Assume that im contains an array of uint8 values with the following shape: (100,200).

The following function displays that array as a grayscale image:

```
import matplotlib.pyplot as plt
plt.imshow(im, vmin=0, vmax=255, cmap=”gray”)
```

Discuss the intuitive effect of each of the following calls on the displayed image:

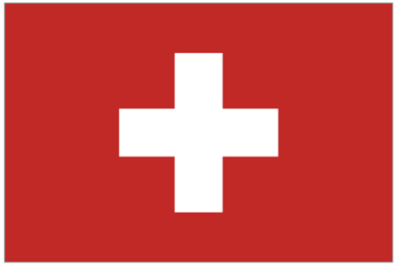
- plt.imshow(im, vmin=0, vmax=100, cmap=”gray”)
- plt.imshow(im, vmin=100, vmax=255, cmap=”gray”)
- plt.imshow(im, vmin=100, vmax=155, cmap=”gray”)
- plt.imshow(255-im, vmin=0, vmax=255, cmap=”gray”)

Briefly explain the effect of each of the following calls: what does new_im contain?

- new_im = im[50:,:]
- new_im = im[::4,:]
- new_im = im[:,::-2]
-

The array flag contains the flag of switzerland as an RGB image:

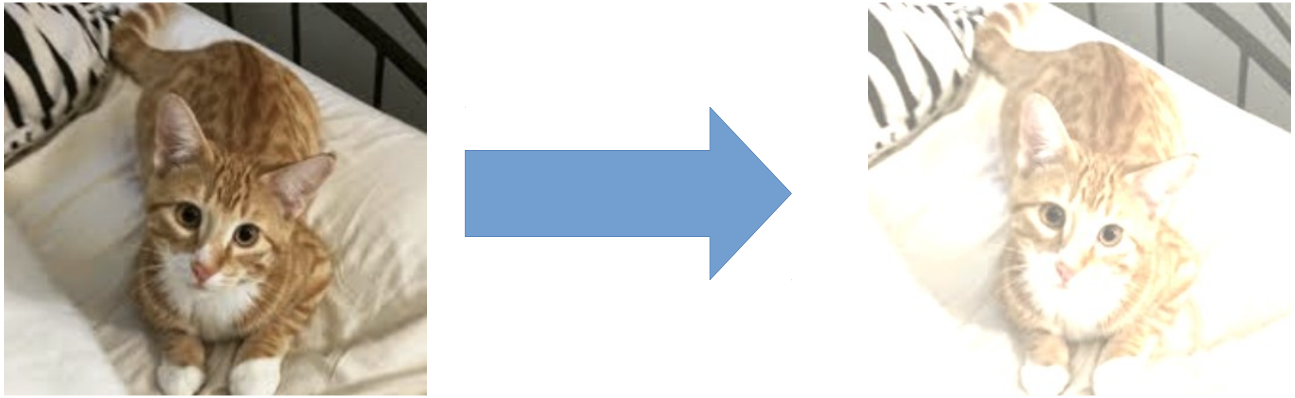
```
>> print(flag.dtype)
uint8

>> print(flag.shape)
(200,300,3)

>> plt.imshow(flag)
```



Write or briefly describe the output you expect from each of these commands

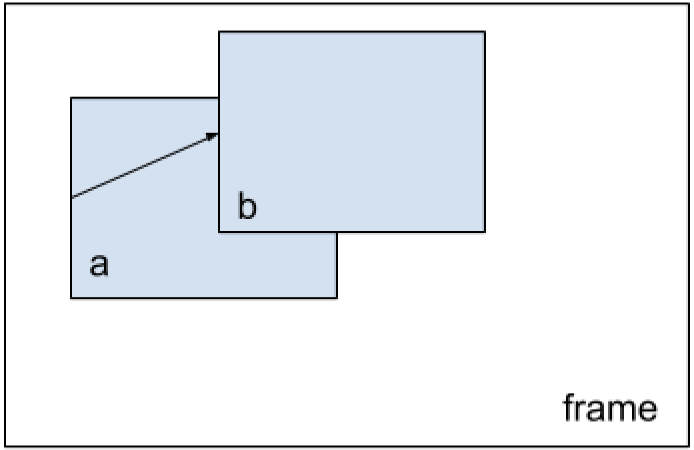
- `print(flag[0,0,0], flag[0,0,1], flag[0,0,2])`
- `print(flag[:,150,1])`
- `plt.imshow(flag[:,:,::-1])`

You are asked to implement a function that, given an RGB image as a numpy array with shape (N,M,3) and dtype uint8, returns a brightened version of that image. Briefly describe how you could implement it, or (better) write such function using python or pseudocode.



You are given a grayscale video containing 100 frames. The video is stored in an array with shape (100,N,M). Each frame, accessible as video[frame_number,:,:], is a grayscale image.

The video depicts a black rectangle on a white background, moving from position a (in frame 0) to position b (in frame 99) with a constant speed.



Briefly discuss how each of the following images looks like:

- np.max(video, axis=0)
- np.min(video, axis=0)
- np.mean(video, axis=0)
- np.median(video, axis=0)

The following line of code returns one array of numbers. Briefly describe what it represents:

- video[:,a,b]

Discuss the contents of such array, depending on the values of a and b.

