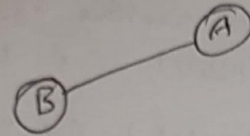
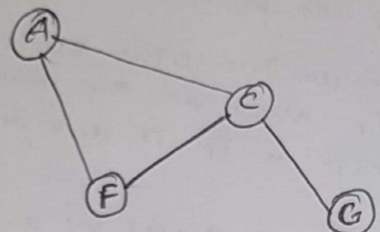
1

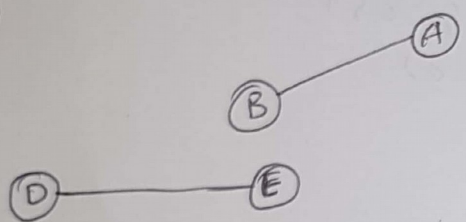
a



b



c



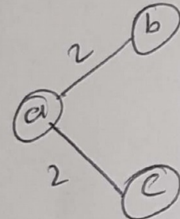
d NO, eventhough we have 3 vertices (A,B,F) in H that exists in G, the edges are not equal. H has three edges & the subset of G has four edges.

(2) Let us show that any two minimum spanning trees T_1 & T_2 of G are the same tree. Let (u,v) be an arbitrary edge of T_1. From the proof, any edge in an MST is a light edge crossing some cut of the graph. Let $(S, V-S)$ be a cut for which (u,v) is a light edge.

Consider the edge $(x,y) \in T_2$ crossing $(S, V-S)$. (x,y) must exist, as otherwise T_2 would not be a spanning tree. (x,y) must also be a light edge, as otherwise T_2 would not be a minimum spanning tree.

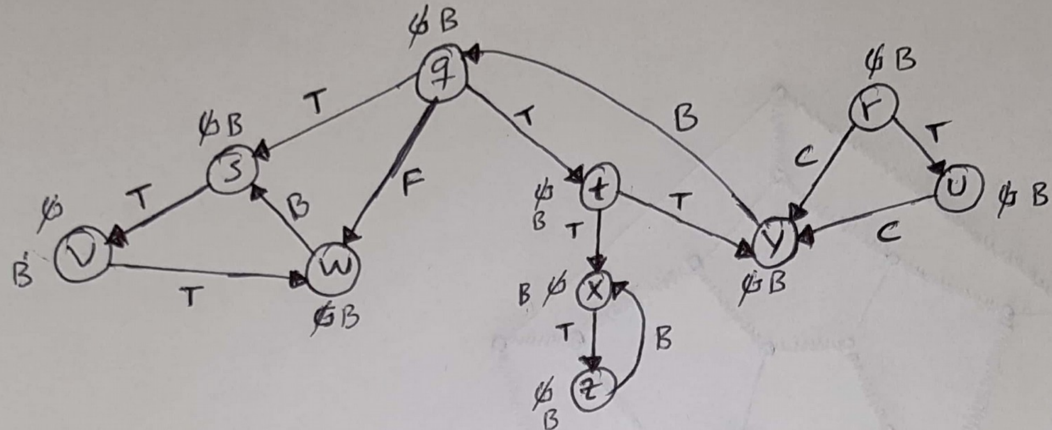
By problem statement, there is an unique light edge crossing any cut of G. Thus, $(u,v) \in T_1$ & $(x,y) \in T_2$ must be the same edge. As (u,v) is an arbitrary edge of T_1, every edge in T_1 is also in T_2 & thus T_1 & T_2 are the same tree.

The converse, is not true, as demonstrated by a counter example.

