CS 300 HOMEWORK 5

Q1)

Step 1

	Weight	Known
A	infinity	False
B	infinity	False
C	2	False
D	infinity	False
E	infinity	False
F	infinity	False
G	0	True
H	2	False

Step 2

	Weight	Known
A	Infinity	False
B	6	False
C	2	True
D	infinity	False
E	infinity	False
F	8	False
G	0	True
H	2	False

Step 3

	Weight	Known
A	6	False
B	6	False
C	2	True
D	infinity	False
E	infinity	False
F	8	False
G	0	True
H	2	True

Step 4

	Weight	Known
A	6	True
B	6	False
C	2	True
D	infinity	False
E	15	False
F	8	False
G	0	True
H	2	True

Step 5

	Weight	Known
A	6	True
B	6	True
C	2	True
D	infinity	False
E	7	False
F	8	False
G	0	True
H	2	True

Step 6

	Weight	Known
A	6	True
B	6	True
C	2	True
D	10	False
E	7	True
F	8	False
G	0	True
H	2	True

Step 7

	Weight	Known
A	6	True
B	6	True
C	2	True
D	10	False
E	7	True
F	8	True
G	0	True
H	2	True

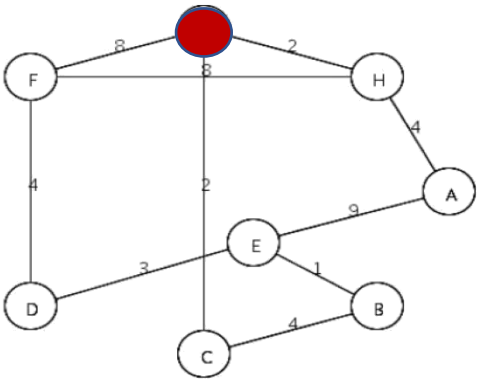
Step 8

	Weight	Known
A	6	True
B	6	True
C	2	True
D	10	True
E	7	True
F	8	True
G	0	True
H	2	True

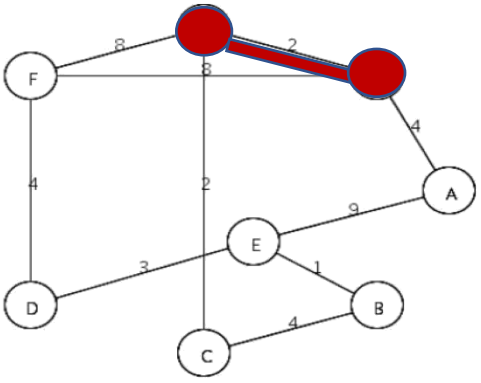
Q2)

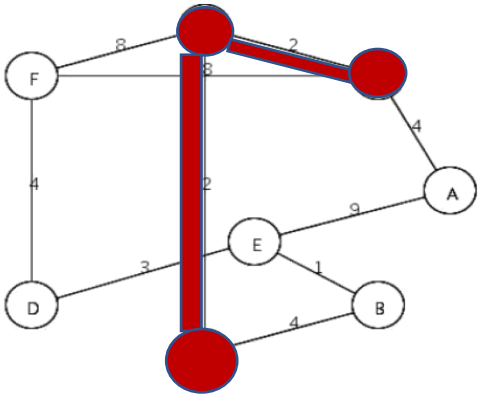
Selected vertices and edges are shown in red for every step.

