Introduction

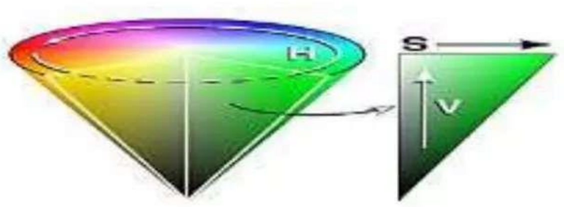
- The characteristics of color image are distinguished by its properties brightness, hue and saturation.
- simplifies object extraction and identification.
  - ✓ Motivation to use color
  - ✓ Brightness
  - ✓ Hue

Motivation to use color:

- Powerful descriptor that often simplifies object identification and extraction from a scene
- Humans can discern thousands of colour shades and intensities, compared to about only two dozen shades of gray

Hue:

- Attribute associated with the dominant wavelength in a mixture of light waves
- Hue is somewhat synonymous to what we usually refer to as "colors". Red, green, blue, yellow, and orange are a few examples of different hues.
- Mean wavelength of the spectrum

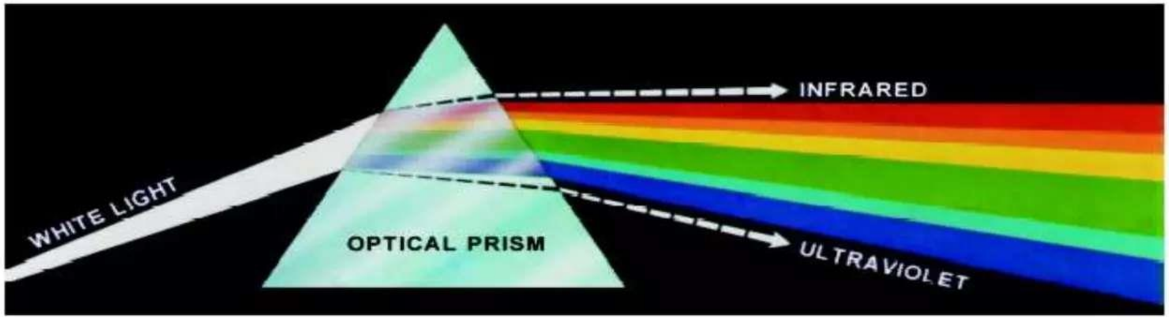


Brightness:

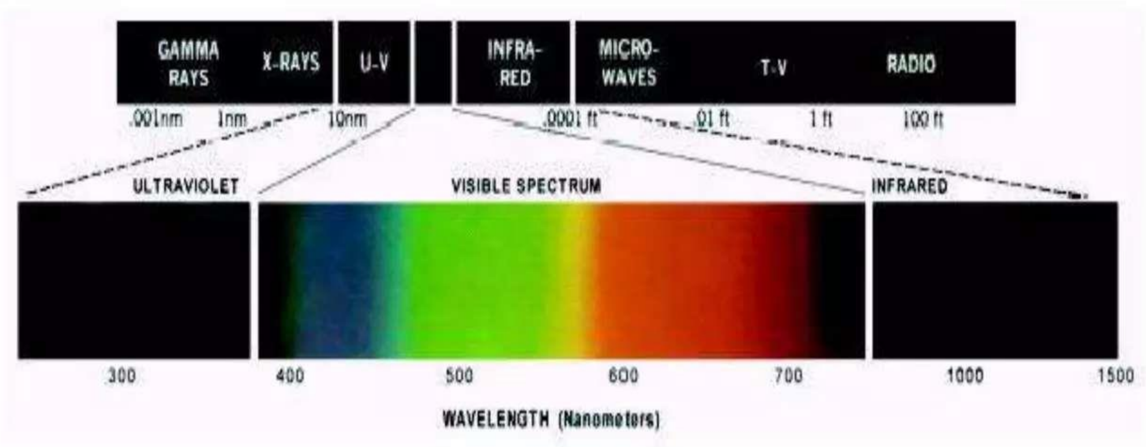
- Intensity
- Perceived luminance
- Depends on surrounding luminance

Color Fundamental:

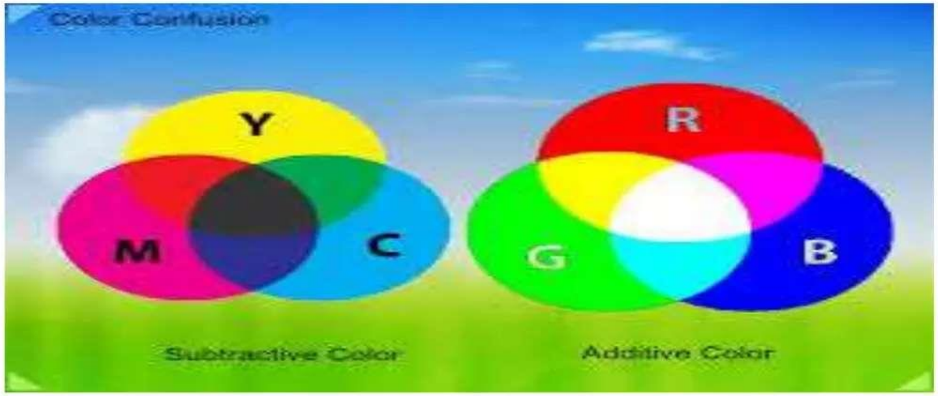
- In 1666 Sir Isaac Newton discovered that when a beam of sunlight passes through a glass prism, the emerging beam is split into a spectrum of colors



- A chromatic light source, there are 3 attributes to describe the quality:



- Primary colors can be added to produce the *secondary colors of light:*
  - ✓ Cyan (green plus blue)
  - ✓ Yellow (red plus green)
  - ✓ Magenta (red plus blue)



- The three basic quantitles useds to describe the quantity of a chromatic light source are:
  - ✓ Radiance
  - ✓ Luminance
  - ✓ Brightness

Radiance:

- The total amount of energy that flows from the light source (measured in watts)

Luminance:

- The amount of energy an observer *perceives* from the light source (measured in lumens)
- we can have high radiance, but low luminance

Brightness:

- A subjective (practically unmeasurable) notion that embodies the intensity of light

Changing brightness and contrast

- **Changing brightness and contrast**: To adjust the brightness and contrast of an image in Python, we can use the OpenCV or Pillow library.

To increase the brightness of an image using OpenCV, use

cv2.add() function

To adjust the contrast, we can use the **cv2.convertScaleAbs() function.**

```
import cv2


img = cv2.imread('image.jpg')          # Load the image
                                     # Adjust brightness and contrast
alpha = 1.5 #                        Increase brightness
beta = 50 #                          Increase contrast
adjusted = cv2.convertScaleAbs(img, alpha=alpha,
beta=beta)
                                     # Display the adjusted image
cv2.imshow('Adjusted Image', adjusted)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

- import cv2
- import matplotlib.pyplot as plt
- import numpy as np
- image =cv2.imread('D://SJC//ImageProcessing//images//image2.jpg') plt.subplot(1, 2, 1)
- plt.title("Original")
- plt.imshow(image)
- brightness = 10
- contrast = 2.3
- image2 = cv2.addWeighted(image, contrast, np.zeros(image.shape, image.dtype), 0, brightness)
- cv2.imwrite('modified_image.jpg', image2)
- plt.subplot(1, 2, 2) plt.title("Brightness & contrast")
- plt.imshow(image2) plt.show()

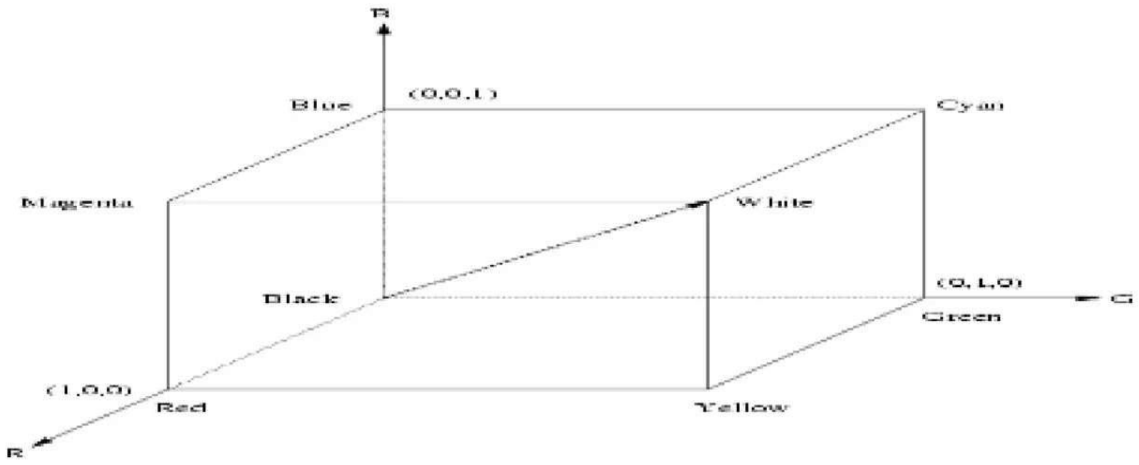
TYPES OF COLOR MODELS:

- ✓ RGB Model
- ✓ CMY Model
- ✓ HSI Model
- ✓ YIQ Model

RGB Model:

- ➢ Color monitor, color video cameras
- ➢ In the RGB model, an image consists of three independent image planes, one in each of the primary colors: red, green and blue.
- ➢ Specifying a particular colour is by specifying the amount of each of the primary components present.
- ➢ The geometry of the RGB colour model for specifying colors using a Cartesian coordinate system. The greyscale spectrum,

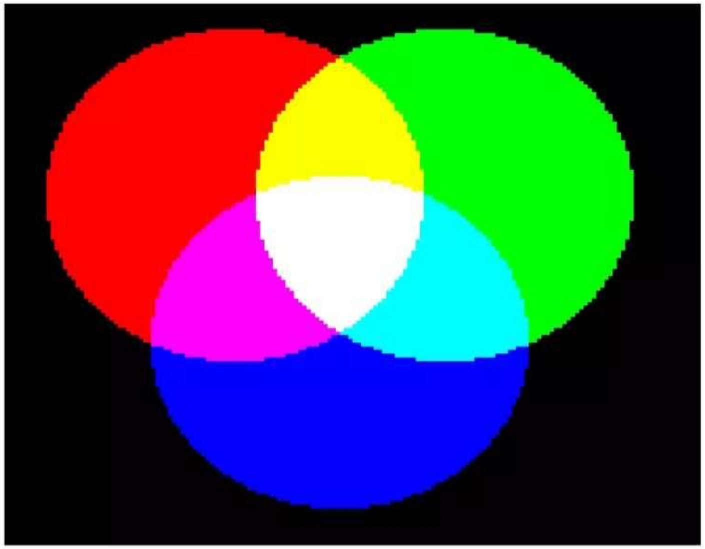
.

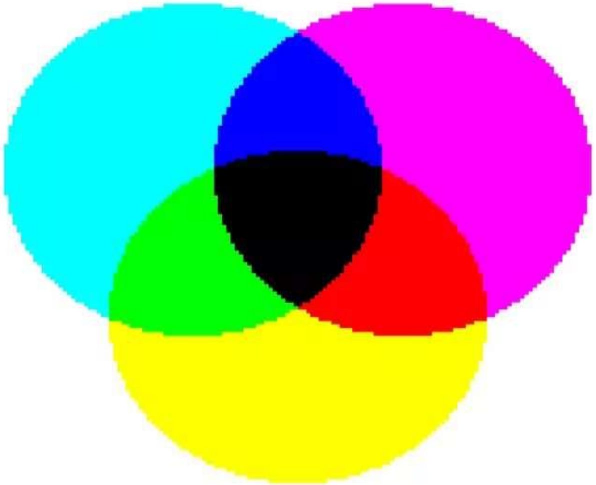


- The RGB color cube. The grayscale spectrum lies on the line joining the black and white vertices.

CMY Model:

- The CMY (cyan-magenta-yellow) model is a *subtractive* model appropriate to absorption of colors, for example due to pigments in paints
- Whereas the RGB model asks what is added to black to get a particular color, the CMY model asks what is subtracted from white.
- In this case, the primaries are cyan, magenta and yellow, with red, green and blue as secondary colors

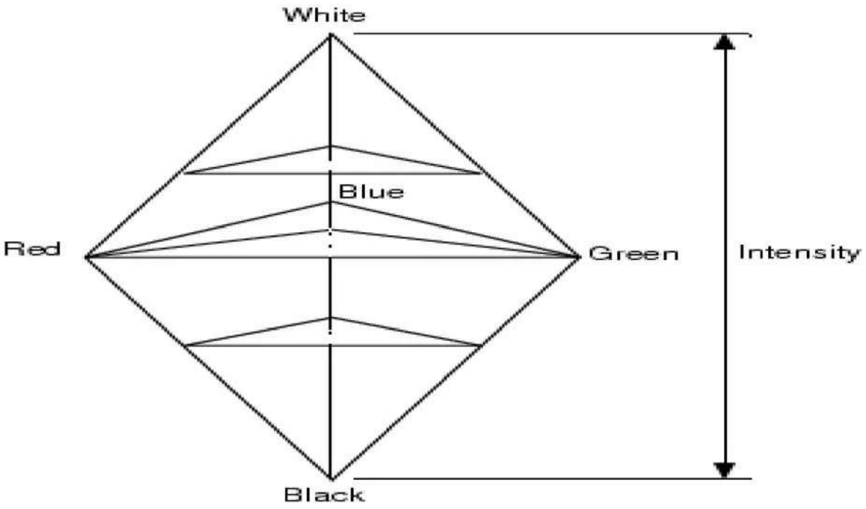


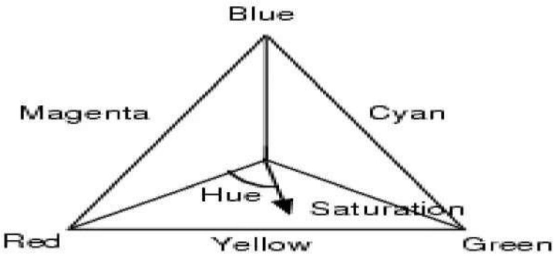


- The relationship between the RGB and CMY

HSI Model:

- As mentioned above, colour may be specified by the three quantities hue, saturation and intensity.
- This is the HSI model, and the entire space of colors that may be specified in this way is shown





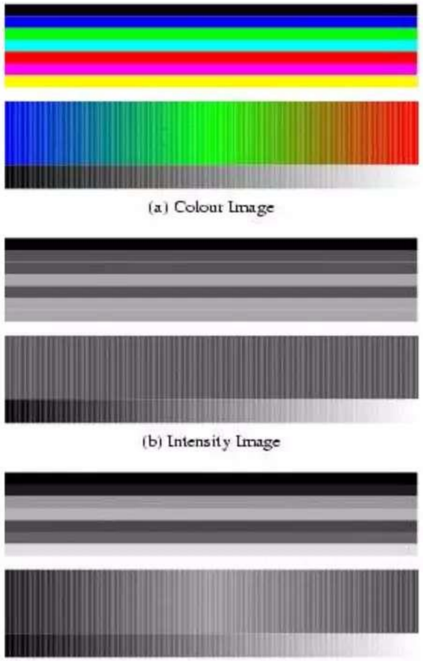
- Conversion between the RGB model and the HSI model is quite complicated. The intensity is given by

$$I = R+G+B$$

- where the quantities R, G and B are the amounts of the red, green and blue components, normalised to the range [0,1]. The intensity is therefore just the average of the red, green and blue components.
- The saturation is given by:S = 1 –min

YIQ Model:

- The YIQ (luminance-inphase-quadrature)model is a recoding of RGB for colour television, and is a very important model for colour image processing. The importance of luminance was discussed in
