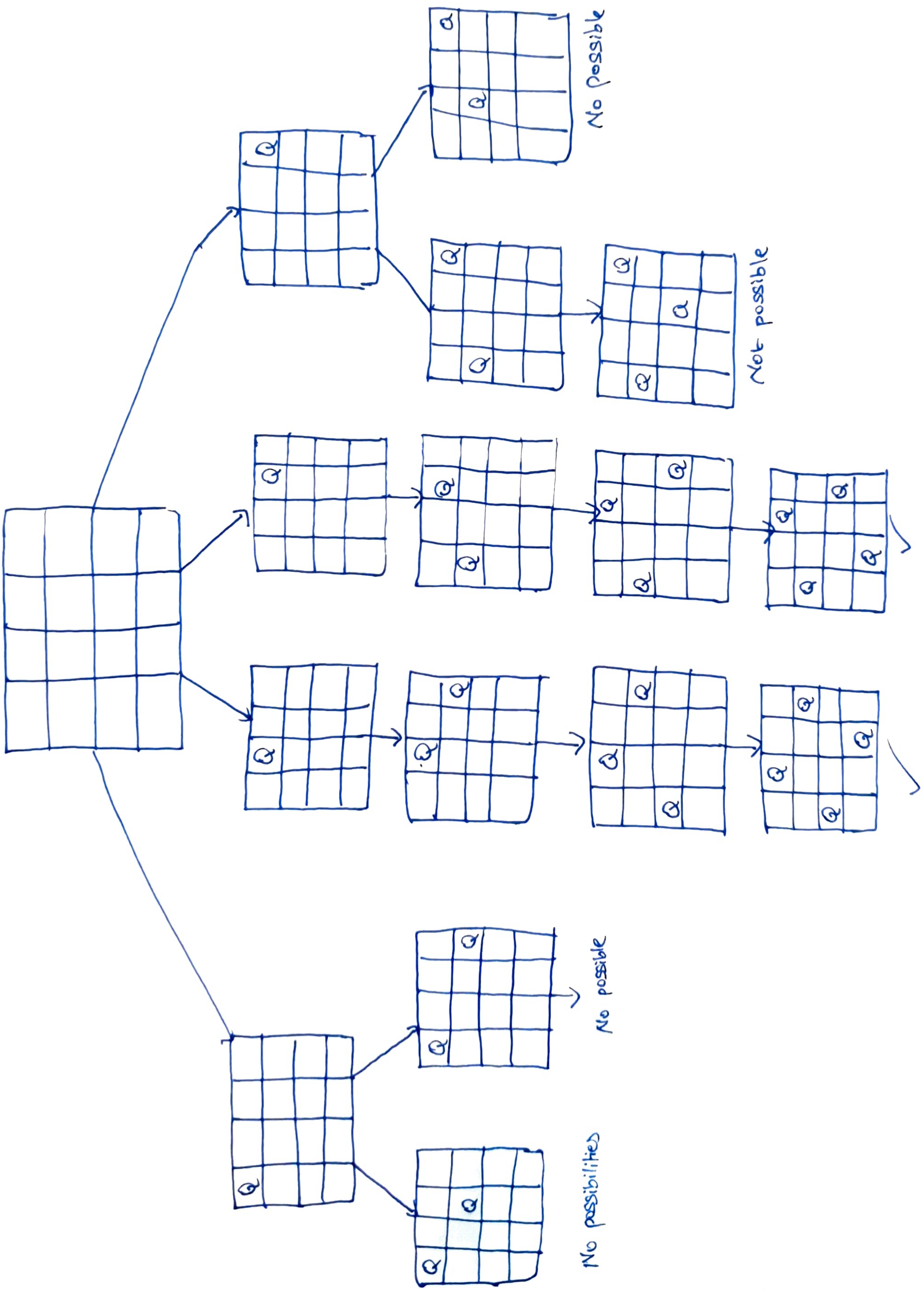
