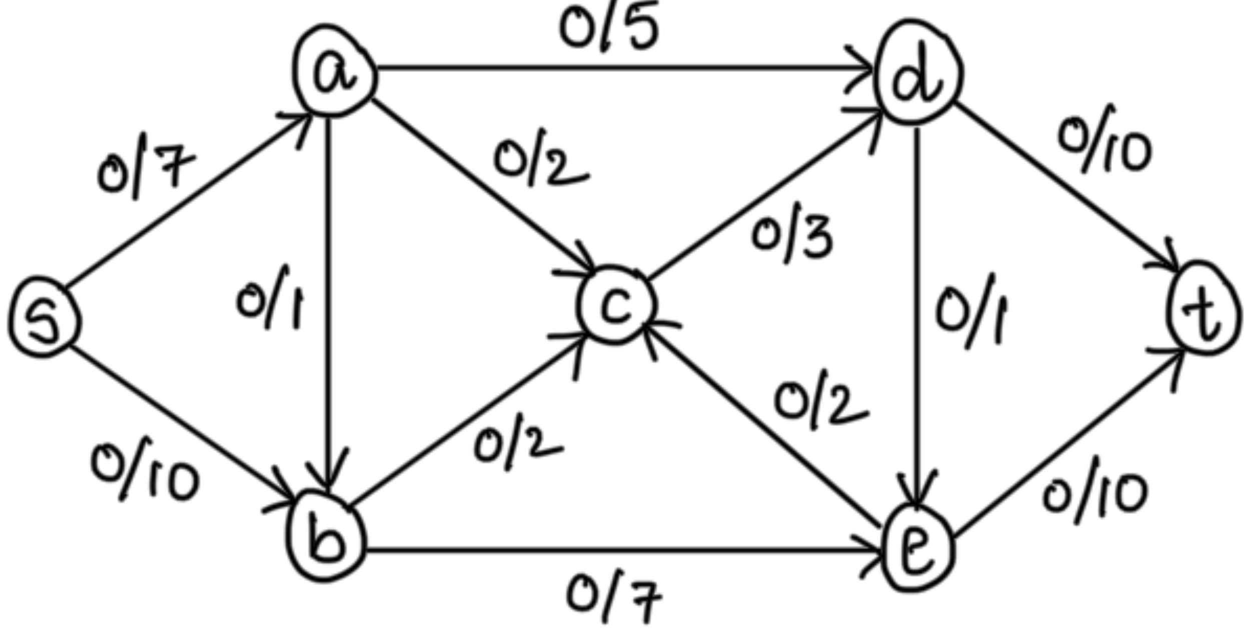
Q1.