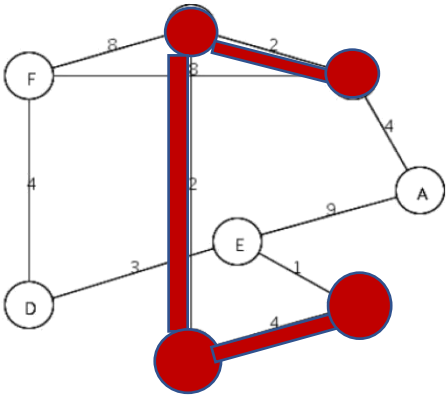
Step 1

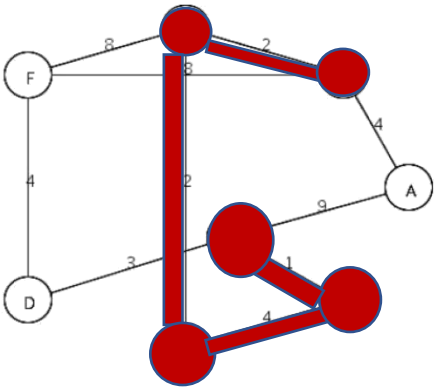


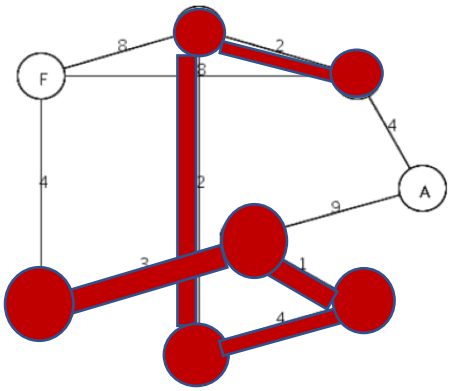
Step 2

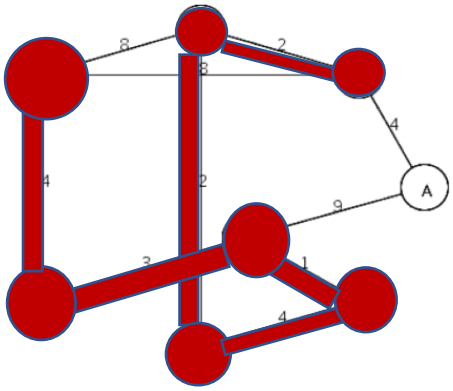


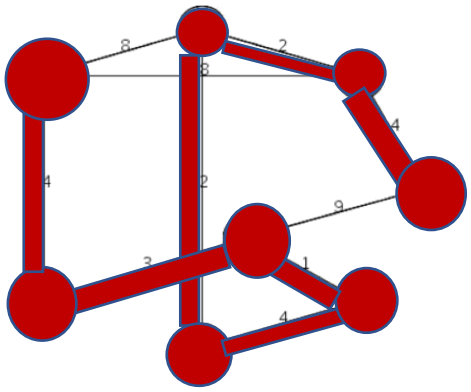








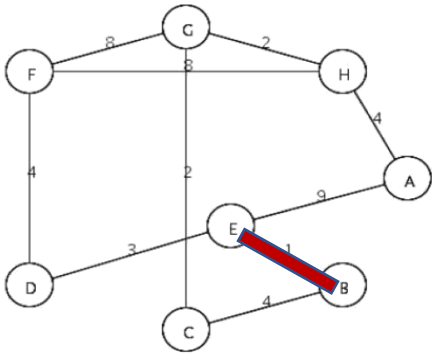

8
2
8
4
4
2
A
9
3
1
4


8
2
8
4
4
2
9
3
1
4

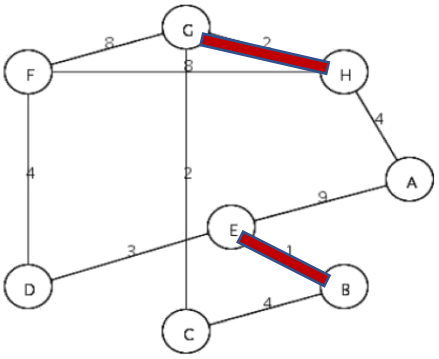
Q3)

The selected edges are shown in red at every step of the program.