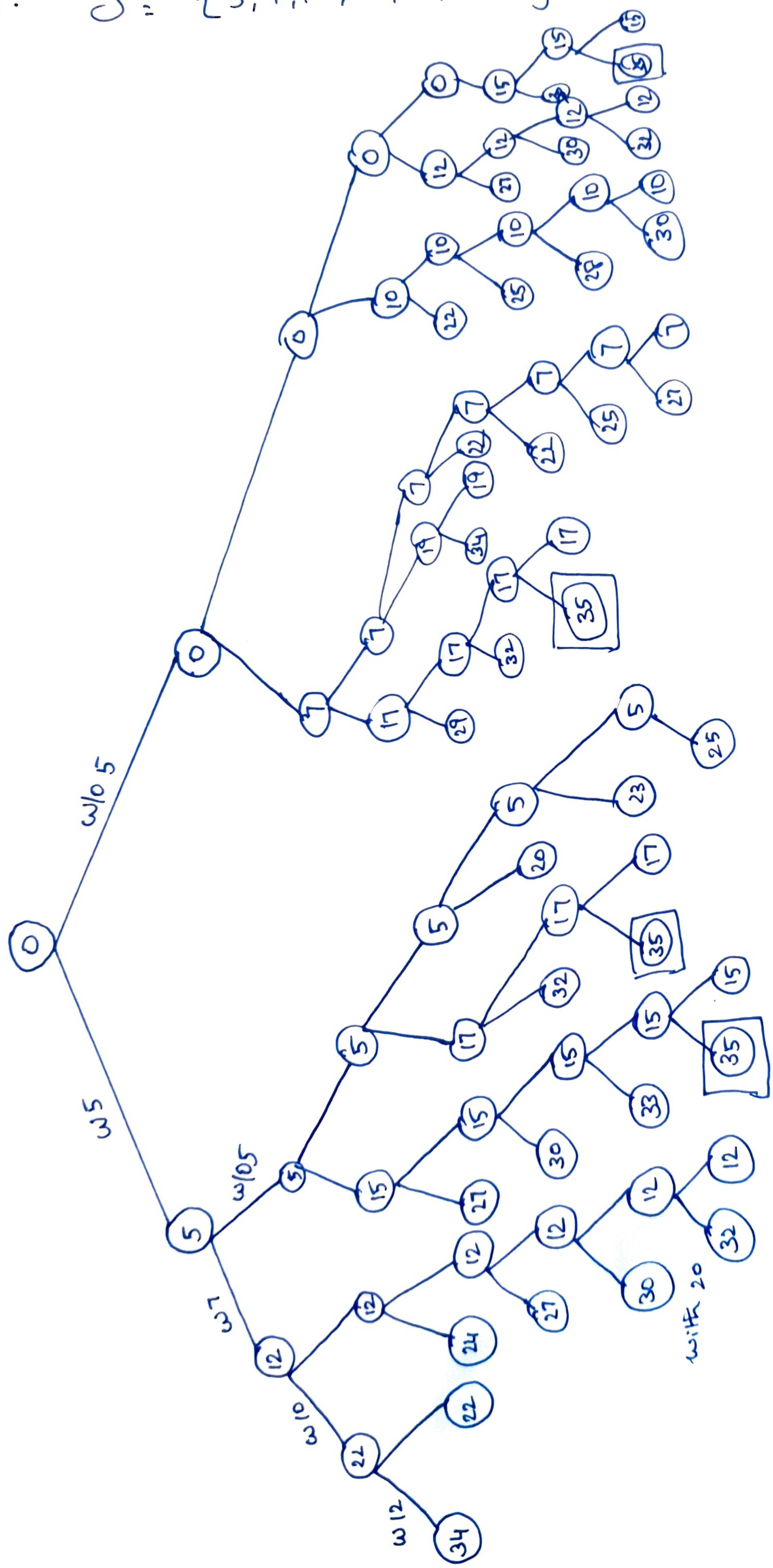
DAA Assignment