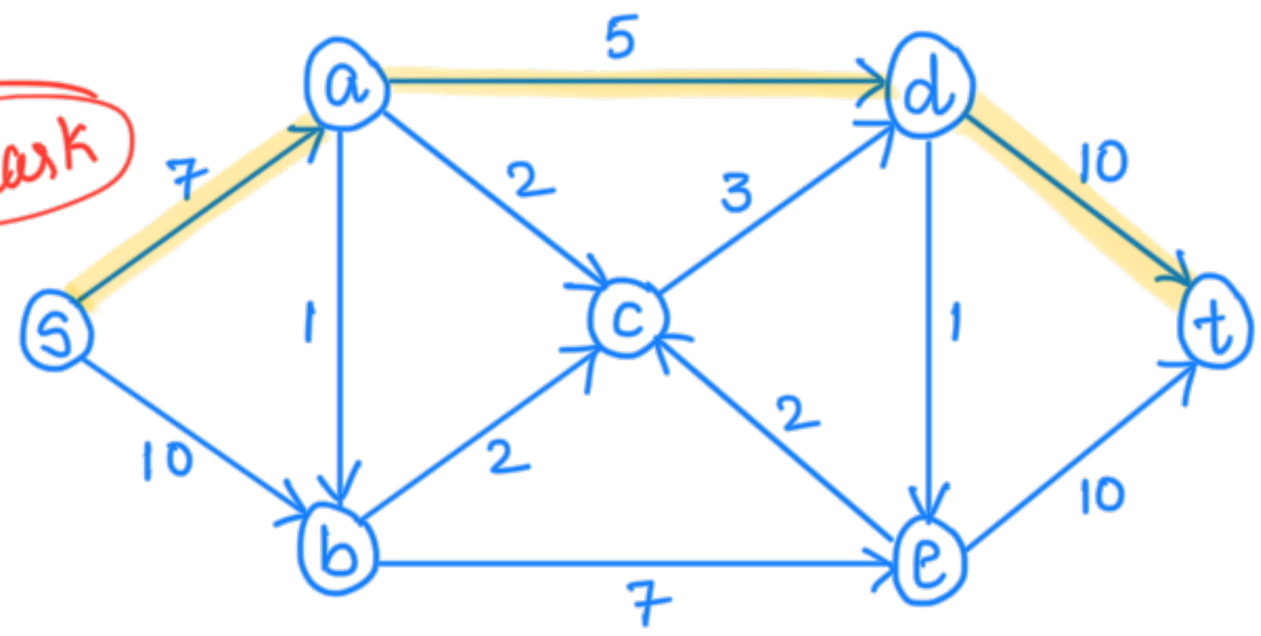
INPUT GRAPH

flow = 0

RESIDUAL GRAPH



1/2 mark

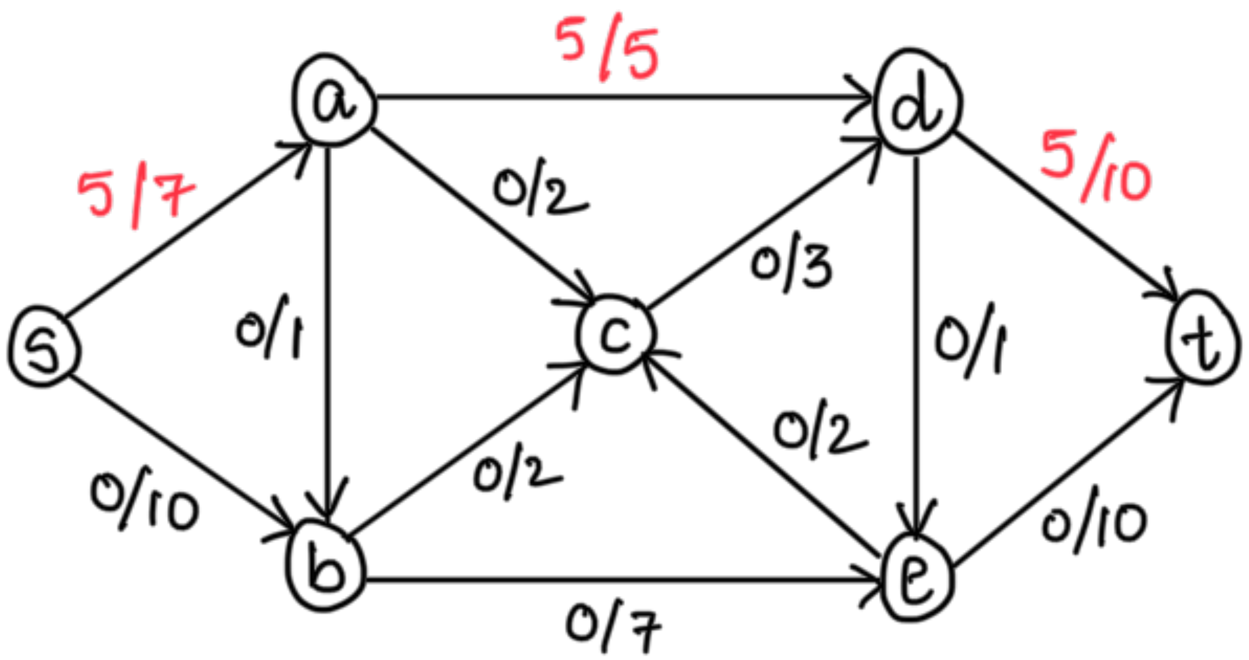


1/2 mark

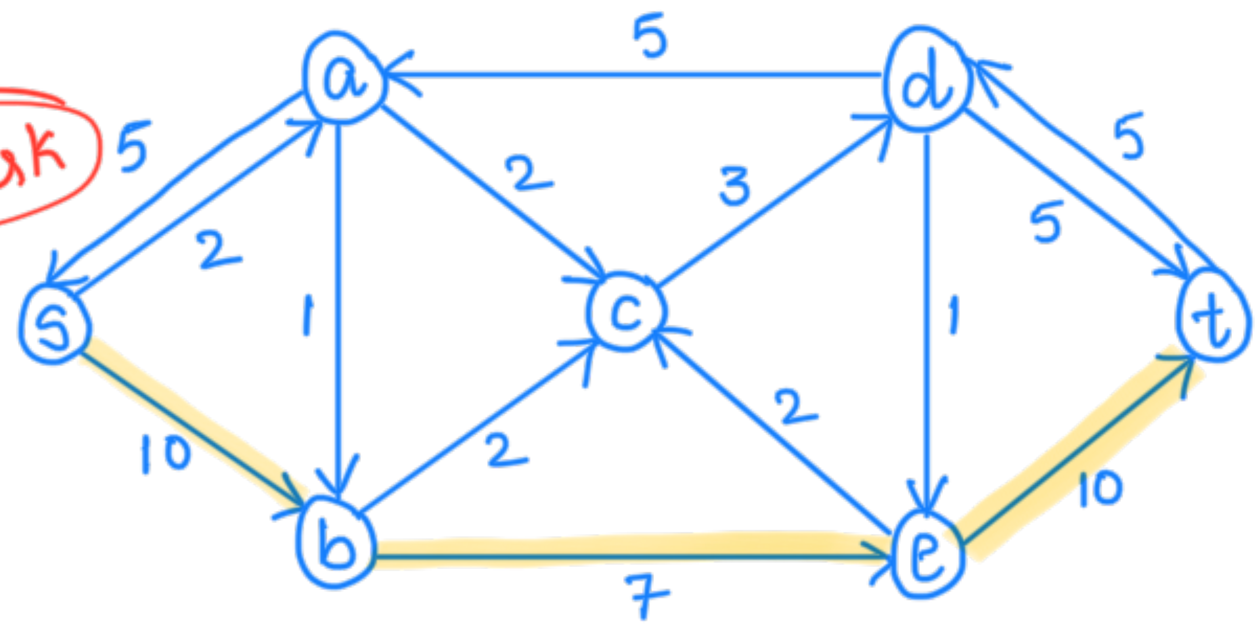
augmenting path: $s \to a \to d \to t$

residual capacity: $\min(7, 5, 10) = 5$

flow $= 0 + 5 = 5$



1/2 mark

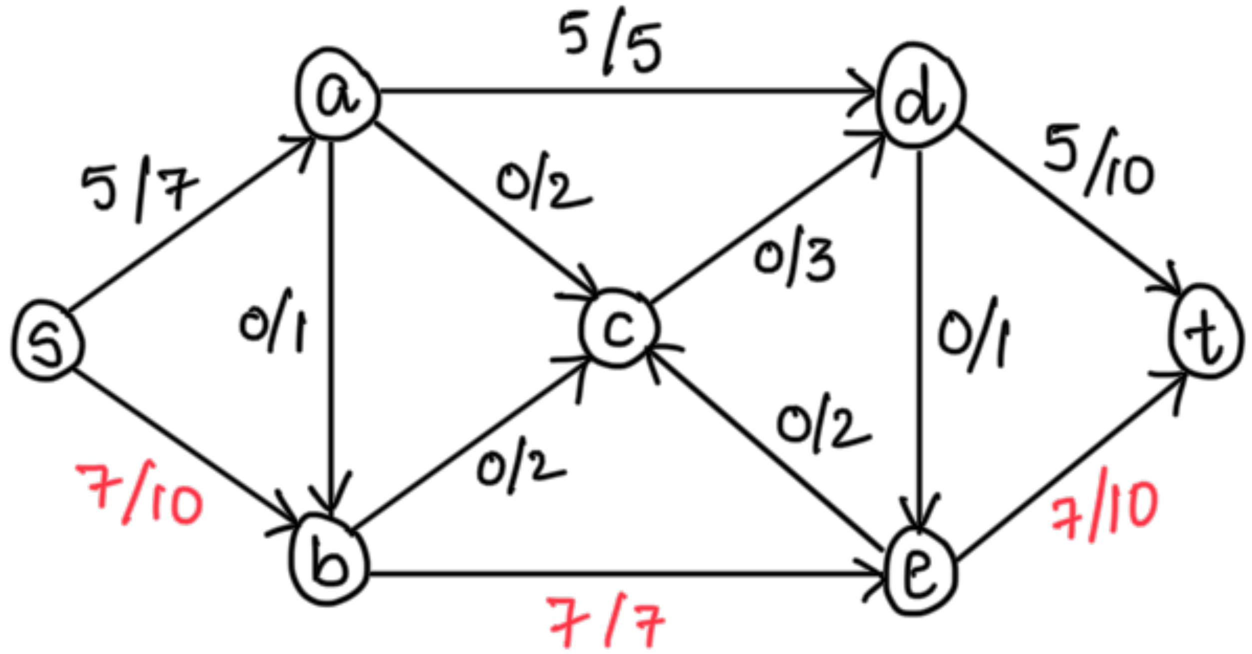


