

Instructions:

- 1) There are 50 points on the examination (14 points for Problem 1, 24 points for Problem 2 and 12 points for Problem 3).
- 2) This exam is “open book,” which means you are permitted to use any materials handed out in class, your own notes from the course, the text book, internet searches and anything
- 3) The exam must be taken completely alone. Showing it or discussing it with anyone is forbidden.
- 4) You may not consult with any other person regarding the exam
- 5) You may not copy answers (parts or all) from internet searches.
- 6) You must complete the exam in the answer sheet provided only
- 7) You can ask questions on discord in **#finalexam**, further announcement will be posted on **#announcement**
- 8) You must submit

(1) Answer sheet (in pdf)

(2) .py code for problem 2

In MyCourseVille before 17:59, December 9, 2021. Otherwise, the exam will not be accepted.

THE EXAM IS **DUE** ON 17:59, DECEMBER 9, 2021

Rules of Conduct for Students during Examinations

Unacceptable examination conduct includes

- 1) Communicating with other students or anyone regarding the exam.
- 2) Looking at exam materials of other students or anyone or allowing others to look at their exam materials.
- 3) Copy some or all of the answer from internet searches. (Suggest to write in your own word)

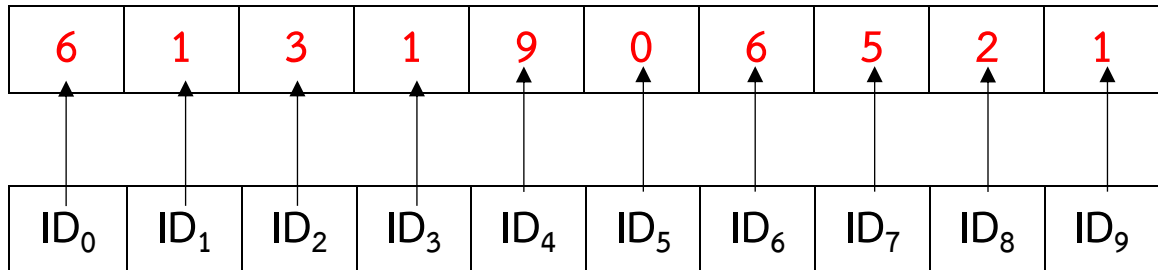
When a student cheats or is in suspicion, the supervisory officer has the power to investigate the matter. There are serious consequences to academic misconduct, including receiving an “F” on the work or examination, an “F” in the course.

Name_____ID_____

Problem 1 (14 points)

Preliminaries:

Your ID sequence will be used to create an image for Problem 1.1 and 1.2. $ID_0 - ID_9$ correspond each of your id as shown below, **for example,**



Then, use the quantization table below to quantize your ID into an intensity value in a range $[0, 3]$. The quantized IDs denote qID .

Original ID (ID_n)	Quantized ID (qID_n)
0-1	0
2-3	1
4-6	2
7-9	3

A 4x10 image created from your quantized ID has 4 different intensity levels in a range of $[0,3]$ below.

qID_0	qID_1	qID_2	qID_3	qID_4	qID_5	qID_6	qID_7	qID_8	qID_9
qID_0	qID_1	qID_2	qID_3	qID_4	qID_5	qID_6	qID_7	qID_8	qID_9
qID_9	qID_8	qID_7	qID_6	qID_5	qID_4	qID_3	qID_2	qID_1	qID_0
qID_9	qID_8	qID_7	qID_6	qID_5	qID_4	qID_3	qID_2	qID_1	qID_0

(Write in the answer sheet provided)

Fill in your ID here

↑	↑	↑	↑	↑	↑	↑	↑	↑	↑
ID ₀	ID ₁	ID ₂	ID ₃	ID ₄	ID ₅	ID ₆	ID ₇	ID ₈	ID ₉

Fill in your 4x10 quantized image here

Otsu's threshold is a non-parametric and unsupervised method for automated threshold selection. An image has L total levels in a range $[0, \dots, L - 1]$. To derive Otsu's threshold, the gray-level histogram is normalized and regarded as a probability distribution:

$$p_i = n_i/N$$

Where n_i is the number of pixels at level i and N is the total number of pixels. A pixel in the image can be classified into two classes, C_0 and C_1 (background and objects) by a threshold level k ; C_0 denotes pixels with level $[0, \dots, k]$ and C_1 denotes pixels with levels $[k + 1, \dots, L - 1]$. The probability of class occurrence of C_0 and C_1 denotes $\omega_0(k)$ and $\omega_1(k)$, respectively, and are given by

$$\omega_0(k) = \sum_{i=0}^k p_i$$

$$\omega_1(k) = \sum_{i=k+1}^{L-1} p_i = 1 - \omega_0(k)$$

Total mean μ_T is given by

$$\mu_T = \sum_{i=0}^{L-1} ip_i$$

The first-order cumulative moment $\mu(k)$ of the histogram up to level k is given by

$$\mu(k) = \sum_{i=0}^k ip_i$$

The optimal (Otsu's) threshold is selected from the value that can maximize the separability of the resultant classes or the between-class variance in the formula below

$$\text{Between-class variance } \sigma_B^2(k) = \frac{(\mu_T \omega_0(k) - \mu(k))^2}{\omega_0(k) \omega_1(k)}$$

Answer questions 1.1.1-1.1.4 in the answer sheet provided.

1.1.1 Determine total mean (μ_T)

Total mean (μ_T) =

1.1.2 Determine the values in the table

Intensity Level i	0	1	2	3
n_i				
p_i				

For each selected threshold level k , calculate the between-class variance $\sigma_B^2(k)$ and other parameters below.

Threshold level k	0	1	2	3
$\omega_0(k) = \sum_{i=0}^k p_i$				
$\mu(k) = \sum_{i=0}^k ip_i$				
$\omega_1(k) = 1 - \omega_0(k)$				
Between-Class Variance $\sigma_B^2(k)$				

1.1.3 The selected threshold k or Otsu's threshold =

--

1.1.4 Fill in values of the thresholded image after apply **inverse** thresholding using this Otsu's threshold by highlighting black color if the value = 0, and remain white box for value = 1.

1.2 Image Compression

1.2.1 Use the minimum fixed bits method to encode Code #1 to represent the 4 intensity levels. (Write in the answer sheet provided)

Intensity level (r_k)	Code #1
0	
1	
2	
3	

** Select code #1 should be a binary increment corresponding to the intensity levels.

1.2.2 Determine Huffman coding to encode Code #2 to represent the 4 intensity levels. For any level that has the same probabilities, then also rank them by intensity levels.

Intensity level (r_k)	Code #2	Probability of intensity occurrence $P(r_k)$
0		
1		
2		
3		

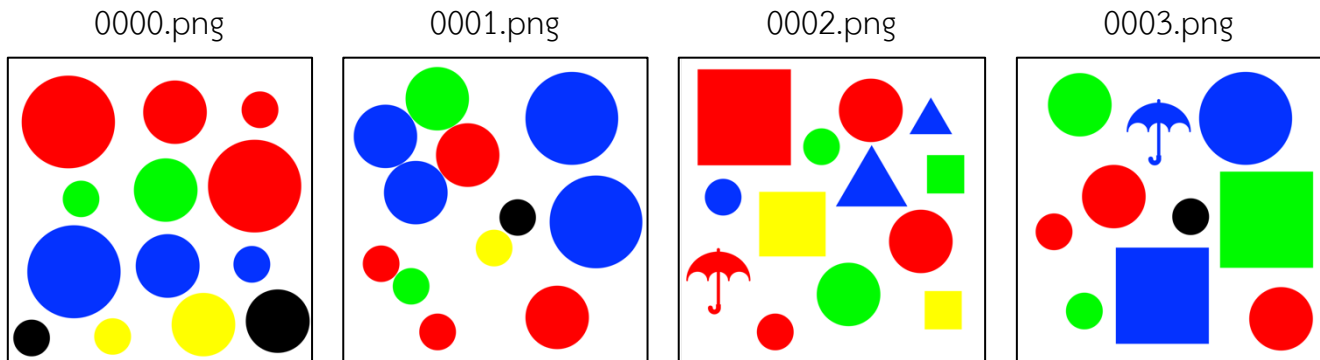
1.2.3 Encode this image sequence below using Code #1 & Code #2

ID ₀	ID ₁	ID ₂	ID ₃	ID ₄	ID ₅	ID ₆	ID ₇	ID ₈	ID ₉
-----------------	-----------------	-----------------	-----------------	-----------------	-----------------	-----------------	-----------------	-----------------	-----------------

Code #1	
Code #2	

Problem 2. Counting objects (24 points)

Write python code in the provided functions in the provided final_exam_p2.py file to count various objects from the image samples below.