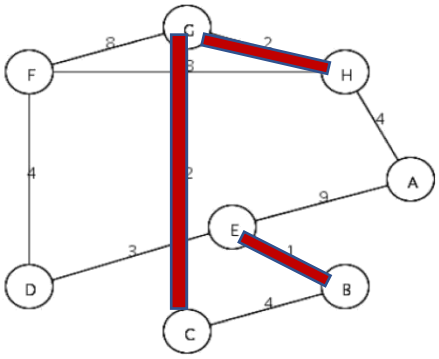
Step 1



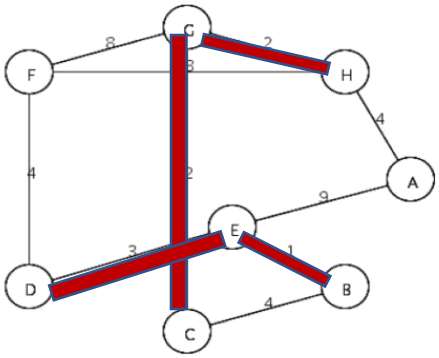
Step 2



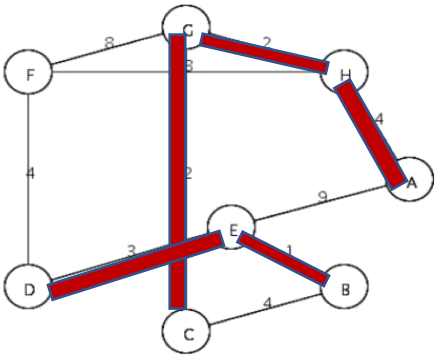
Step 3



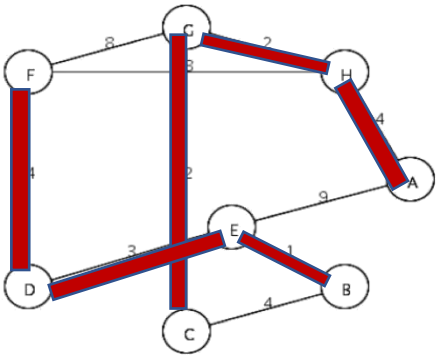
Step 4



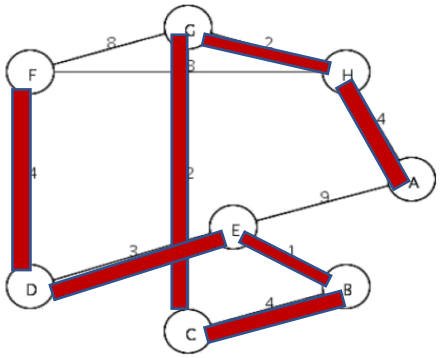
Step 6



Step 7



Step 8



Q4)

Step 1: Queue: S

Step 2: Queue: ~~S~~ – B – A – D

Step 3: Queue: ~~S~~ – ~~B~~ – A – D (We can not add new vertex to the queue since there is no adjacent vertex that is unknown)

Step 4: Queue: ~~S~~ – ~~B~~ – ~~A~~ – D – C

Step 5: Queue: ~~S~~ – ~~B~~ – ~~A~~ – ~~D~~ – C – E – T – F

Step 6: Queue: ~~S~~ – ~~B~~ – ~~A~~ – ~~D~~ – ~~C~~ – E – T – F

Step 7: Queue: ~~S~~ – ~~B~~ – ~~A~~ – ~~D~~ – ~~C~~ – ~~E~~ – T – F – G

Step 8: Queue: ~~S~~ – ~~B~~ – ~~A~~ – ~~D~~ – ~~C~~ – ~~E~~ – ~~T~~ – F – G

Step 9: Queue: ~~S~~ – ~~B~~ – ~~A~~ – ~~D~~ – ~~C~~ – ~~E~~ – ~~T~~ – ~~F~~ – G

Step 10: Queue: ~~S~~ – ~~B~~ – ~~A~~ – ~~D~~ – ~~C~~ – ~~E~~ – ~~T~~ – ~~F~~ – ~~G~~

Our breadth first search terminates until the queue is empty.

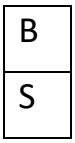
If I print the vertices after I dequeue it from the queue, the output will be below.

Output: S – B – A – D – C – E—T – F – G

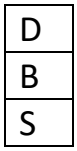
Q5)

a) Step 1: Stack:

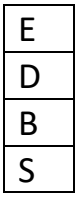
Step 2: Stack:



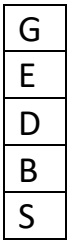
Step 3: Stack



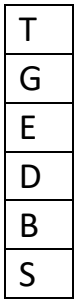
Step 4: Stack:



Step 5: Stack:

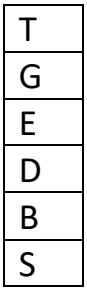


Step 6: Stack:

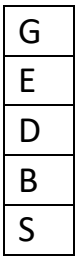


Step 7: Stack:

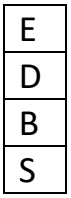
Step 8: Stack:



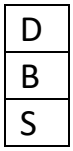
Step 9: Stack:



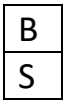
Step 10: Stack:



Step 11: Stack:



Step 12: Stack:



Step 13: Stack:

S

Step 14: Stack:

A
S

Step 15: Stack:

C
A
S

Step 16: Stack:

A
S

Step 17: Stack:

S

Our DFS terminate when there is no element in the stack.

Output: S – B – D – E – G – T – F – A – C (If I print the vertices that I visit that are unknown).

b) Post Orders:

S: 9

B: 6

A: 8

D: 5

C: 7

E: 4

G: 3

T: 2

F: 1

c) Pre Orders:

S: 1

B: 2
A: 8
D: 3
C: 9
E: 4
G: 5
T: 6
F: 7

d)

Tree arcs: (S, B), (B, D), (D, E), (E, G), (G, T), (T, F), (S, A), (A, C)

Forward arcs: (S, D), (D, T), (D, F),

Backward arcs: (B, S), (G, E)

Cross arcs: (C, E), (E, T), (C, D)

Q6) Since the vertex A and G are degree 0, I enqueue them in to the queue(I picked the first one arbitrarily), after that, when I dequeue the vertex from the queue, I looked the adjacent vertices of that vertex and if the vertex's indegree is zero when I decrement the indegree of that vertex by one I add it to the queue.

A – G – B – D – C – E – F

Efe Şencan 25083