1/2 mark

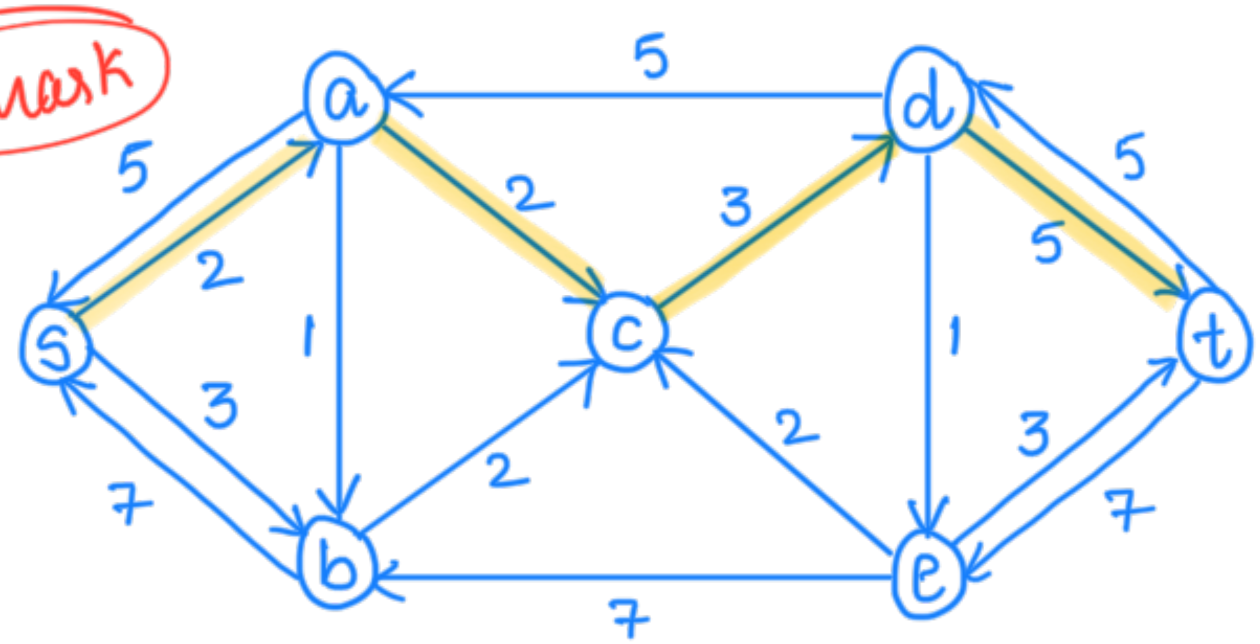
augmenting path: $s \to b \to e \to t$

residual capacity: $\min(10, 7, 10) = 7$

flow $= 5 + 7 = 12$



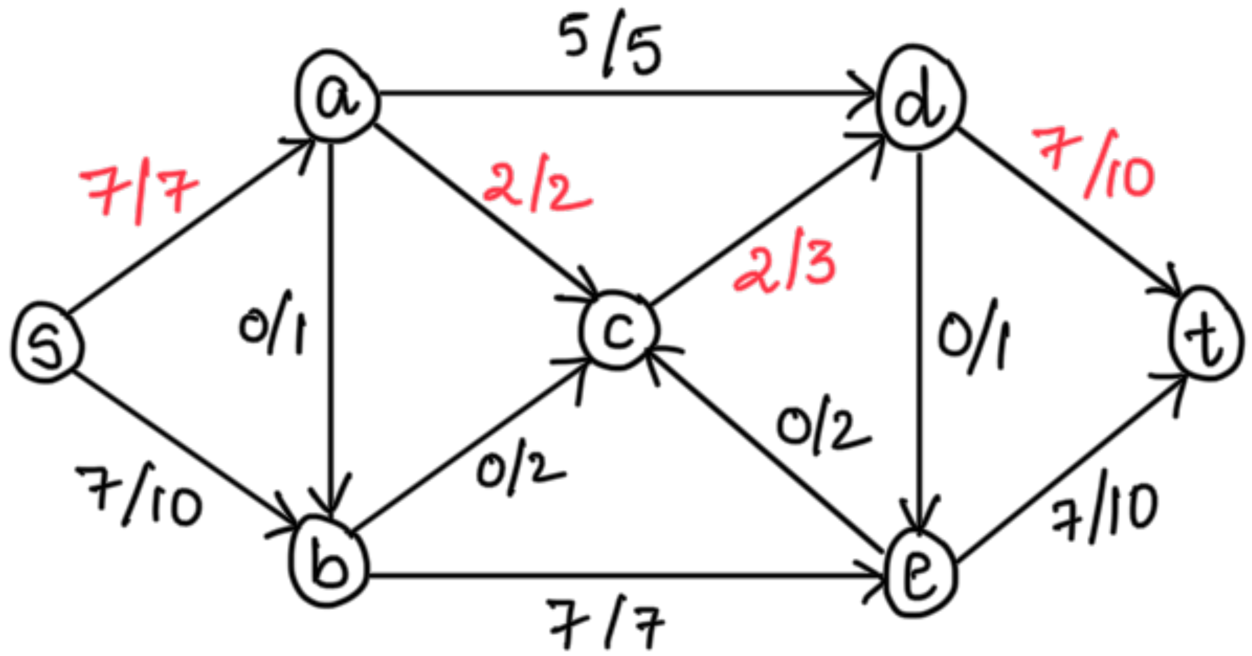
1/2 mark



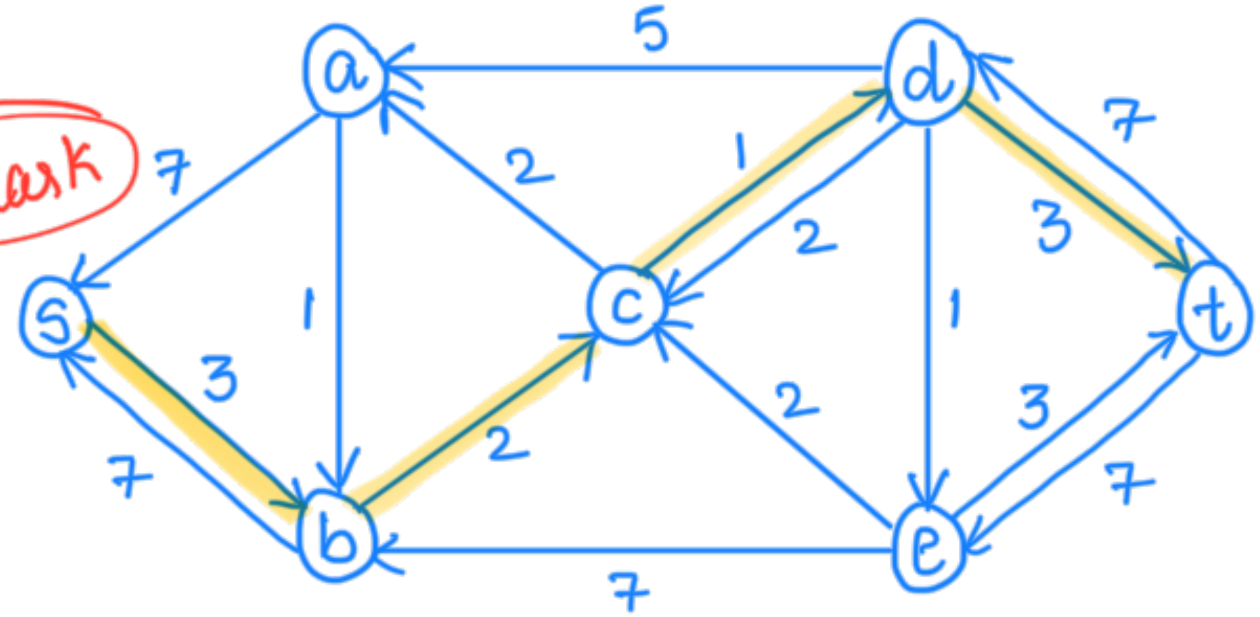
augmenting path: s → a → c → d → t

1/2 mark residual capacity: min(2, 2, 3, 5) = 2

