**Digital Image Processing**

**Title:** **Digit Recognition, Pattern Recognition, Classification**

**Objectives:** to understand template matching, Digit Recognition, Pattern Recognition, and solve a classification problem

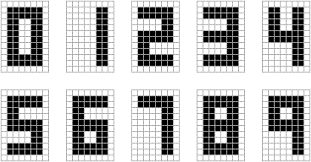
**Tools Used:** Python

**Procedure:** Open IDLE and perform the following tasks

Use the template matching algorithm (function) you wrote. Make a small dataset of digits (0 to 9) with 5 samples in each class. You can use test images with only single digits and of the same size as the template, for simplicity and ease, if you want. This means you don’t need to move the template window over test image. Both are of same size! Use KNN algorithm to find the digit in the test image.

Digit recognition is a classical pattern recognition problem. Each digit 0-9 represents a class. Features extracted from images of the digits can be used to generate a reference base (training data). A query digit image presented to the system can then be recognized using a classifier.

For this task, you are provided with multiple images of each digit as training data. Load each image, extract features and store the features of each image along with class label. Using k-nn classification scheme, classify each of the images in the test data set into one of the digit classes. Examples of digits are illustrated in the following table.



**Code:**

import cv2

clusters = [[],[],[],[],[],[],[],[],[],[]]

for i in range(10):

path = r'C:\Users\Naeem\Desktop\Jahanzeb\DIP\DIP Lab\Project 2\number\c' + str(i) + '.jpg'

template = cv2.imread(path, cv2.IMREAD\_GRAYSCALE)

clusters[i].append(template)

i = 11

for j in range(10):

for k in range(4):

path = r'C:\Users\Naeem\Desktop\Jahanzeb\DIP\DIP Lab\Project 2\number\t' + str(i+k) + '.jpg'

sample = cv2.imread(path, cv2.IMREAD\_GRAYSCALE)

clusters[j].append(sample)

i += 4

if i == 51:

break

test = cv2.imread(r'C:\Users\Naeem\Desktop\Jahanzeb\DIP\DIP Lab\Project 2\number\test.jpg', cv2.IMREAD\_GRAYSCALE)

distance = 10000000

closest = 0

for i in range (10):

for j in range(len(clusters[i])):

dist = clusters[i][j] - test

temp = list(map(sum, dist))

total = sum(temp)

if total < distance:

distance = total

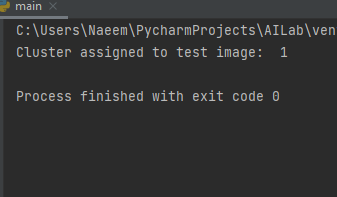
closest = i

print('Cluster assigned to test image: ', closest)

cv2.waitKey(0)

cv2.destroyAllWindows()

**Screenshot:**



Folder where images of digits are stored:

