Assignment 1

1. Identify connected components in an image

2. Print labelled image after Intermediate and Final processing steps

In [1]:

```
import numpy as np
```

Unprocessed Image

In [2]:

```
intensityMatrix = np.array([[0,0,0,0,0,0,0,0,0],
                            [0,0,0,0,0,0,1,0,0],
                            [0,1,0,1,1,0,1,1,0],
                            [0,1,1,0,1,0,1,1,0],
                            [0,1,1,0,0,0,0,1,1],
                            [0,0,1,0,0,0,0,1,1],
                            [0,1,1,0,0,0,1,1,1],
                            [0,1,0,0,0,0,0,0,0]])
```

Create another matrix 'labelMatrix' to store labels of connected components

In [3]:

```
labelMatrix = np.zeros([8,9], dtype=int)
```

In [4]:

```
print("Initial label matrix:\n\n{}".format(labelMatrix))
```

```
Initial label matrix:

[[0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]]
```

Algorithm to label 4-connected components

In [5]:

```
L = 1                   # label starts from 1
equalLabels = dict()    # dictionary to store new label for 8-connected components

# if first pixel of image is 1, label it as 1
if intensityMatrix[0][0] == 1:
    labelMatrix[0][0] = L
    L += 1

# for first row
for col in range(1, 9):
    if intensityMatrix[0][col] == 1:
        if intensityMatrix[0][col-1] == 1:
            labelMatrix[0][col] = labelMatrix[0][col-1]
        else:
            labelMatrix[0][col] = L
            L += 1
```

```python
# for first column
for row in range(1, 8):
    if(intensityMatrix[row][0] == 1):
        if intensityMatrix[row-1][0] == 1:
            labelMatrix[row][0] = labelMatrix[row-1][0]
        else:
            labelMatrix[row][0] = L
            L += 1

# for rest of the image cells
for row in range(1, 8):
    for col in range(1, 9):
        leftCell = intensityMatrix[row][col-1]
        leftCellLabel = labelMatrix[row][col-1]
        topCell = intensityMatrix[row-1][col]
        topCellLabel =labelMatrix[row-1][col]
        currCell = intensityMatrix[row][col]

        if currCell == 1:

            # left and top cells both have value 1
            # assign one of their labels to current pixel
            # make a note that the two labels are equal
            if leftCell == 1 and topCell == 1:
                if leftCellLabel != topCellLabel:
                    equalLabels[leftCellLabel] = topCellLabel
                labelMatrix[row][col] = topCellLabel

            # compare left cell, if value equals 1
            # assign left cell's label
            elif leftCell == 1:
                labelMatrix[row][col] = leftCellLabel

            # compare top cell, if value equals 1
            # assign top cell's label
            elif topCell == 1:
                labelMatrix[row][col] = topCellLabel

            # otherwise, assign a new label
            else:
                labelMatrix[row][col] = L
                L += 1
```

Intermediate Processing: Obtained Connected components after component labelling

In [6]:

```python
print("Connected components(before final processing) = {0}\n\n{1}\n".format(L-1,labe
```

```
Connected components(before final processing) = 5

[[0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 1 0 0]
 [0 2 0 3 3 0 1 1 0]
 [0 2 2 0 3 0 1 1 0]
 [0 2 2 0 0 0 0 1 1]
 [0 0 2 0 0 0 0 1 1]
 [0 4 2 0 0 0 5 1 1]
 [0 4 0 0 0 0 0 0 0]]
```

Final Processing Step: Obtained all truly Connected Components after replacing equivalent labels

In [7]:

```
# replace equivalent labels
for row in range(1, 8):
    for col in range(1, 9):
        if labelMatrix[row][col] in equalLabels:
            labelMatrix[row][col] = equalLabels[labelMatrix[row][col]]
```

In [8]:

```
print("Equivalent labels:\n\n{}\n".format(equalLabels))
print("Connected components post-processing:\n\n{}\n".format(labelMatrix))
print("Final Connected Components = {}".format(L-len(equalLabels)-1))
```

```
Equivalent labels:

{4: 2, 5: 1}

Connected components post-processing:

[[0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 1 0 0]
 [0 2 0 3 3 0 1 1 0]
 [0 2 2 0 3 0 1 1 0]
 [0 2 2 0 0 0 0 1 1]
 [0 0 2 0 0 0 0 1 1]
 [0 2 2 0 0 0 1 1 1]
 [0 2 0 0 0 0 0 0 0]]

Final Connected Components = 3
```