flow = 12 + 2 = 14



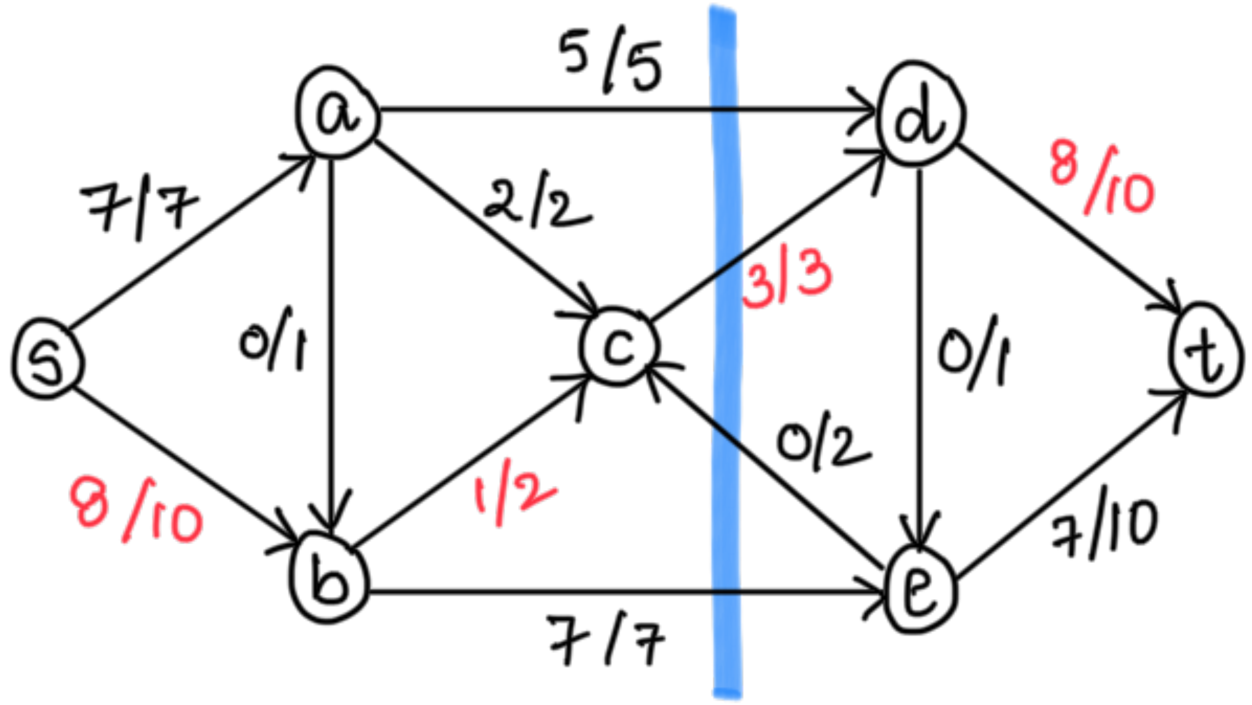
1/2 mark



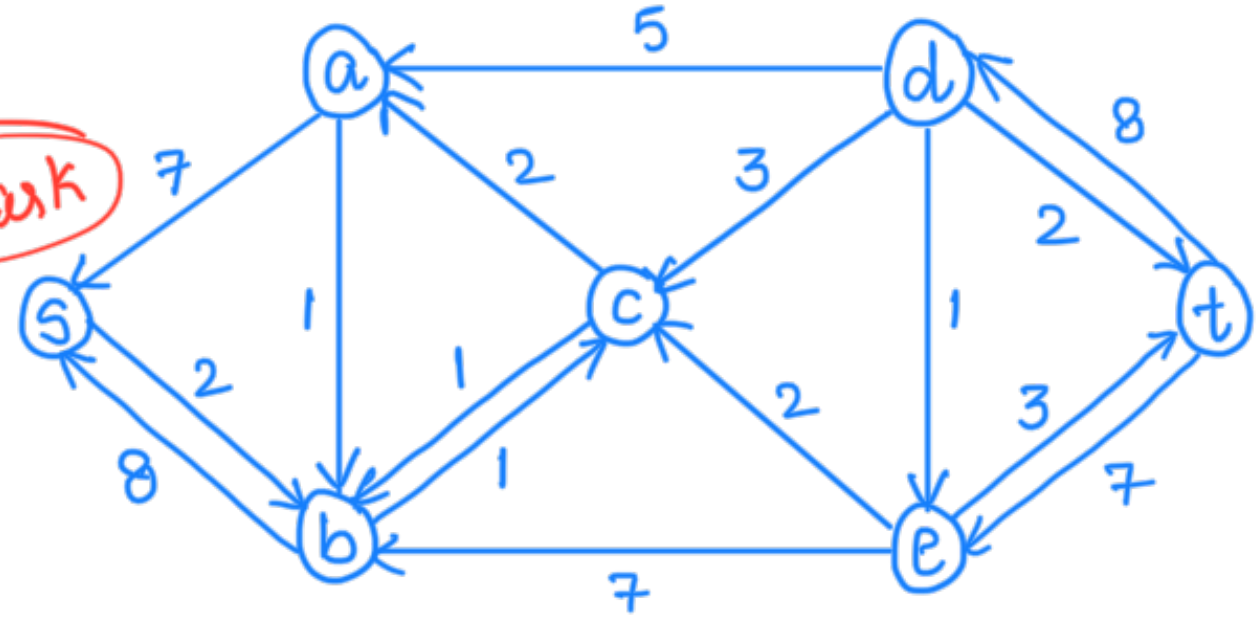
augmenting path: s → b → c → d → t

1/2 mark residual capacity: min(3, 2, 1, 3) = 1

flow = 14 + 1 = 15



1/2 mark



No augmenting path in the residual network.

Min-cut:

cut (S,T) = {(a,d), (c,d), (e,c), (b,e)}

1 mark

S = {s, a, b, c} T = {d, e, t}

1/2 mark Maximum flow = 15

1 mark If each edge capacity is increased by a value of 1, then the changed maximum flow is 18. The flow will be increased by a value of 1 in each of the augmenting paths except the last one.

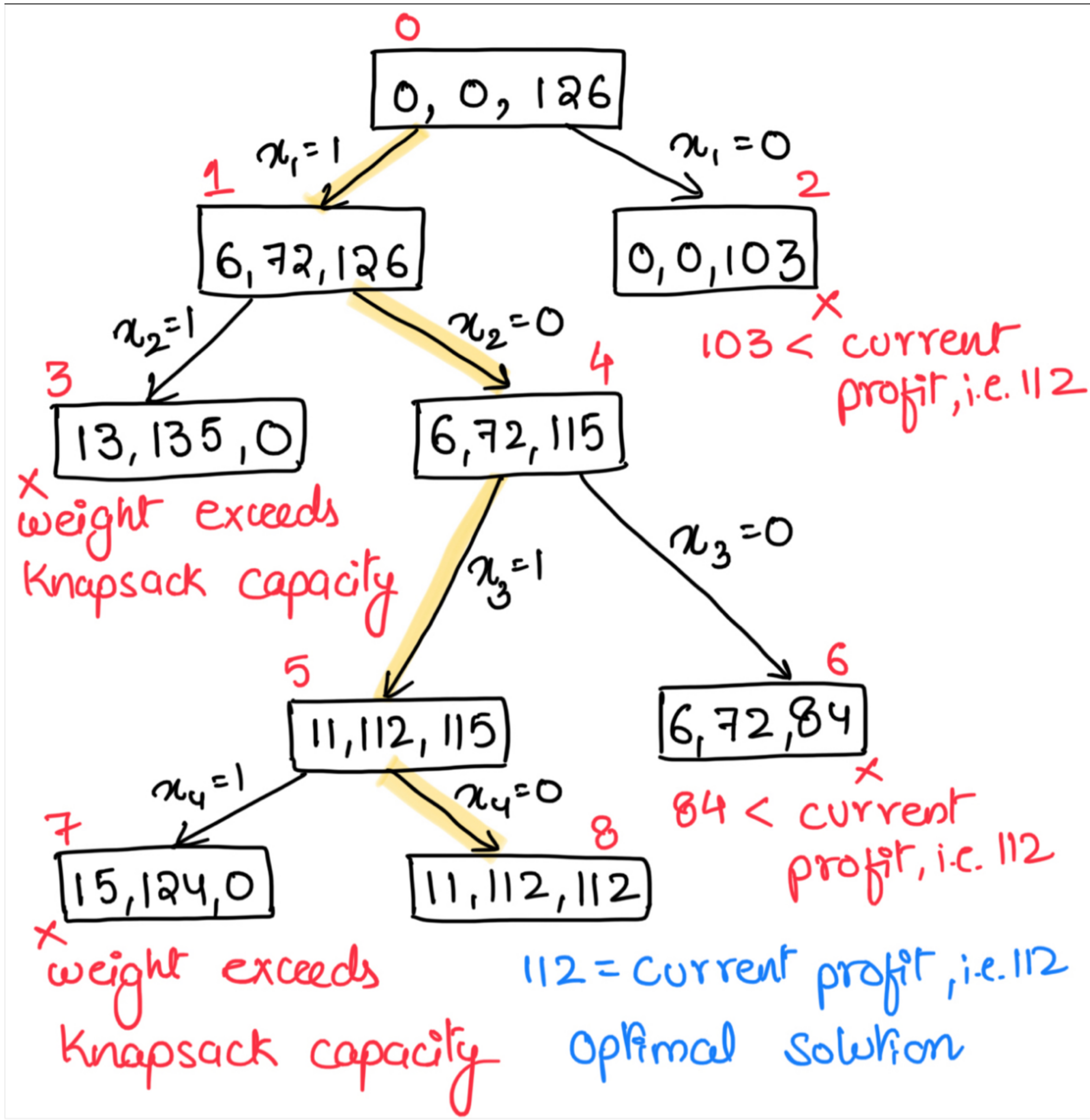
Q2.

