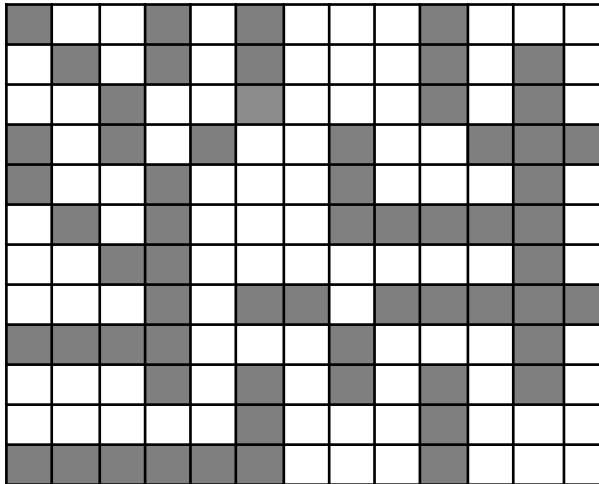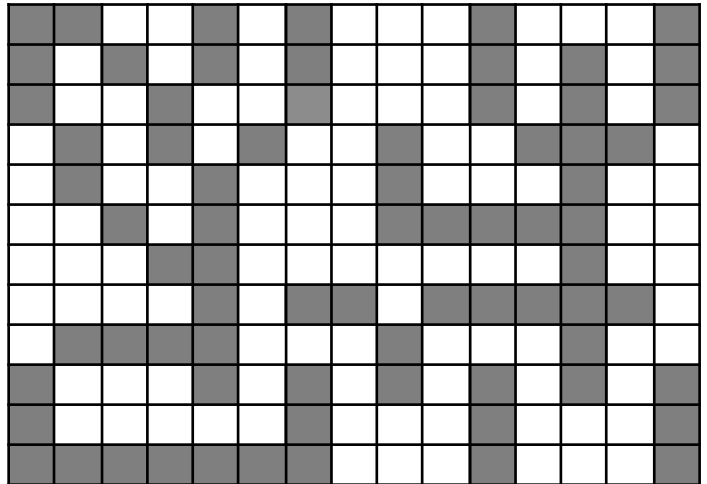1.   **Connected Components.** Non-programming exercise. Label the following image with colors starting 1.

   a.   *5 points.* 0.5 hrs. Use 4 connected to label the background (white) regions. Show the equivalence table and the final color in each box.

**Your Output for 4-connected Labeling:**   **Your Output for 8-connected Labeling:**

   b.   *5 points.* 0.5 hrs. Use 8 connected to label foreground (black) regions. Show the equivalence table and the final color in each box.

2.   **Handwritten Digit Recognition.** Use the digits.png file as template for digits 0, 1, …, 9. Write a python program to cut out each digit as a labeled dataset from 0..9, each of which is 20x20. <u>Note</u>: You may also use this *exact same dataset* with 100 samples of each digit 0..9 using 20 x 20 pixels from the internet along with libraries to read/load the dataset, if it's easier for you.

Rescale all images from 20 x 20 to 24 x 24. This can be done in memory as a python class data and need not be stored in a file. Use 80% of the data as the training set, reserving 20% for testing.
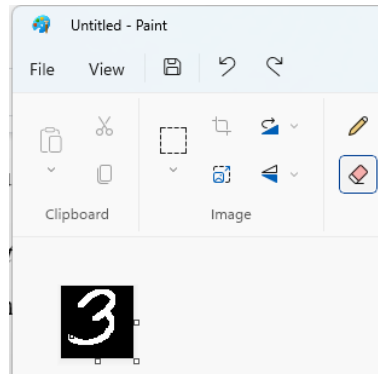
   a.   *30 pts.* 6 hrs. Then try recognition by using test images and report the accuracy percent for these 4 (classifier, feature type) combinations: (KNN K = 5, gray scale features), (KNN K = 5, HOG features), (KNN K = 1, gray scale features), (KNN K = 1, HOG features). For K = 5, if 2 or more digits have the same maximum vote count, just report one of them. For HOG, use 20° histogram orientations of non-directional gradients (ie., 9 bins) with 16 x 16 overlapping pixel windows for each 24 x 24 digit. Each digit will, thus, have 144 HOG features from 4 x 4 x 9, with 9 histo values x 4 per 16 x 16 block x 4 such blocks per 24 x 24 image.
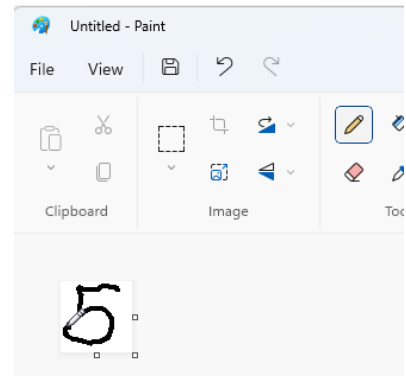
b. Use KNN K = 1 with HOG features to report:

   i. *5 pts. 1.0 hrs.* Result for 2 test images *per digit 0..9* cut out from **digits.png** itself but not aligned at the original 20 x 20 image, so you may have smaller or bigger sizes. You will have to rescale each test image to 24 x 24 because HOG must be recalculated after scaling.

   ii. *5 pts. 1.0 hrs.* Result for 2 test images *per digit 0..9* you create in a Paint program to see if you can find your character. Each test image should be big to start with such as 50x50, but you should rescale it to 24 x 24 before testing.

   iii. *5 pts. 1.0 hrs.* Result for 4 test images of digit 5 you create in Paint with white background.
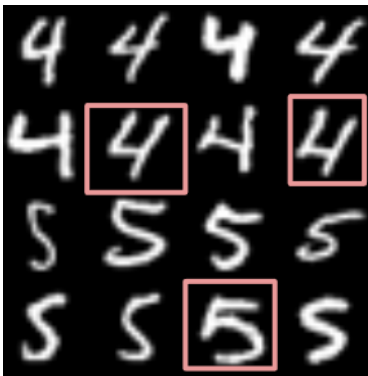
2. b. i. Cut digits at different sizes   2. b. ii. Create your own digits 0-9.   2. c. ii. "5" in white background.

c. *20 pts. 3 hrs.* For each 0..9 digit in your dataset of 100 characters, use OpenCV's auto threshold and then connected components to find the bounding box. Use that bounding box to cut out each gray-scale image and resize each back to 24 x 24. This will be your new dataset (training and testing, combined). Report the accuracy percent for KNN K = 1 using HOG features. Is the result here better than in problem 2a for KNN = 1 using HOG features?

digits.png Image Format:

- 0: Uses rows 1-5 (i = 0..99), each 100 characters, each 20 x 20.
- 1: Uses rows 6-10 (i = 100..199), each 100 characters, each 20 x 20.
- ...
- 9: Uses rows 46..50 (i = 900..999), each 100 characters, each 20 x 20.

Submission. All your work should be put into 1 pdf file and uploaded to the LMS before the due date.

Reminder Late Quiz Submission Policy. 7 days late allowed, after that 2 points deducted per day late.

Reminder Honor Code Agreement. Work copied from others is cheating, resulting in F grade.