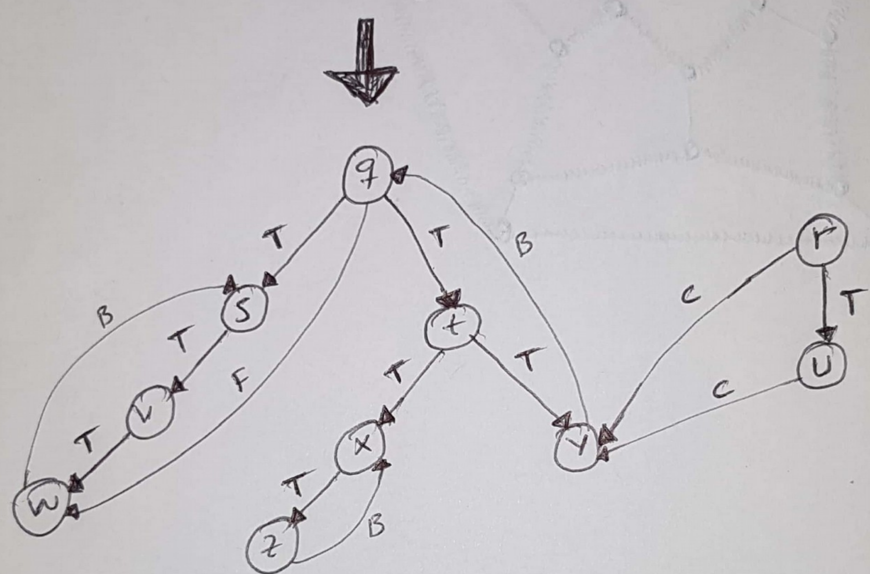


Here, the graph is its own (unique) MST, but the cut $(\{a\}, \{b, c\})$ has two light edges - (a,b) & (a,c).

Q1. (A)

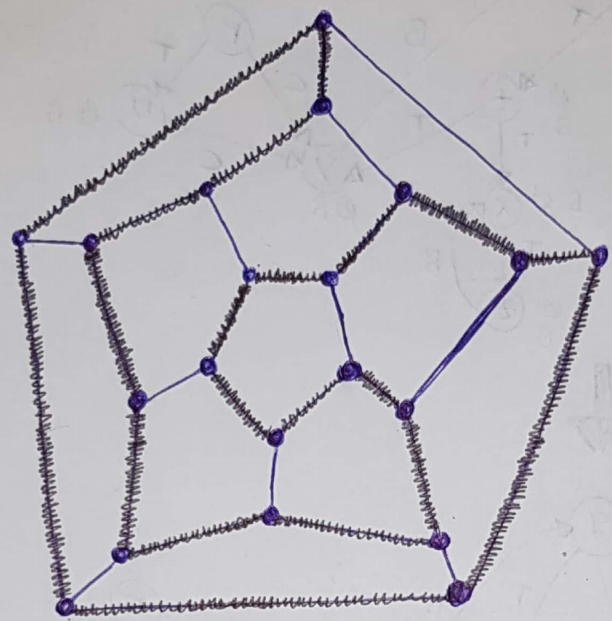


ϕB
q
T
T
B
ϕB
r
s
ϕB
C
T
ϕ
T
B
F
ϕ
B
t
T
u
ϕB
v
w
B
T
B ϕ
x
C
y
ϕB
T
ϕB
B
z
ϕ
B



q
T
T
B
r
B
s
T
t
C
T
T
F
T
T
u
v
C
x
y
w
T
z
B

(3)



(Q4) Yes, we can compute maximum spanning tree using kruskal or prim's Algorithm with edges sorted in decreasing order.

Also, Another alternative to use kruskal or Prim's algorithm for maximum spanning tree is multiplying the weight of each edge by -1 (negating the edges).

⇒ For Example we can follow the following the following steps to compute maximum spanning tree by using kruskal's Algorithm

(1) Sort the edges of G into decreasing order by weight. Let T be the set of edges comprising the maximum weight spanning tree. Set $T = \emptyset$.

(2) Add the first edge to T.

(3) Add the next edge to T if & only if it does not form a cycle in T. If there are no remaining edges exit & report G to be disconnected.

(4) If T has n-1 edges (where n is the number of vertices in G) stop & output T. Otherwise go to step 3.