Algorithm Design and Algorithms
Homework #2

1 Q Ans:

Topological ordering defined as ordering of nodes so that all the edge points should be in forward direction i.e. if there is an edge (Vi, Vj) then "i" should be less than "j" i<j.

So one approach to solve this kind of problem is, find out the possible orderings. Here in this case we will get 6.5.4.3.2.1 = 720 combinations. Then we should be able to check the topological ordering for each combination, which usually eats huge amount of time.

As the above approach takes huge amount of time, we think about other possibilities. One other option is, as from the definition, the topological ordering starts with the edge that doesn't have any edge coming into it and ends with the node which doesn't have any edge leaving it. So in our case, the topological ordering starts with the node "a" and ends with the node "f". In between these nodes, we have other 4 nodes (b, c, d, e) and these 4 nodes shouldn't arranged randomly. There are two constraints with respect to these 4 nodes: the edge (b, c) enforces the requirement that b must come before c; similarly in case of edge (d, e) d must come before e and the nodes b, c, d, e can be placed anywhere but the above two constraints should be satisfied as a whole.

And the possible topological orderings are:

a ,b ,c ,d ,e ,f

a ,d ,e ,b ,c ,f

a ,d ,b ,e ,c ,f

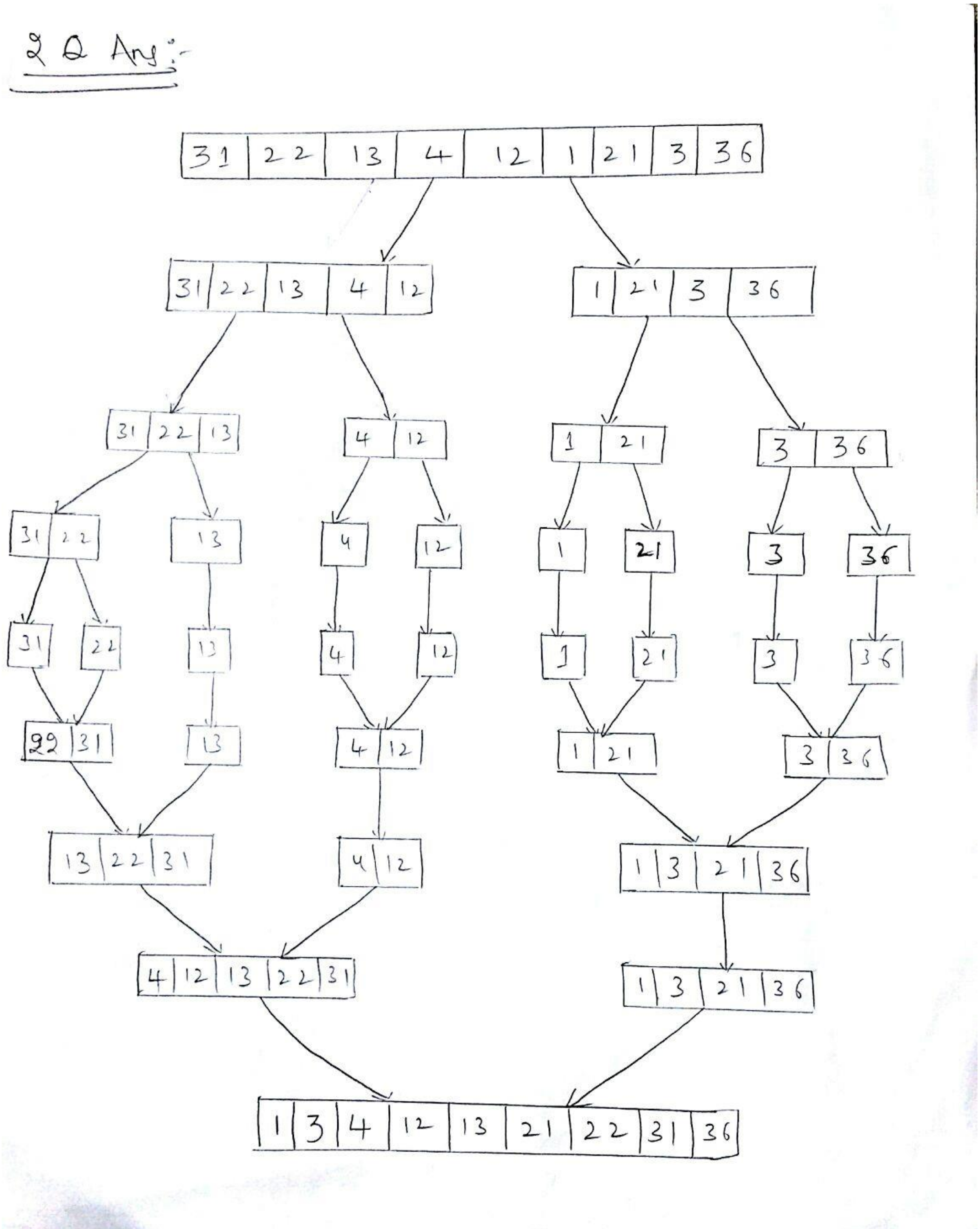
a ,b ,d ,c ,e ,f

a ,d ,b ,c ,e ,f

a ,b ,d ,e ,c ,f

Algorithm Design and Algorithms
Homework #2

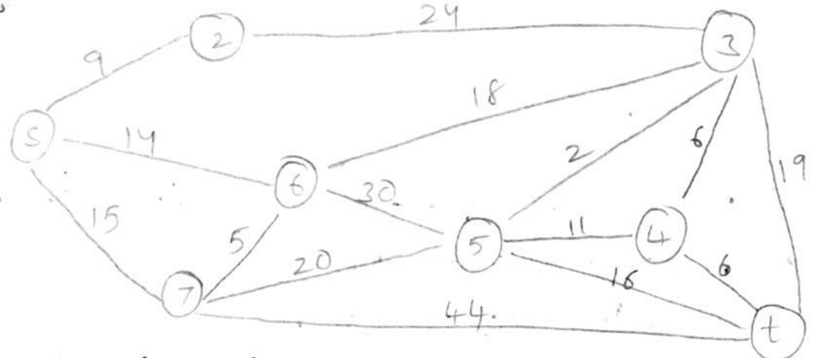
2 Q Ans:



3 Q Ans:

Algorithm Design and Algorithms
Homework #2

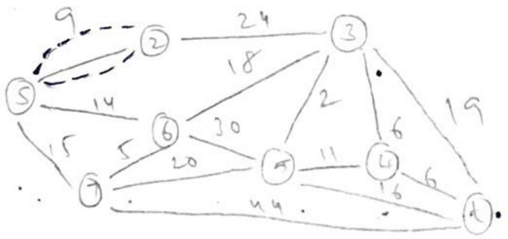
3Q Ans:-



The prims algorithm, will find out the minimum spanning tree of a graph. The method it uses is, on visiting the every node it adds the minimum edge from the set of edges going out from the node

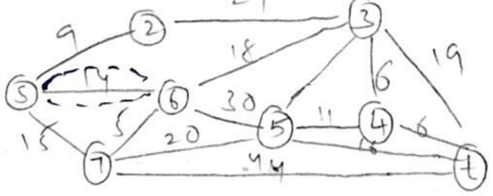
Step 1:- In the very first step, from the node 's', there are two 3 edges (S,2), (S,6), (S,7) with weights as follows (9,14,15). As '9' is minimum weight

edges => (S,2), (S,6), (S,7) => (9,14,15) => adds (S,2)



step 2:- It compares the edge weights

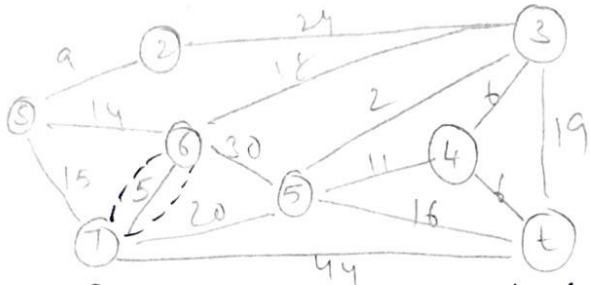
{(S,6), (S,7), (2,3)} => {14, 15, 24} => adds (S,6)



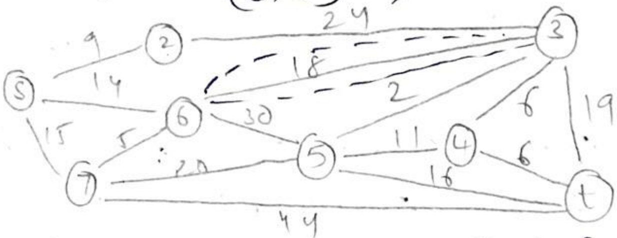
step 3: Compare the edges,

{(S,7), (2,3), (6,3), (6,5), (6,7)} => {15, 24, 18, 30, 5} => adds (6,7)

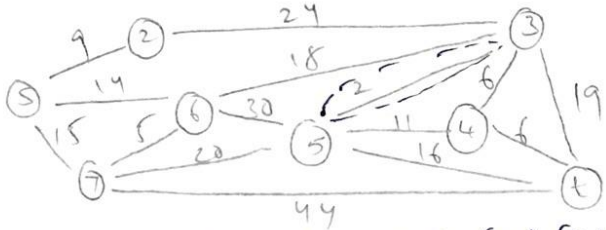
Algorithm Design and Algorithms
Homework #2



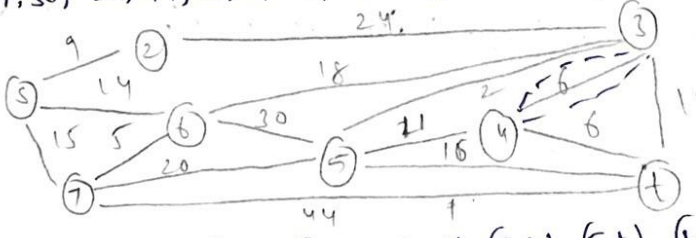
step4: {(S,7), (2,3), (6,3), (6,5), ~~(6,7)~~ (7,5), (7,44)} = {15, 24, 18, 30, ~~5~~ 20, 44}
(S,7) is the minimum weight. But it makes a cycle. So ~~so~~ the algorithm selects (6,3) => 18



step5: {(S,7), (2,3), ~~(6,3)~~, (6,5), (7,5), (7,44), (3,5), (3,4), (3,t)} =
{15, 24, 30, 20, 44, 2, 6, 19} adds => (3,5) => 2 weight

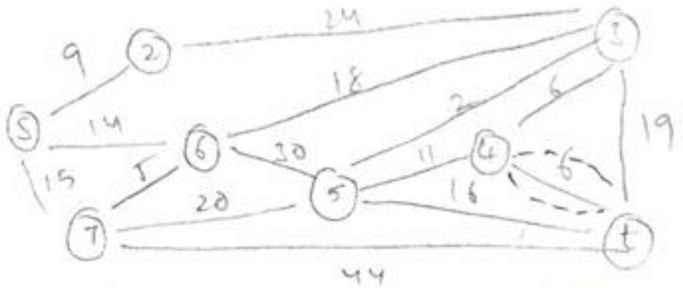


step6: {(S,7), (2,3), (6,5), (7,5), (7,44), (3,4), (3,t), (5,4), (5,t)} =>
{15, 24, 30, 20, 44, ~~2~~, 6, 19, 11, 16} => adds (3,4) => weight 6

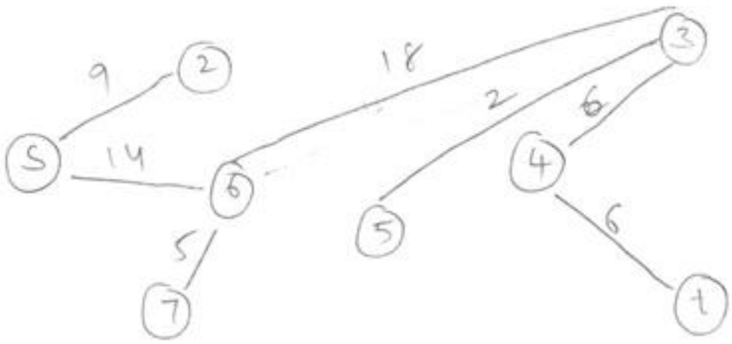


step7: {(S,7), (2,3), (6,5), (7,5), (7,44), (3,t), (5,4), (5,t), (4,t)}
=> {15, 24, 30, 20, 44, ~~6~~, 19, 11, 16, 6} => adds (4,t) => weight 6

Algorithm Design and Algorithms
Homework #2



Now, the ~~gra~~ algorithm has traversed all the nodes ~~cut~~
The total weight of this spanning tree is
= 9+14+5+18+2+6+6 = 60

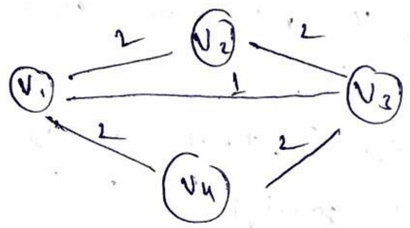


4 Q Ans:

Algorithm Design and Algorithms
Homework #2

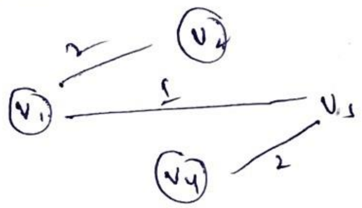
4Q. Ans:

Considering the graph, with the edge cost between (V_1, V_2), (V_2, V_3) & (V_3, V_4) is 2 and the cost between edges $(V_1, V_3) = 1$.

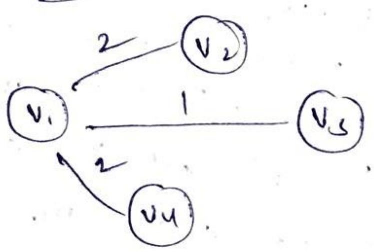


Frist will calculate the minimum spaning trees by applying the all the algorithms

a) Prims algorithm

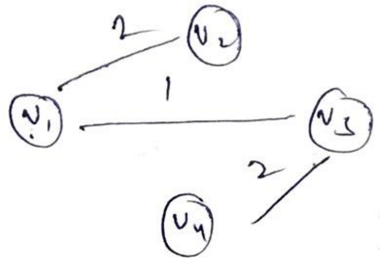


b) Kruskal algorithm

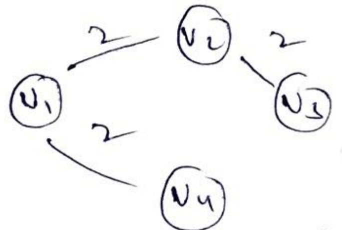


c) Reverse Delete Frist:

Algorithm Design and Algorithms
Homework #2



In all the tree algorithm, the minimum spaning tree Weight is $= 2+1+2 = 5$.
Now, will take a spaning tree T, that every edge $e \in T$, e belongs to ~~some~~ minimum cost spaning tree in G; ~~But~~ then. if we take another example, where minimum spaning tree T' is also minimum spaning tree to G'. then, we didn't consider edge '1' (V_1, V_3).



Here the overall weight is $2+2+2 = 6$
then T' is a spaning tree. and every edge present in T', spaning trees created
is also the edge in ~~previous~~ minimum spaning tree.
earlier. ~~But~~ So the T is not a minimum spaning tree.
So the ~~tree~~ spaning tree T with all the edges without
the minimum edge can't be minimum.