Senthil Sivaraman. S
CSE 'C'

1. 4- Queen's problem



2. $S = \{5, 7, 10, 12, 15, 18, 20\}$ $d = 35$



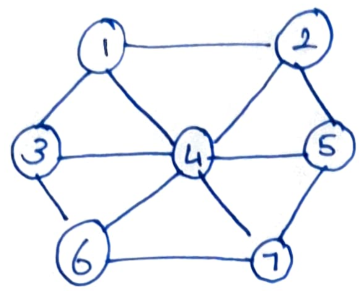
$S_1 = \{20, 10, 5\}$

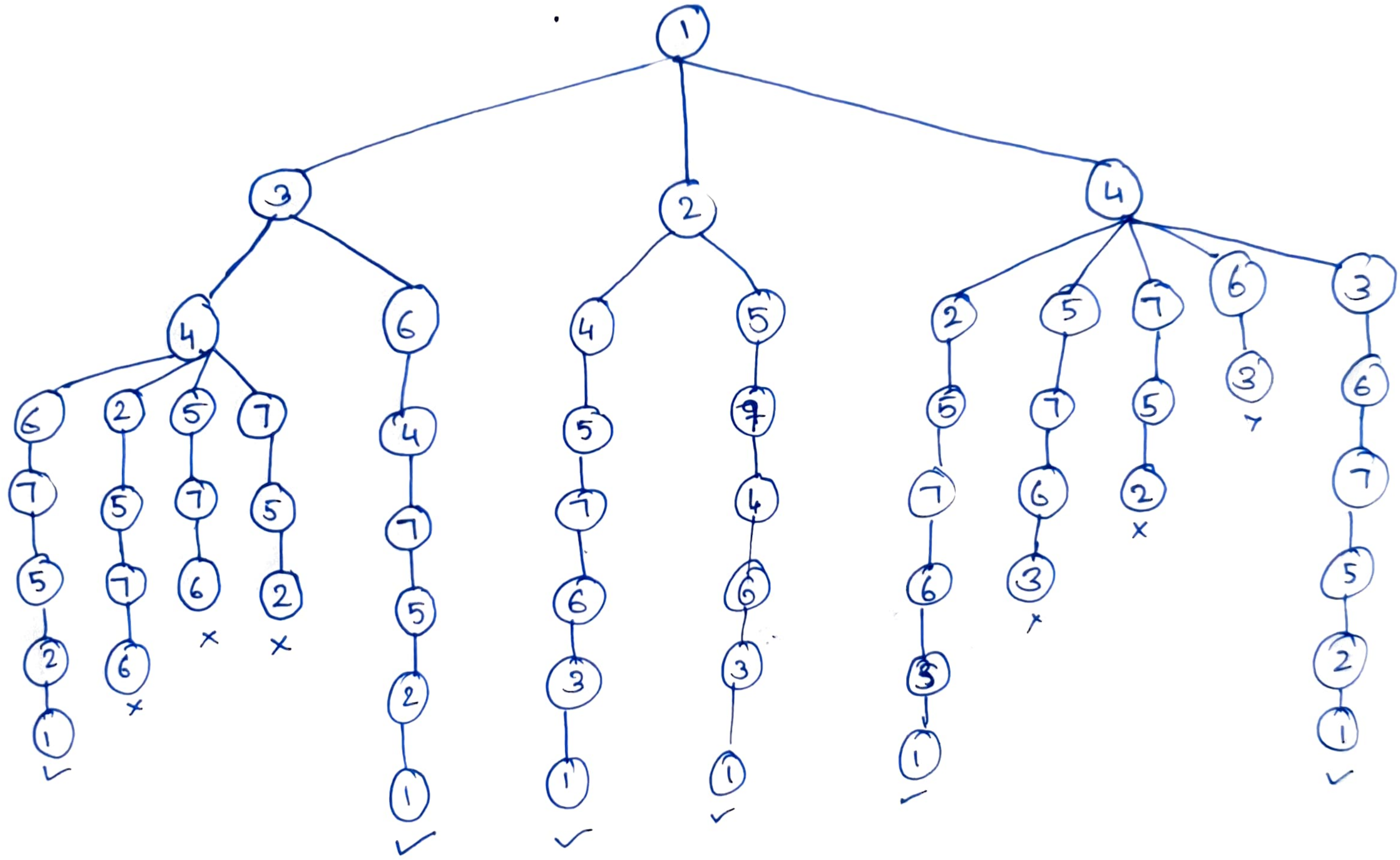
$S_2 = \{18, 12, 5\}$

$S_3 = \{18, 10, 7\}$

$S_4 = \{20, 15\}$

3.





