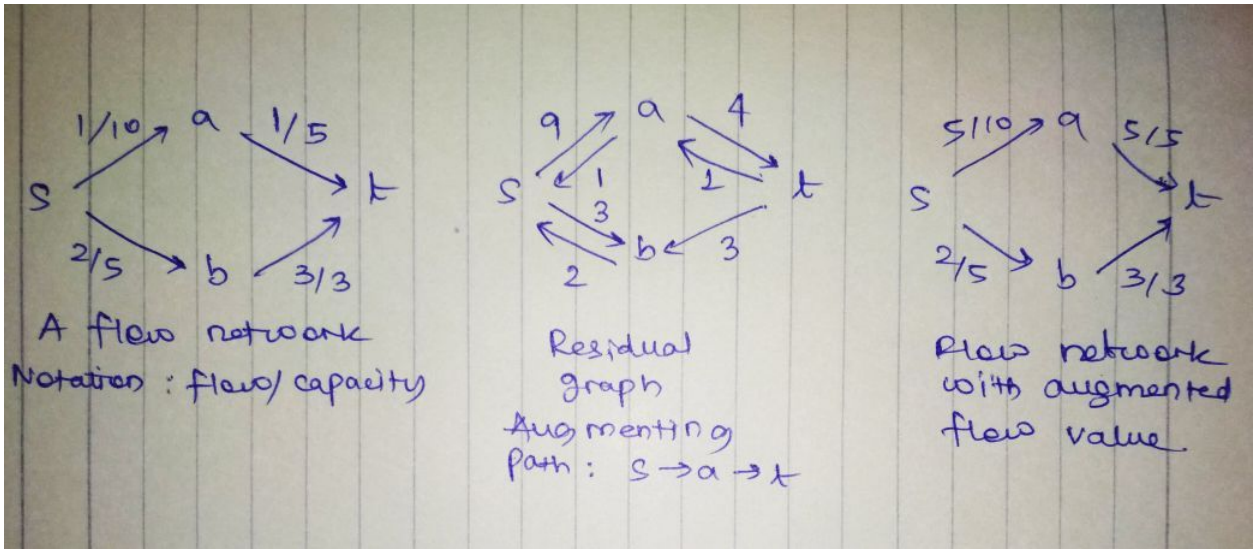
Solutions and rubric for the mid-semester exam

Q1

Ans

In the context of flow networks, augmenting paths are used to augment the flow value in a flow network, and thereby find the maximum flow value. This is done by constructing the residual graph corresponding to a flow. Any path from s to t in the residual graph can be used to augment the flow value. When no such augmenting paths exist, it implies we have found the maximum flow.

Example:



Rubric:

Total marks= 4

Correctness of logic= 1 mark, Marking: Binary

Correctness of example= 3 marks, Marking: subjective

Breakdown for example:

Fully correct example: 3 marks

Partially correct example 1 or 2 marks (depends on clarity of answer)

Incorrect example, or example that doesn't contribute anything : 0

A correct example should have both the original network with capacities and flow values, and the residual network (with back edges for added flow).

Q2

Ans

a) TRUE

As $f(n) = \Theta(g(n)$

$\therefore \exists$ p,q such that $(p * g(n) \leq f(n) \leq q^* g(n))$

Also as g(n) = Θ(h(n)
∴ ∃ r,s such that (r * h(n) ≤ g(n) ≤ s* h(n))

∴ (s/p * h(n)) ≤ f(n) ≤ (r/q * h(n))
∴ (q/r * f(n)) ≤ h(n) ≤ (p/r * f(n))
Thus h(n) = θ(f(n))

b) TRUE

As f(n) = O(g(n)) and g(n) = O(h(n)), so
h(n) = Ω(f(n))
and it is the same as f(n) = O(h(n))

c) FALSE

Give counter example and show. For eg : take f(n) = n and g(n) = n + 1 .
Now f(n) = O(g(n)) and g(n) = O(f(n)) but
f(n) ≠ g(n)

Rubric :

3 Marks for each sub part.
1 mark for TRUE/FALSE
2 for explanation
Total = 3 * (1+2) = 3*3 = 9

Q3

Ans :

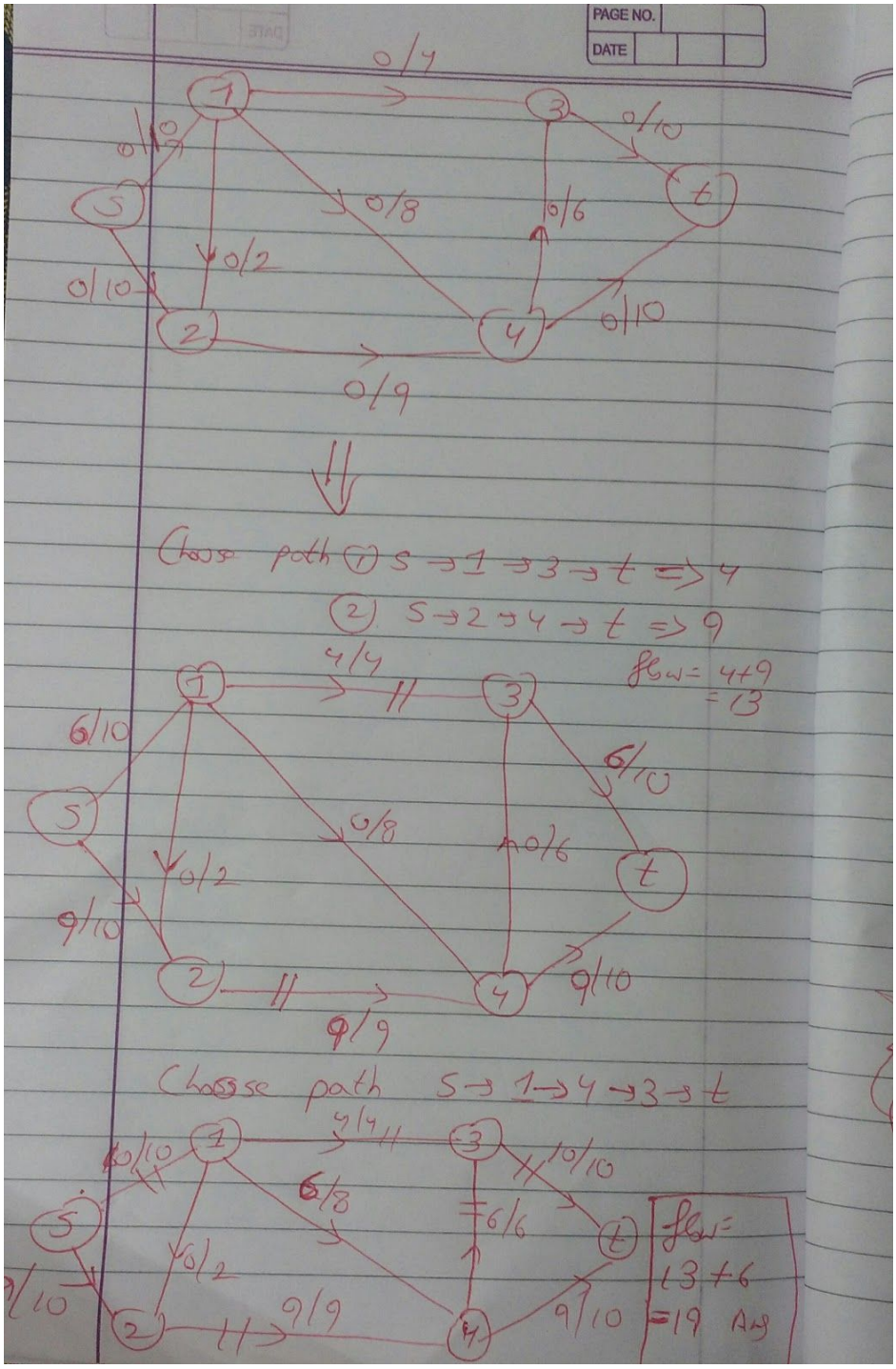
Log2n = Log10(n) / Log10(2) (where 10 is in the base)
Dividing or multiplying a constant factor Log10(2) does not affect the complexity. Thus O(Log2n) is equal to O(Log10n).

Rubric :

Total Marks = 3
Correct calculation = 2 marks, Marking Binary
Correct explanation = 1 mark, Marking Binary

A correct solution contains the complete calculation and also a little explanation reasoning the equality of complexity based on the calculation.

Q4 ans



Rubric Correct ans(2 marks)
steps(4 marks)

Q5

Ans

Using Breadth First Search (BFS).
1. Assign RED color to the source vertex (putting into set U).
2. Color all the neighbors with BLUE color (putting into set V).
3. Color all neighbor's neighbor with RED color (putting into set U).
4. This way, assign color to all vertices such that it satisfies all the constraints of m way coloring problem where m = 2.
5. While assigning colors, if we find a neighbor which is colored with same color as current vertex, then the graph cannot be colored with 2 vertices (or graph is not Bipartite)

If graph is represented using adjacency list, then the complexity becomes O(V+E).

Rubric : If explanation not clear -> -1
If complexity > O(V+E) -> give out of 3
If silly errors in code -> -0.5/1
Give marks for complexity only if it's the same as the written algo.

Q6

Take the two points with minimum and maximum x coordinates (say A and B), add them to hull, and join them so that the point space is divided into two halves S1 and S2. For both S1 and S2: take point C which is at max perpendicular distance from AB, add C to hull, and form triangle ACB. The points within ACB can't be part of the hull. Take the set of points to left of AC and right of BC and recursively repeat the process. The base case is when there is no such point C i.e no points left.

The above has to be written in pseudocode, the rubric will be as follows.

Rubric:

5 points if all steps of the algorithm are mentioned.

If base case not there, deduct 1
If not added points to hull / missed recursion step / getting directions wrong (especially when denoting set of points outside of triangle ACB) / not returning anything from functions, deduct 0.5.

Q7
Ans

Kahn's algorithm for Topological Sorting

https://www.geeksforgeeks.org/topological-sorting-indegree-based-solution/
DFS's 's algorithm for Topological Sorting
https://www.geeksforgeeks.org/topological-sorting/
Topological ordering of the Given graph:
Can be many solution, one out of them: 0 1 2 3 4 5

Rubric:

if steps of one of these two algorithm partially right [2]
If right topological order without right algorithm steps [3]
if steps of one of these two algorithm correct [4]
If partially right steps with right topological order [5]
If right steps with right topological order [8]