 Start with node (t) Visited = { }

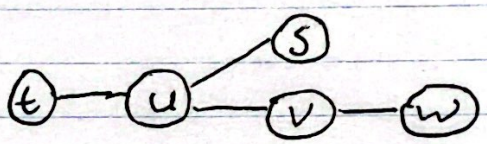
(2) Greedy Choice we add (u) Visited = { t, u }

(3) edge (u, s) is the next lowest so Select (s)
Visited = { t, u, s }

(4) next (u, v) has the lowest weight so (v) is selected
Visited = { t, v, s, v }

(5) next (v, w) is selected and (w) added
visited = { t, u, s, v, w }

add done



is our spanning tree

Priority Queue

(1) (t, u) 2
(t, s) 5
(t, v) 7

(2) (u, s) 3
(u, v) 4
(u, w) 5
(t, s) 5
(t, v) 7

(3) (u, v) 4
(u, w) 5
(t, s) 5
(s, v) 6
(t, v) 7

(4) (v, w) 4
(u, w) 5
(t, s) 5
(s, w) 6
(t, v) 7

Hilroy

Table 1

	4	8	12	15	18	20
Backtrack	7 0.001420021057	12 0.001426219940	22 0.014761924743	42 0.093585968017	47 0.988433122634	56 5.20864176750
1st Optimization	7 0.000590085983	12 0.001451015472	22 0.0111470222	42 0.0971131324	47 0.9805781	56 4.966504
2nd Optimization						