The conditions of the image samples and other test samples (for later evaluation) are as follows:

- There are only 4 kinds of object shapes (circle, rectangle, triangle, umbrella), no other shape.
- No overlapping objects, may have slightly close to each other, such as 0001.png
- No rotation
- Rectangle, triangle and circle can have 3 different sizes (fixed size of small, medium and large) and 5 different colors (red, green, blue, black and yellow)
- Umbrella has only one size, but can have up to 4 different colors
- Image size is fixed

You are required to write the following functions. The function descriptions and required input/output are listed below.

1. To count all objects in an input image

```
def count_all_objs(image_bgr):  
    # input: image_bgr - is a bgr image read by opencv lib  
    # output: (1) num_count is the number of all objects in image_bgr  
    #          (2) result_image is a bgr image (your own design) to visualize  
    the results PROPERLY.  
  
    # write your code here  
  
    return num_count, result_image
```

2. To count the number of circles in an input image

```
def count_circles(image_bgr):  
    # input: image_bgr - is a bgr image read by opencv lib  
    # output: (1) num_count is the number of circles in image_bgr  
    #          (2) result_image is a bgr image (your own design) to visualize  
    the results PROPERLY.  
  
    # write your code here  
  
    return num_count, result_image
```

3. To count the number of small-size circles in an input image

```
def count_small_circles(image_bgr):  
    # input: image_bgr - is a bgr image read by opencv lib  
    # output: (1) num_count is the number of small circles in image_bgr  
    #          (2) result_image is a bgr image (your own design) to visualize  
    the results PROPERLY.  
  
    # write your code here  
  
    return num_count, result_image
```

4. To count the number of small-size circles in an input image

```
def count_med_circles(image_bgr):  
    # input: image_bgr - is a bgr image read by opencv lib  
    # output: (1) num_count is the number of medium-size circles in image_bgr  
    #          (2) result_image is a bgr image (your own design) to visualize  
    the results PROPERLY.  
  
    # write your code here  
  
    return num_count, result_image
```

5. To count the number of large-size circles in an input image

```
def count_large_circles(image_bgr):  
    # input: image_bgr - is a bgr image read by opencv lib  
    # output: (1) num_count is the number of large circles in image_bgr  
    #          (2) result_image is a bgr image (your own design) to visualize  
    the results PROPERLY.  
  
    # write your code here  
  
    return num_count, result_image
```

6. To count the number of triangles in an input image

```
def count_triangles(image_bgr):  
    # input: image_bgr - is a bgr image read by opencv lib  
    # output: (1) num_count is the number of triangles in image_bgr  
    #          (2) result_image is a bgr image (your own design) to visualize  
    the results PROPERLY.  
  
    # write your code here  
  
    return num_count, result_image
```

7. To count the number of rectangles in an input image

```
def count_rectangles(image_bgr):  
    # input: image_bgr - is a bgr image read by opencv lib  
    # output: (1) num_count is the number of rectangles in image_bgr  
    #          (2) result_image is a bgr image (your own design) to visualize  
    the results PROPERLY.  
  
    # write your code here  
  
    return num_count, result_image
```

