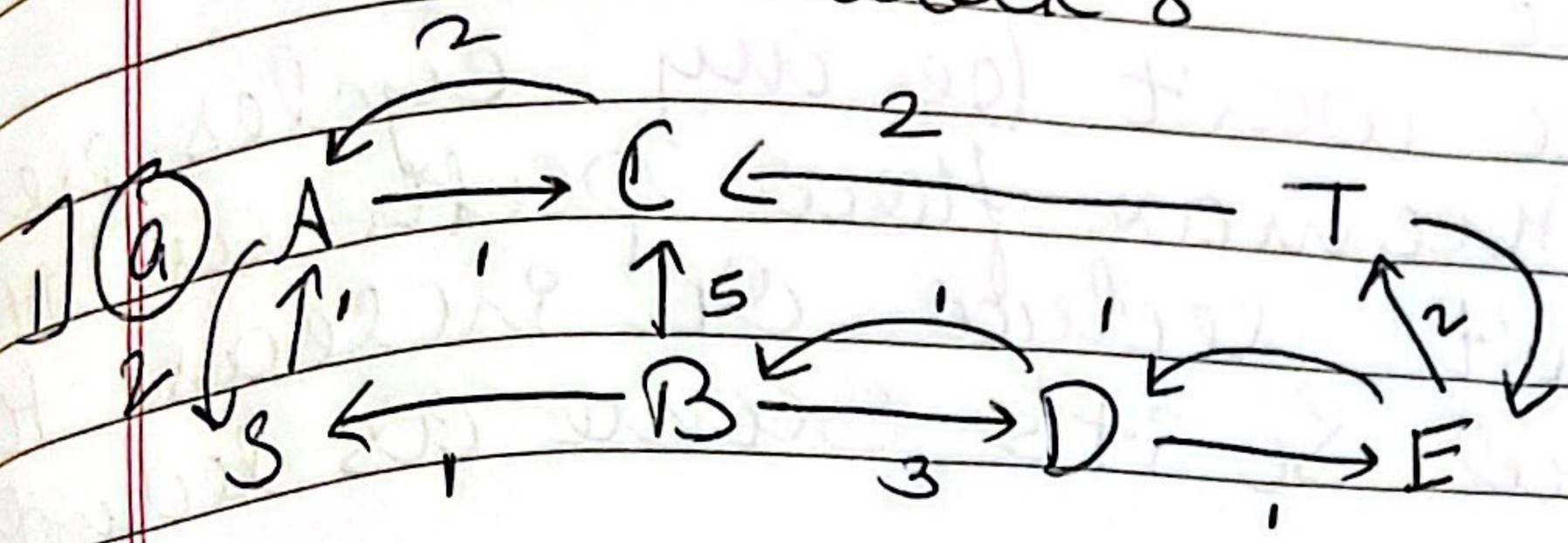
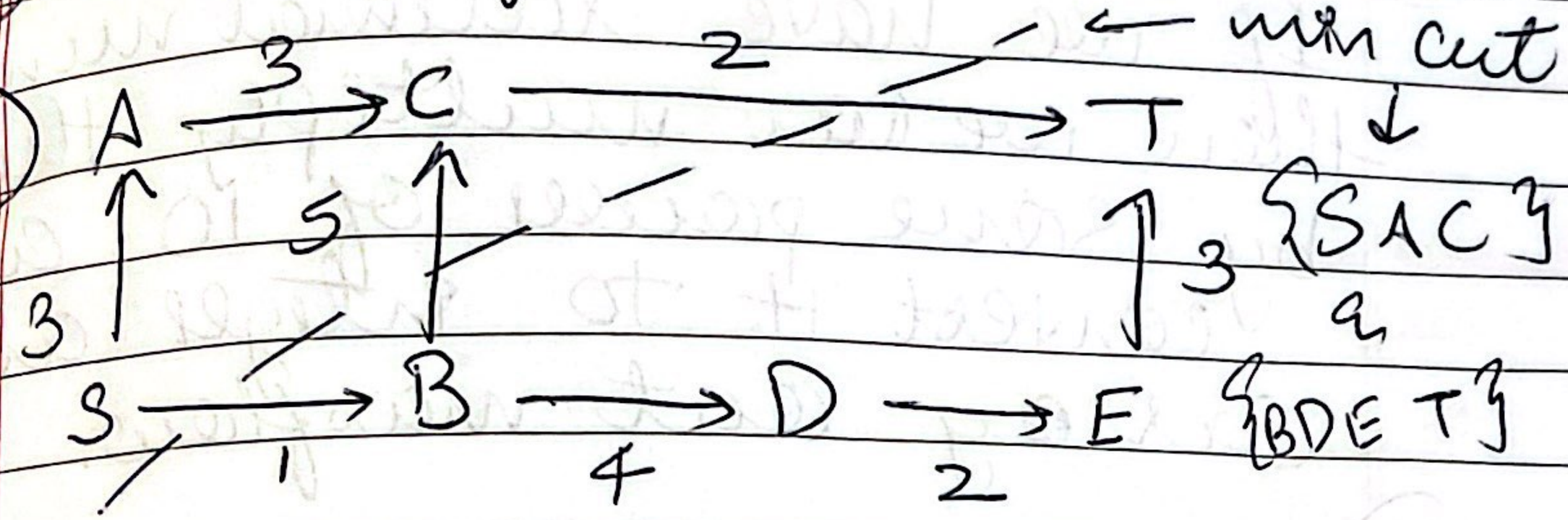
Homework 8

1] (a)



(b) The max flow value is 3

(c)



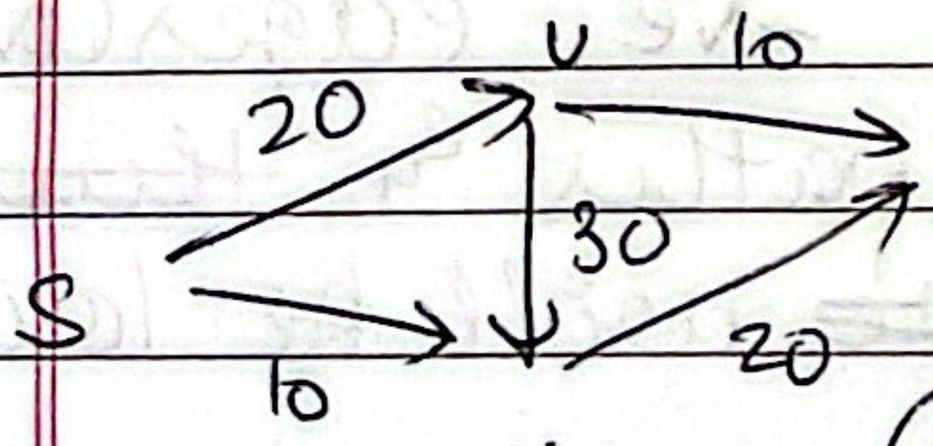
2] (a) False

This statement is not true as we can have -ve edges combining the disjoints paths & ~~then~~ then path S to T ~~be~~ will be lower.

(b) True
There won't be any cycles present in the max flow path as it won't reduce or increase the flow. So it's same as redundant.

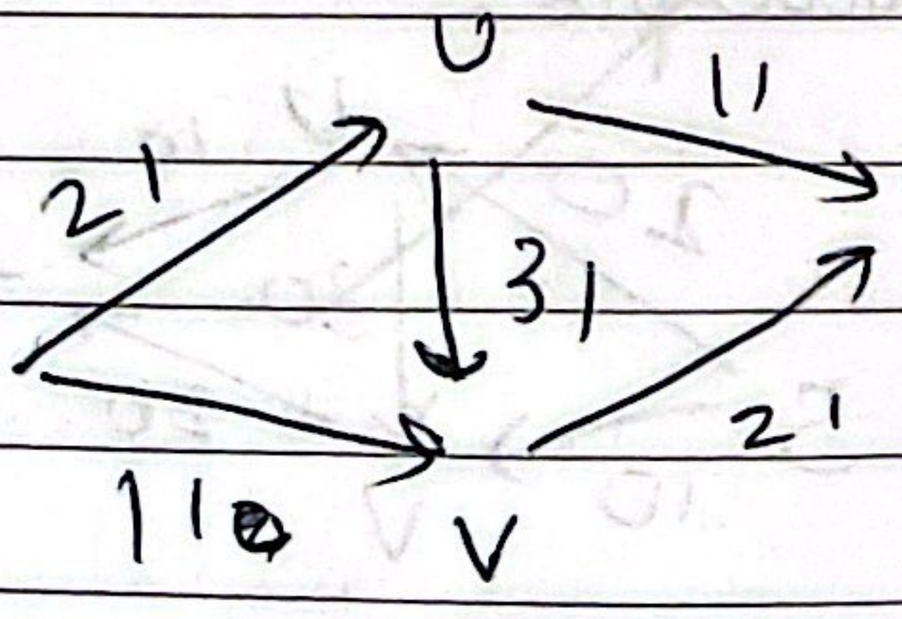
(c) False
If we have rational numbers then we can multiply them by some power of 10 and convert it to integer and carry out max flow.

(d) False
Consider the example:



Right now the maximum flow value is 30

Now when we add 1 to ~~edge~~ ~~the~~ each edge it becomes



Now the max flow value is 33, which is ~~31~~ 1+1<

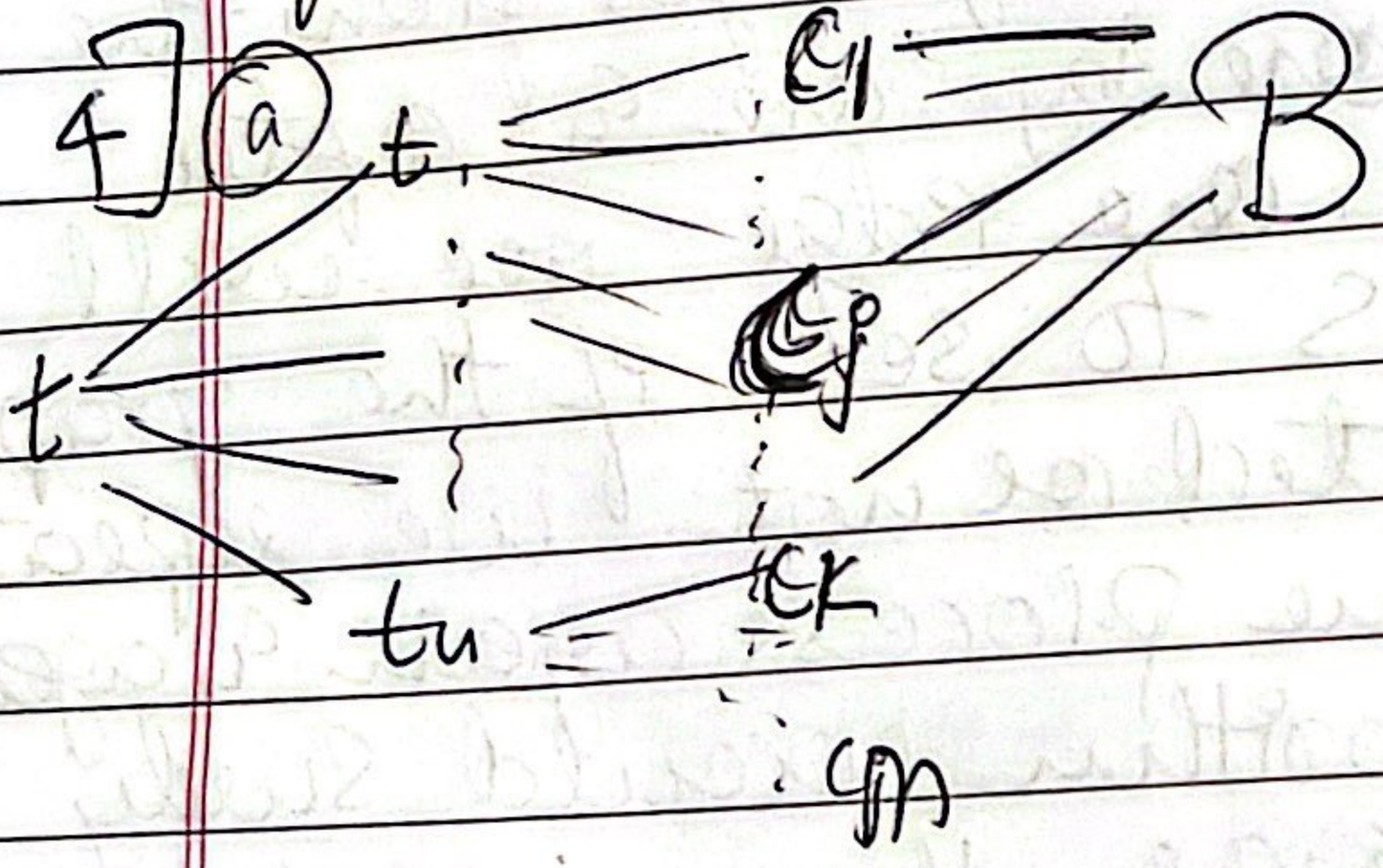
(e) True
~~It same as~~ As the max flow value would also be k times the original value, the min cut remains unaffected.