(a) 3/2-approximation algorithm utilizes the concept of minimum-cost perfect matching to get an Eulerian graph.

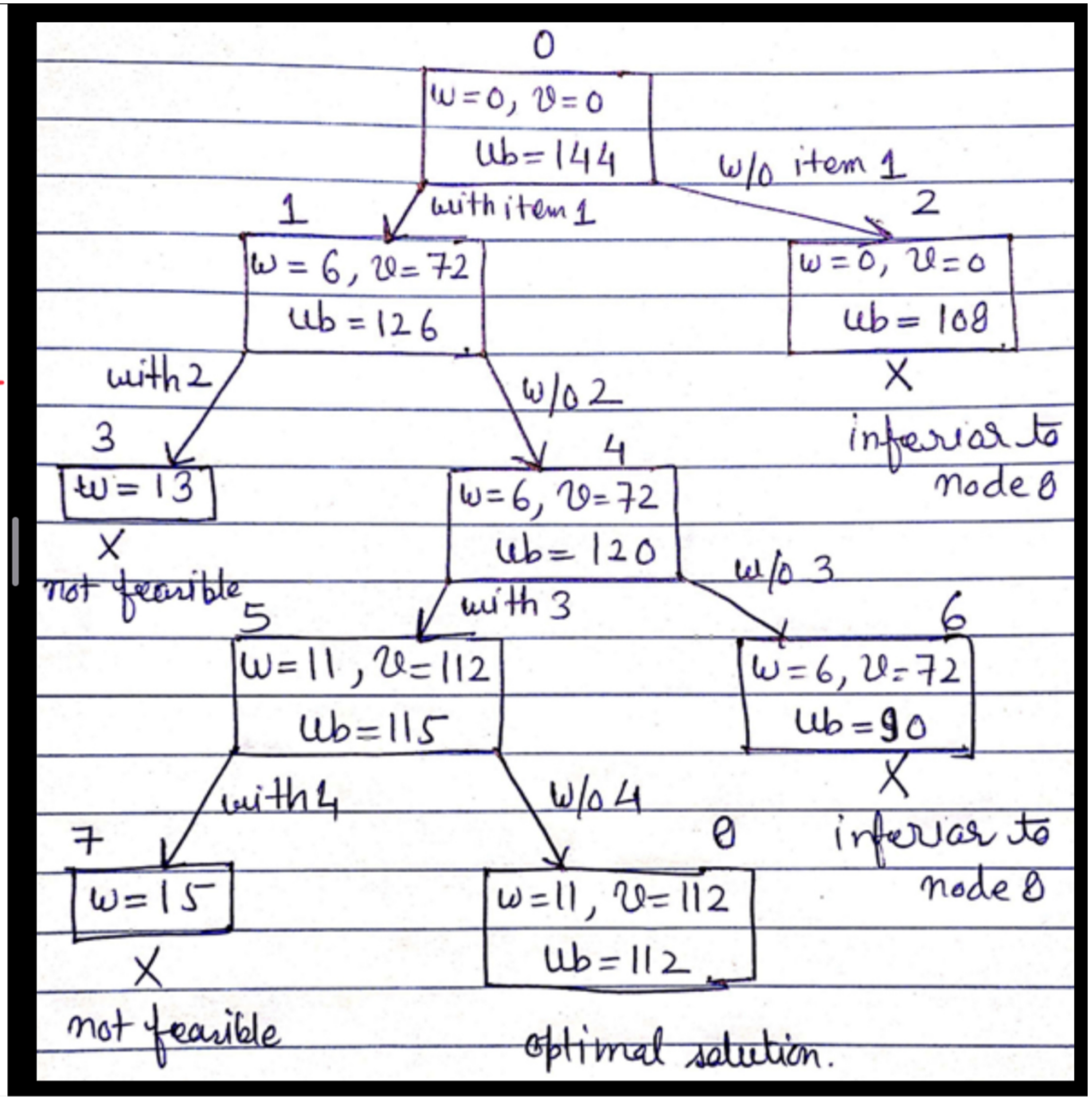
1 mark

2-approximation algorithm utilizes the concept of pre-order traversal in a tree.

(b)



0
0, 0, 126
x1=1
x1=0
1
2
6, 72, 126
0, 0, 103
103 < current profit, i.e. 112
x2=1
x2=0
3
4
13, 135, 0
6, 72, 115
weight exceeds Knapsack capacity
x3=1
x3=0
5
6
11, 112, 115
6, 72, 84
84 < current profit, i.e. 112
x4=1
x4=0
7
8
15, 124, 0
11, 112, 112
weight exceeds Knapsack capacity
112 = current profit, i.e. 112
optimal solution



0
w=0, v=0
ub=144
with item 1
w/o item 1
1
2
w = 6, v= 72
ub = 126
w=0, v=0
ub= 108
inferior to node 8
with 2
w/o 2
3
4
w= 13
not feasible
w=6, v=72
ub= 120
with 3
w/o 3
5
6
w=11, v=112
ub=115
w=6, v=72
ub=90
inferior to node 8
with 4
w/o 4
7
8
w=15
not feasible
w=11, v=112
ub=112
optimal solution.

Marking Scheme Q2(b)

5-6 marks - State space tree is perfect with clear depiction of nodes correct sequence and reason for un-explored nodes.

3-4 marks - State space tree is correct.

0-2 marks - State space tree is partially correct atleast 50%

Q3:

	0	1	2	3
p_i		4/17	1/17	4/17
q_i	3/17	1/17	1/17	3/17

$$\text{root}[\,][\,]: \quad \begin{array}{cc} & \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} 1 & 1 & 1 \text{ or } 3 \\ \times & 2 & 3 \\ \times & \times & 3 \end{bmatrix} \end{array}$$

$$c[\,][\,]: \quad \begin{array}{cc} & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{bmatrix} 3/17 & 12/17 & 18/17 & 38/17 \\ \times & 1/17 & 5/17 & 18/17 \\ \times & \times & 1/17 & 12/17 \\ \times & \times & \times & 3/17 \end{bmatrix} \end{array}$$

$$w[\,][\,]: \quad \begin{array}{cc} & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{bmatrix} 3/17 & 8/17 & 10/17 & 17/17 \\ \times & 1/17 & 3/17 & 10/17 \\ \times & \times & 1/17 & 8/17 \\ \times & \times & \times & 3/17 \end{bmatrix} \end{array}$$

$l = 1$:

$$e[1][1] = e[1][0] + e[2][1] + w[1][1]$$
$r=1$
$$= 3/17 + 1/17 + 8/17 = 12/17$$

$$e[2][2] = e[2][1] + e[3][2] + w[2][2]$$
$r=2$
$$= 1/17 + 1/17 + 3/17 = 5/17$$

$$e[3][3] = e[3][2] + e[4][3] + w[3][3]$$
$r=3$
$$= 1/17 + 3/17 + 8/17 = 12/17$$

$l = 2$:

$$e[1][2] = \min\begin{Bmatrix} e[1][0] + e[2][2] \\ e[1][1] + e[3][2] \end{Bmatrix} + w[1][2] = \min\begin{Bmatrix} 3/17 + 5/17 \\ 12/17 + 1/17 \end{Bmatrix} + 10/17$$
$r = 1, 2$

$$= \min(8/17, 13/17) + 10/17 = 18/17 \quad \text{for } r=1$$

$$e[2][3] = \min\begin{Bmatrix} e[2][1] + e[3][3] \\ e[2][2] + e[4][3] \end{Bmatrix} + w[2][3] = \min\begin{Bmatrix} 1/17 + 12/17 \\ 5/17 + 3/17 \end{Bmatrix} + 10/17$$
$r = 2, 3$

$$\min(13/17, 8/17) + 10/17 = 18/17 \quad \text{for } r=3$$

$l = 3$:

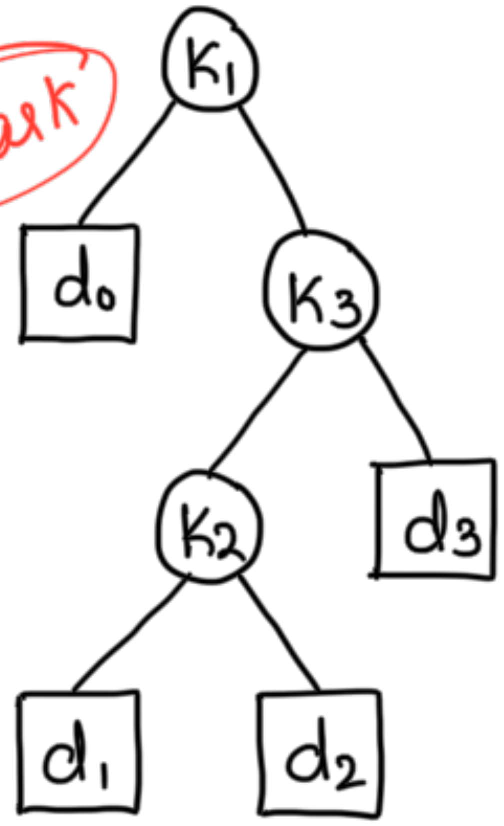
$$e[1][3] = \min \begin{cases} e[1][0] + e[2][3] \\ e[1][1] + e[3][3] \\ e[1][2] + e[4][3] \end{cases} + w[1][3] = \min \begin{cases} 3/17 + 18/17 \\ 12/17 + 12/17 \\ 18/17 + 3/17 \end{cases} + 17/17$$

$r = 1, 2, 3$

$$= \min(21/17,\ 24/17,\ 21/17) + 17/17 = 38/17 \text{ for } r = 1 \text{ or } 3$$

Structure 1:

1 mark

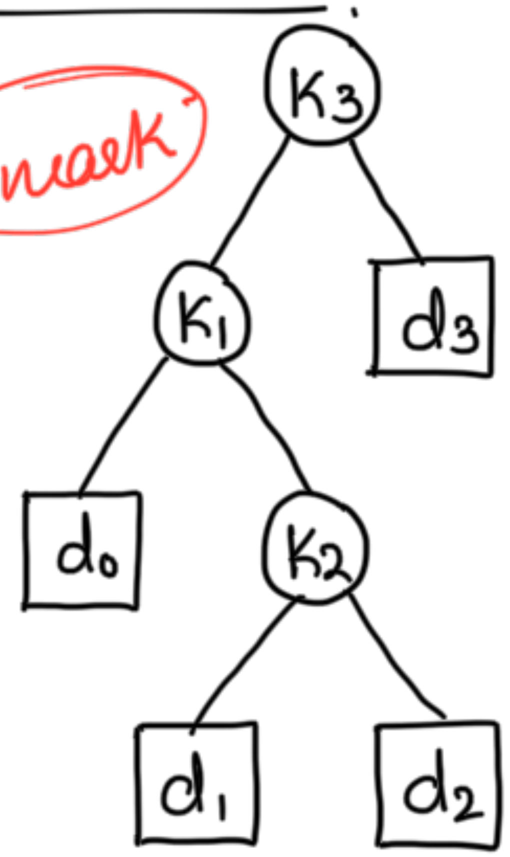


$K_1 = 4 \times 1 = 4$
$K_2 = 1 \times 3 = 3$
$K_3 = 4 \times 2 = 8$
$d_0 = 3 \times 2 = 6$
$d_1 = 1 \times 4 = 4$
$d_2 = 1 \times 4 = 4$
$d_3 = 3 \times 3 = 9$