Given the following matrix, compute the result of the convolution with a square 3x3 kernel. Consider “valid” padding, i.e. only compute the convolution for windows that fully overlap the input image

Matrix:

0	2	2	0	0
0	2	0	0	0
0	2	0	0	0
2	4	10	0	0

Kernel:

1	1	1
1	1	1
1	1	1

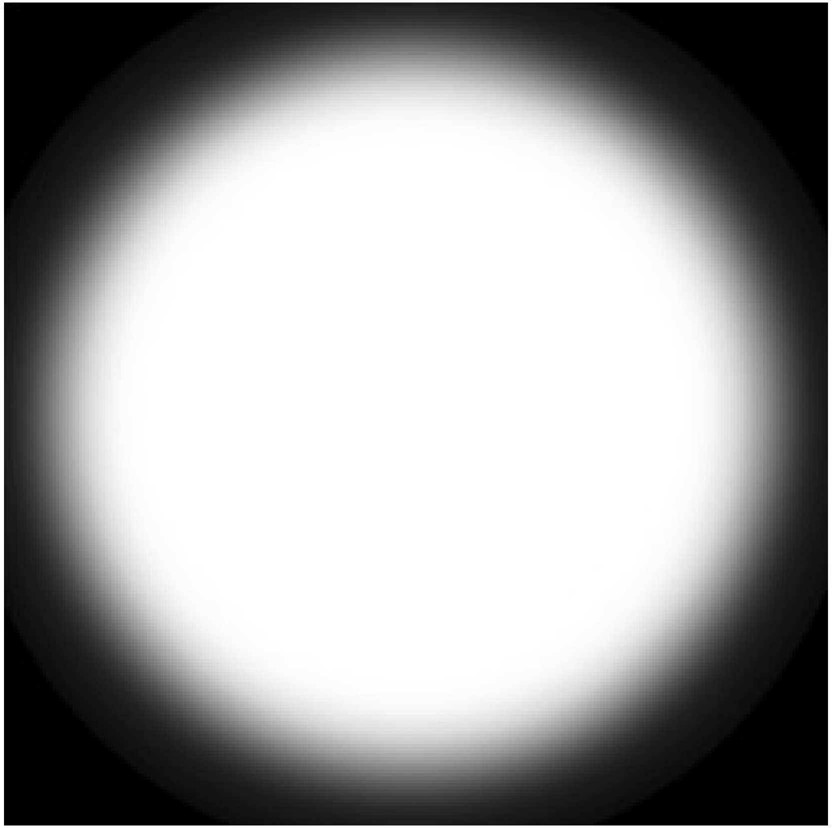
Consider the convolution of the previous question.

What is the visual effect of such convolution on a grayscale image?

Assume we are given a grayscale image which is completely black, except for a few isolated white pixels. What is the result of the convolution with the above-mentioned kernel?

Consider the following grayscale image

- Briefly comment on the direction and modulo of the gradient on every point in the image.
- Mark on the image the areas (if any) where you expect the gradient vector to have zero length



Briefly discuss the relation between image edges and convolutions.

Explain in one short sentence why smoothing is used in the Canny algorithm.

You are given a grayscale image im representing a number of gray objects (such as coins) on a dark background.

You threshold im as follows:

```
imb = im > t
```

imb is now a binary image

Answer true/false and briefly motivate:

- If you increase t, the number of True pixels in imb does not increase
- If you increase t, the number of connected components in imb (i.e. “islands” of True pixels) will either decrease or stay the same

You are given a grayscale uint8 image with a uniform gray background

On this image, there are some uniformly white objects and some uniformly black objects.

Write the pseudocode for a function that:

- Returns the number of the black objects
- Returns the cropped image of the largest white object

Consider the hough algorithm for finding lines.

There are multiple possible parametrizations for lines in the hough space: for example, we could represent lines by their angular coefficient and intercept. Briefly discuss why this is not a good idea, and what parametrization is usually adopted for lines in the hough space

Consider four points at the vertices of a square in the image space. Discuss their representation in hough space.

What is the purpose of binning in the hough algorithm? Choosing smaller bins yields better accuracy; does it also has disadvantages?

We want to design a detector for SQUARES with a given (known) edge length E based on the hough approach.

Assume that we expect the squares to be straight, i.e. that their edges are aligned to the x and y axes of the image. Also, the edge length E is known a priori and well defined (say: E=100 pixels).

- Describe a possible parametrization for your models. Hint: just like we parametrize a line by d and theta, we can paramerize a square by…
- How many dimensions does the hough space have?
- What does a point in the hough space represent in the image?
- How is a point in the image represented in the hough space?
- Assume we have a single square visible in the image. How does the hough space look like?
- Assume we have a single square visible in the image, but we can only detect the lateral (vertical) edges. How does the hough space look like?

(difficult): assume we do not know a priori that the squares we are looking for are aligned to the image axes. How can we modify our approach in order to handle this case?

- How many dimensions does the hough space have?
- What does a point in the hough space represent in the image?
- How is a point in the image represented in the hough space?
- Assume we have a single square visible in the image. How does the hough space look like?

Consider a convolutional layer with 2 input feature maps, 3 output feature maps, and a square convolution kernel with edge k.

Write the expression returning the number of parameters of this layer as a function of k. Also include bias parameters.

Briefly discuss:

- What do we mean when we say that convolutional layers enable “parameter sharing”? Why is that considered an important explanation of their good performance on images?
- What do we mean when we say that convolutional layers have “sparse connectivity”?

Consider the following convnet architectures:

- Architecture 1:
  - Convolution 4x4
  - Convolution 3x3
- Architecture 2:
  - Convolution 2x2
  - Max-Pooling 2x2
  - Convolution 2x2

- Compute the number of parameters of each
- Consider a neuron in the output feature map of each; compute the size of its receptive field.