3] We will be using any of the max flow algorithm which gives us polynomial time. We will ~~be making changes in~~ then start removing edges from the source one by one & after removing one edge we will run BFS to see if the graph is connected or not. We repeat the same process again & again. The algorithm would surely terminate as there might be a case where before removing all the k edges the ~~algo~~ Connectivity of the graph is not maintained or ~~after~~ ~~then~~ the algorithm ~~leave~~ successfully removes all k edges.
The Conservation of the algorithm is also maintained as the flow going out of s is going to be equal to the flow going

a to t.

Follow up question :-

If the edges have more than unit capacity then we will remove the maximum capacity edge from the graph instead of removing edges from the source. The rest of the process remains the same.



The collection of tourists as the source & the Bank as the sink node.

$t_1 \ldots t_n$ is the nodes that represent the n tourists.

$C_1 \ldots C_m$ represent the m currencies the edge capacities b/w t & $t_1 \ldots t_n$ are the no. of dollars n tourists are willing to convert.

The edge capacities b/w $t_1 \ldots t_n$ &

C1... Cm are the no. of dollars the tourist want to convert in a particular currency.
The edge capacities from C1... Cm to B are the dollars the bank can convert to a particular currency.
We will be using any max flow algo & initially there won't be any conversion so f(e) = 0 for all e in G.
In the augment step we will be only having the forward flow & no backward flow as the conversion once done can't be undone & we are interested in the max no. of conversions/requests & not the max. value of conversion of dollars.

(b) The algorithm terminates once the B's capacity of Bank is saturated. & the max no. of conversions are outputted. The no. of dollars given by the tourists for conversion equals to the no. of dollars converted

to a different currency.

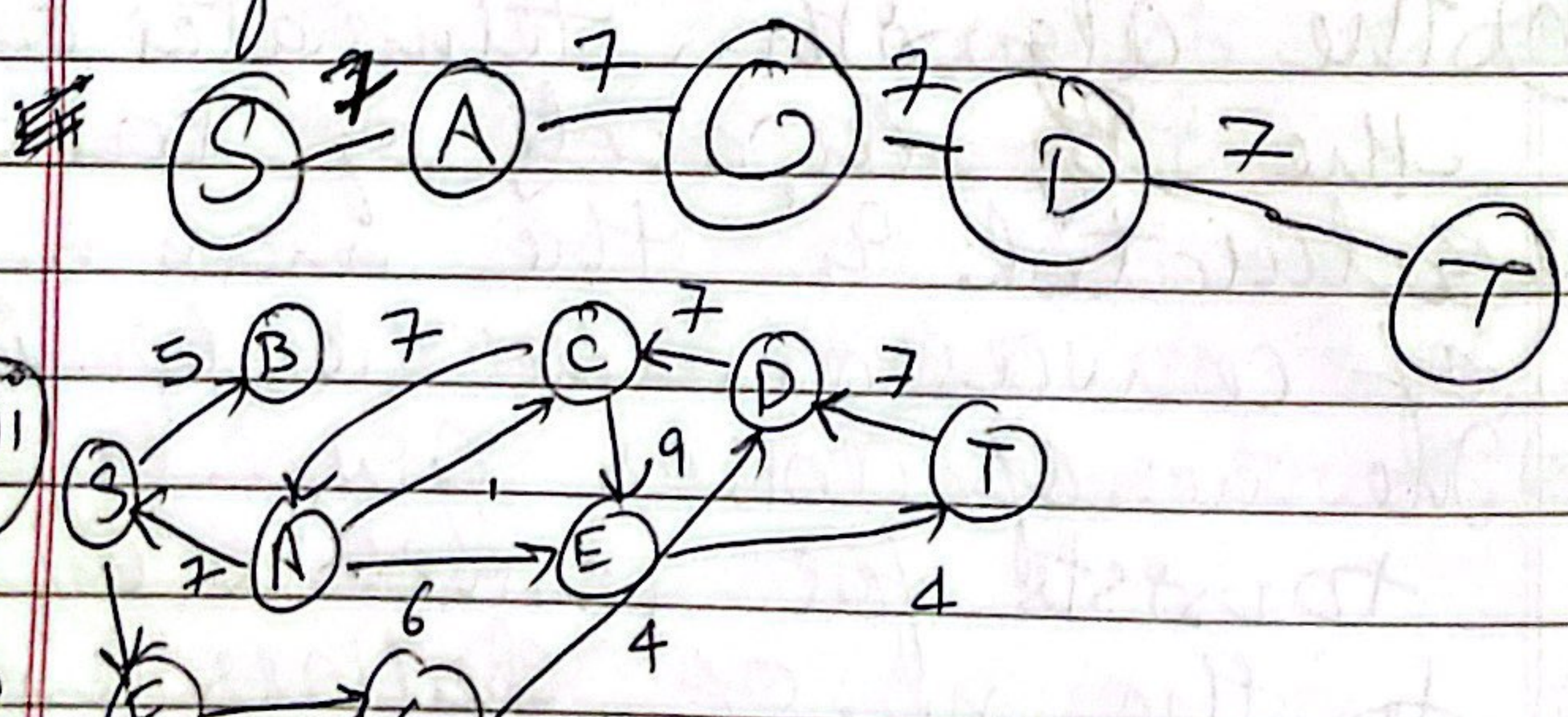
5] (a) We will be using any of the max flow algorithm. But in the augment step, we will be removing all the edges visited by any one of the tours after the 1st tour instead of having forward & backward flow.