$\underline{38}$

Structure 2:

1 mark



$K_1 = 4 \times 2 = 8$
$K_2 = 1 \times 3 = 3$
$K_3 = 4 \times 1 = 4$
$d_0 = 3 \times 3 = 9$
$d_1 = 1 \times 4 = 4$
$d_2 = 1 \times 4 = 4$
$d_3 = 3 \times 2 = 6$

$\underline{38}$

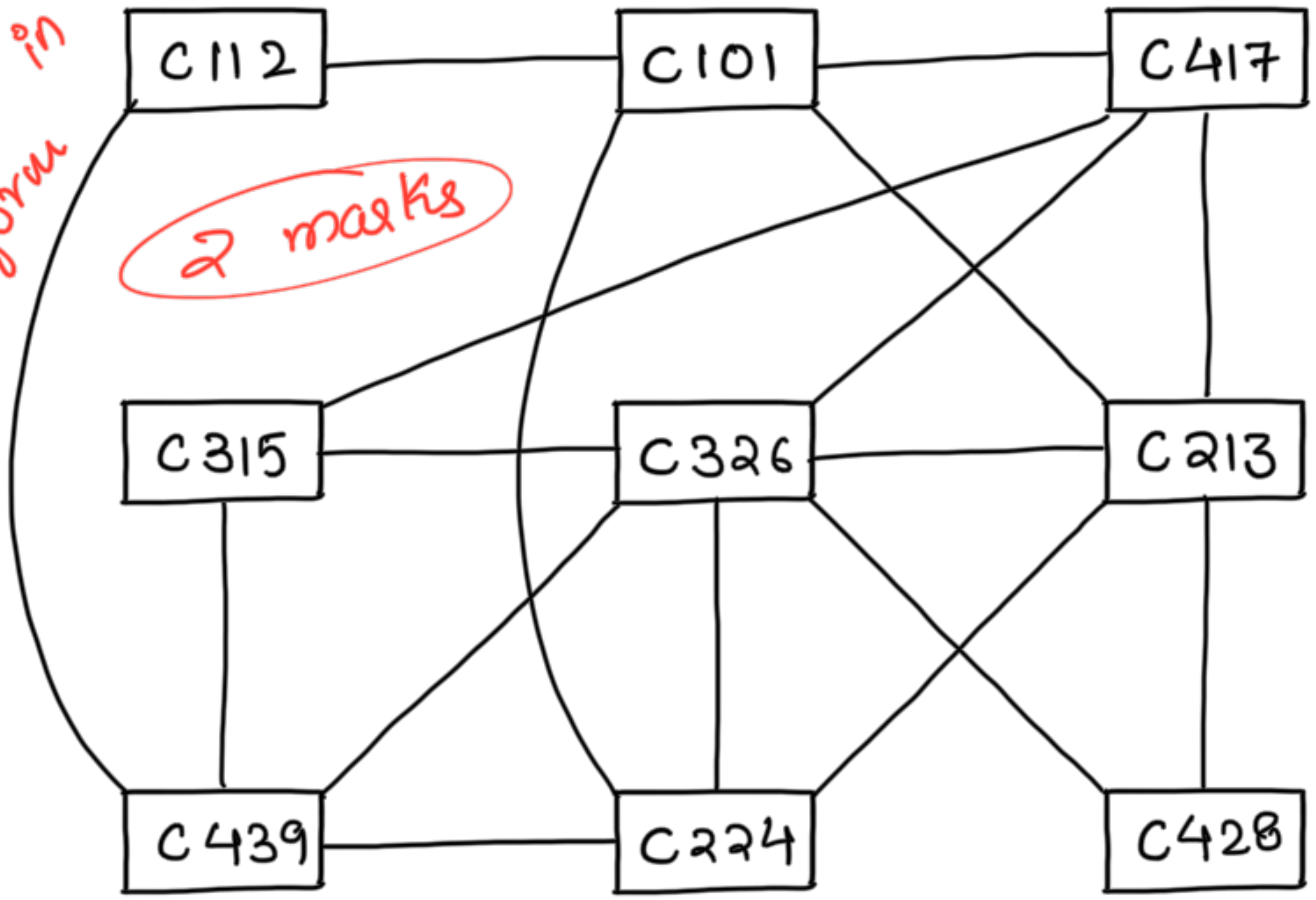
1 mark – Correct w[][] matrix.

1 mark – 1/2 mark for diagonal entries in each matrix, i.e. e[][] and root[][].

3 mark – 1 mark for each of the three correct entries in matrix e[][] and root[][].

Q4.

Graph in any form



This graph can be colored using four colors. Therefore, minimum number of exam slots is four.

1 mark

2 marks - For the algorithm (Back-tracking or Greedy) steps.

2 marks - For intermediate steps.

Q5.

Good suffix preprocessing case 1

```
void bmPreprocess1()
{
    int i=m, j=m+1;
    f[i]=j;
    while (i>0)
    {
1 mark  while (j<=m && p[i-1]!=p[j-1])
        {
  1/2 mark  if (s[j]==0) s[j]=j-i;
  1/2 mark  j=f[j];
        }
1/2 mark i--; j--;
1/2 mark f[i]=j;
    }
}
```

Good suffix preprocessing case 2

```
void bmPreprocess2()
{
    int i, j;
    j=f[0];
    for (i=0; i<=m; i++)
    {
1 mark  if (s[i]==0) s[i]=j;
1 mark  if (i==j) j=f[j];
    }
}
```

2 marks – Computed shift values for "CTTACTTAC"

	1	2	3	4	5	6	7	8	9	10
s[]:	4	4	4	4	4	8	8	8	8	1

For in-efficient solution with good suffix

3.5 marks – algorithm , in case of textual algorithm 2.5 marks

1.5 marks – example