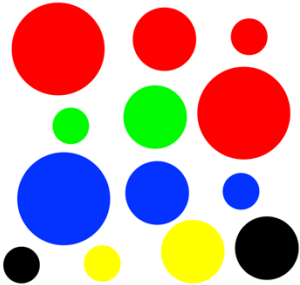

8. To count the number of umbrellas in an input image

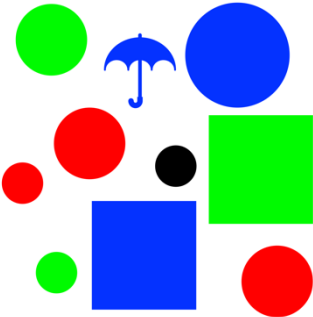
```
def count_umbrella(image_bgr):  
    # input: image_bgr - is a bgr image read by opencv lib  
    # output: (1) num_count is the number of umbrellas in image_bgr  
    #          (2) result_image is a bgr image (your own design) to visualize  
    the results PROPERLY.  
  
    # write your code here  
  
    return num_count, result_image
```

9. There also areas where you can write your own functions, if needed.

```
# write your own function here (if needed)
```

For the results, you can use the uncommented code below and it should show results like this,

Test Code	Possible results
<p>For 0000.png,</p>  <pre> img_rgb = cv2.imread("0000.png") num_count,result_image = count_all_objs(img_rgb) plt.imshow(result_image[:,:,:-1]);plt.show() print("Num of all objects: ", num_count) num_count,result_image = count_circles(img_rgb) print("Num of circles: ", num_count) num_count,result_image = count_small_circles(img_rgb) print("Num of small circles: ", num_count) num_count,result_image = count_med_circles(img_rgb) print("Num of medium circles: ", num_count) num_count,result_image = count_large_circles(img_rgb) print("Num of large circles: ", num_count) num_count,result_image = count_rectangles(img_rgb) print("Num of rectangles: ", num_count) num_count,result_image = count_triangles(img_rgb) print("Num of triangles: ", num_count) num_count,result_image = count_umbrella(img_rgb) print("Num of umbrella: ", num_count) print("-----") ") </pre>	 <pre> Num of all objects: 13 Num of circles: 13 Num of small circles: 5 Num of medium circles: 5 Num of large circles: 3 Num of rectangles: 0 Num of triangles: 0 Num of umbrella: 0 </pre>



```

img_rgb = cv2.imread("0002.png")
num_count,result_image = count_all_objs(img_rgb)
print("Num of all objects: ", num_count)
num_count,result_image = count_circles(img_rgb)
print("Num of circles: ", num_count)
num_count,result_image = count_small_circles(img_rgb)
print("Num of small circles: ", num_count)
num_count,result_image = count_med_circles(img_rgb)
print("Num of medium circles: ", num_count)
num_count,result_image = count_large_circles(img_rgb)
print("Num of large circles: ", num_count)
num_count,result_image = count_rectangles(img_rgb)
print("Num of rectangles: ", num_count)
num_count,result_image = count_triangles(img_rgb)
print("Num of triangles: ", num_count)
num_count,result_image = count_umbrella(img_rgb)
print("Num of umbrella: ", num_count)

```

```

Num of all objects: 10
Num of circles: 7
Num of small circles: 3
Num of medium circles: 3
Num of large circles: 1
Num of rectangles: 2
Num of triangles: 0
Num of umbrella: 1

```

Answer the following questions in the answer sheet provided and upload .py in mcv

2.1) Describe steps of your implementation and key parameters

Steps	Description and purposes
1	
2	
3	

2.2) Captured results

Image	Results (Capture the num_count results of your implemented functions here)
0000.png	
0001.png	
0002.png	
0003.png	

**** in case your code cannot run successful, the result here will help you get some marks.**

2.3) Explain and show how you represent the result_image for each case. Show examples.

Problem 3. Select a topic from Assist. Prof. Dr. Kitiwat' talk. Study on the topic, explore relevant imaging / computer vision techniques that can possibly solve some problems on image data of your selected topic. Write a paragraph (**no more than 350 words**) to summarize the ideas using your own word. (12 points)

Guidelines: you should describe your selected topic and its challenges, if any. Describe the relevant imaging techniques corresponding to the characteristics of the image data of the problems. Report the results (from online research papers / articles) and then analyze the limitations of the techniques on the image data. All might be mentioned in the research articles, but you must paraphrase and summarize in your own word. Also, add the references of the research articles you have mentioned in the essay. (for 2110431, you can write either in Thai or Eng)

Write the answer in the answer sheet provided.