(b) Over here, the process is same as above but will be removing all edges & all nodes visited by any of the tours.

6] (a) i] $\Delta = 4$

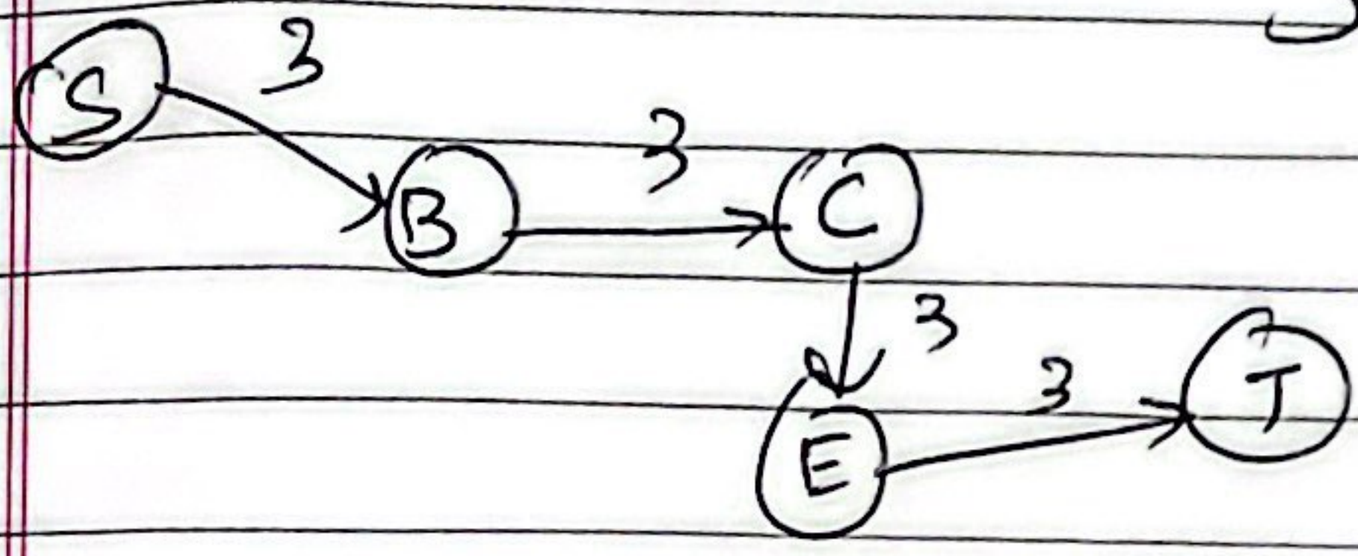
augmented path is
S - A - C - D - T

ii) flow = 7



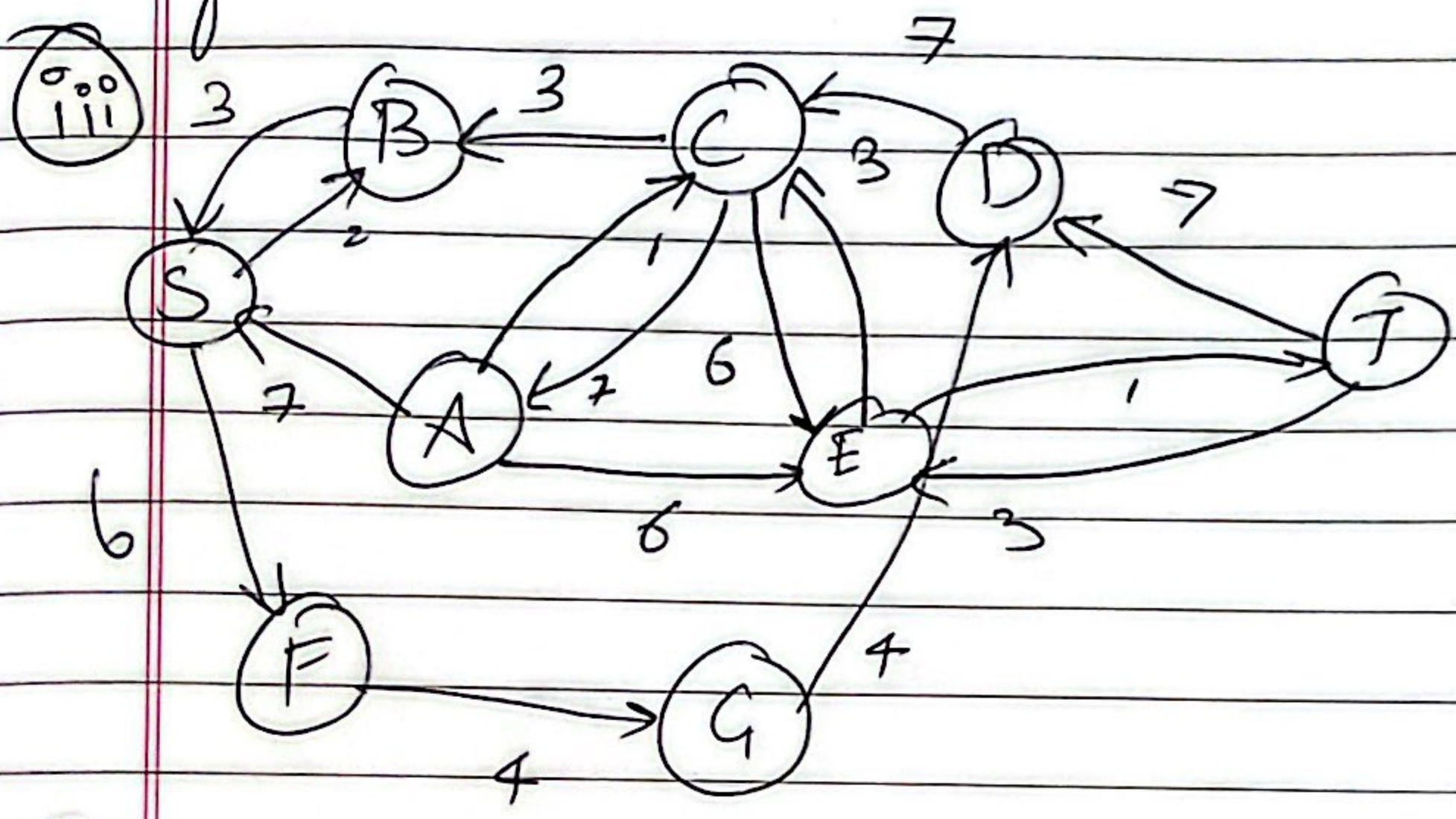
(b) i) $\Delta = 2$

S - B - C - E - T



ii) flow = 3

iii)



(c) Using the scaled f version helps used to reduce the no. of iterations. If we used the ~~fulk~~ Ford-Fulkerson algorithm then we would be first selecting S - B or then S - A or so on. In a way we would end up doing a lot of undoing or then finally to our final answer. This is avoided

a scaled version of the same.