Solution:

1, 3, 4, 6, 7, 5, 2, 1

1, 3, 6, 4, 7, 5, 2, 1

1, 2, 4, 5, 7, 6, 3, 1

1, 2, 5, 7, 4, 6, 3, 1

1, 4, 2, 5, 7, 6, 3, 1

1, 4, 3, 6, 7, 5, 2, 1

4. Solve Gaussian Elimination algorithm

$2x_1 + x_2 - x_3 = 4$

$x_1 + x_2 + x_3 = 2$

$x_1 - x_2 + 2x_3 = 8$

$$\begin{bmatrix} 2 & 1 & -1 & 4 \\ 1 & 1 & 1 & 2 \\ 1 & -1 & 2 & 8 \end{bmatrix} \quad \begin{array}{l} R_2 = R_2 - \frac{1}{2}R_1 \\ R_3 = R_3 - \frac{1}{2}R_1 \end{array}$$

$$\begin{bmatrix} 2 & 1 & -1 & 4 \\ 0 & 1/2 & 3/2 & 0 \\ 0 & -3/2 & 5/2 & 6 \end{bmatrix} \quad R_3 = R_3 + 3R_2$$

$$\begin{bmatrix} 2 & 1 & -1 & 4 \\ 0 & 1/2 & 3/2 & 0 \\ 0 & 0 & 7 & 6 \end{bmatrix}$$

$7x_3 = 6 \Rightarrow x_3 = 6/7$

$\frac{1}{2}x_2 + \frac{3}{2}x_3 = 0 \Rightarrow \frac{1}{2}x_2 + \frac{9}{7} = 0 \Rightarrow \frac{x_2}{2} = -\frac{9}{7} \Rightarrow x_2 = -\frac{18}{7}$

$2x_1 + x_2 - x_3 = 4$

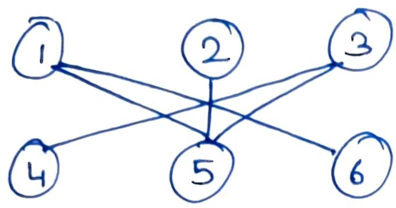
$2x_1 - 18/7 - 6/7 = 4$

$2x_1 = 4 + \frac{24}{7}$

$2x_1 = \frac{52}{7}$

$x_1 = \frac{26}{7}.$

5.



Algorithm for graph Colouring

mcolouring (t)

// This algorithm was formed using recursive backtracking

// The graph is represented by its boolean adjacency matrix

// G [1:n, 1:n]

// All the vertices are assigned distinct integers

// k is the index of the next colour to vertex.

repeat

{ // Generate all legal assignment for x[k]

Nextvalue (k)

if (x [k] = 0) then return; // No new

if (k = n) then // Almost m colors possible

write(x [1:n]);

else mcolouring (k+1);

until (false);

}

}

Algorithm Nextvalue (k)

// Generate next Color

// x[i] .... x[k-1] have been assigned int values.

// value of k is determined in the range of [0, m]

// If no Color exist then x[k] is 0.

```
repeat {
    x[k] = (x[k]+1) mode (m+1); // Next high color
    if (x[k] = 0) then return; // All colors used
    for j ← 1 to n do
    {
        if ((G[k,j] ≠ 0) and (x[k] = x[j]))
            // if (k,j) is edge & if adjacent have same color
                then break;
    }
    if (j = n+1) then return; // new color found
    until (false); // otherwise find next color.
}
```