- The conversion from RGB to YIQ is given by:
- The luminance (Y) component contains all the information required for black and white television, and captures our perception of the relative brightness particular colors.



(a) Colour Image

(b) Intensity Image

(c) Luminance Image

What is Hue?

- The ***hue*** of a color is a component of its chromaticity. Red, green, and blue are the three main colors of light. It is why televisions, computers, and other electronic color visual displays use a ratio of red, green, and blue phosphors to generate all electrically conveyed colors.
- Hue is a single value that describes the color of something and is typically measured in degrees.
- It has the colors red, orange, yellow, green, blue, purple, and magenta all the way back to red.
- Although, magenta and pink colors are not light frequencies, and a rainbow may prove it.
- It begins with red and progresses to different colors, but it doesn't contain magenta and pink because they are not genuine frequencies humans can see.

What is Saturation?

- **Saturation**: The intensity of a color. As saturation increases, colors appear sharper or purer. As saturation decreases, colors appear more washed-out or faded
- ***Saturation*** is defined by the purity of the color and its distance from the grey color. If a color has much more greyness, it has a lower saturation level. Moreover, saturation could be viewed as the hue's dominance in the color.
- The outermost edge of the hue wheel includes the pure hue; as you move inside the wheel to the centre, which contains grey, the hue steadily drops, and the saturation likewise falls.
- It relates to a physical property known as the excitation property, which measures the percentage of brightness mixed with the dominant or pure color.

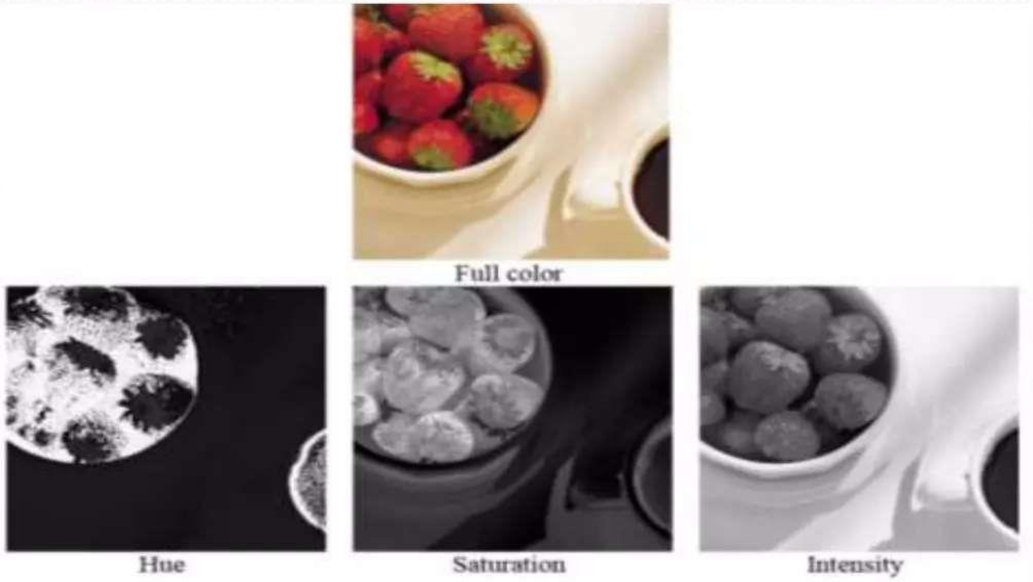
Saturation

- Saturation: Saturation refers to the intensity of colors in an image, and it can be adjusted to make an image look more or less vivid.
- To adjust saturation in Python, you can use the cv2.cvtColor() function in OpenCV to convert an image from one color space to another, and then adjust the saturation channel.

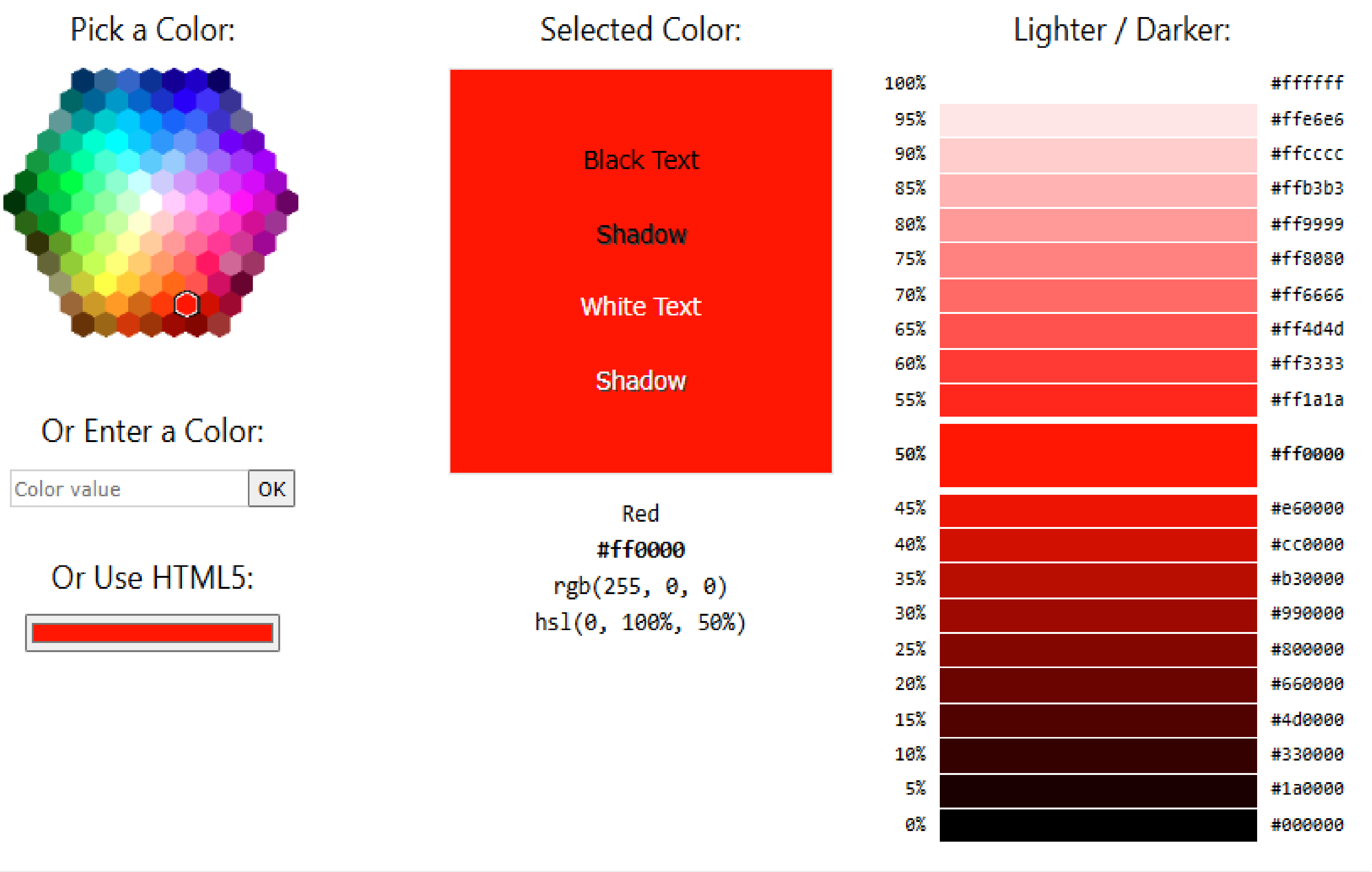
The HIS color model:

- The RGB and CMY color models are not suited for describing colors in terms of human interpretation. When we view a color object, we describe it by its hue, saturation, and brightness(intensity). Hence the HSI color model has been presented.
- The HSI model decouples the intensity component from the color-carrying information (hue and saturation) in a color image. As a result, this model is an ideal tool for developing color image processing algorithms.

- The hue, saturation, and intensity values can be obtained from the RGB color cube.



- That is, we can convert any RGB point to a correspondin point is the HSI color model by working out the geometrical formulas.



Pick a Color:

Selected Color:

Lighter / Darker:

Black Text

Shadow

White Text

Shadow

Or Enter a Color:

Color value [OK]

Or Use HTML5:

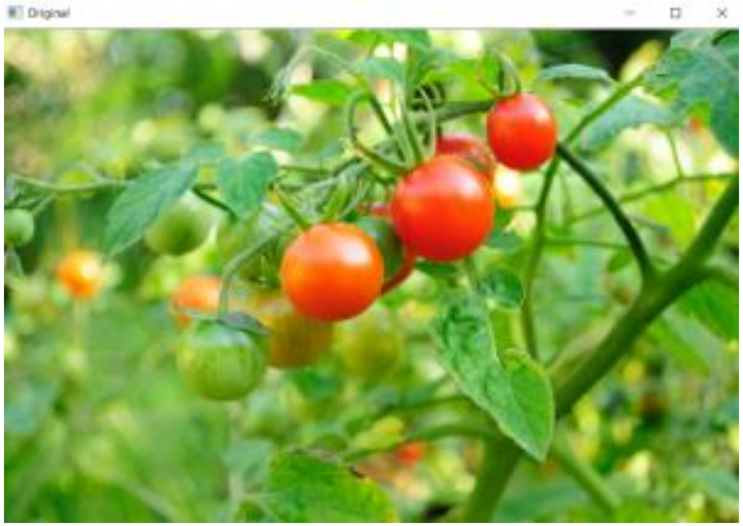
Red

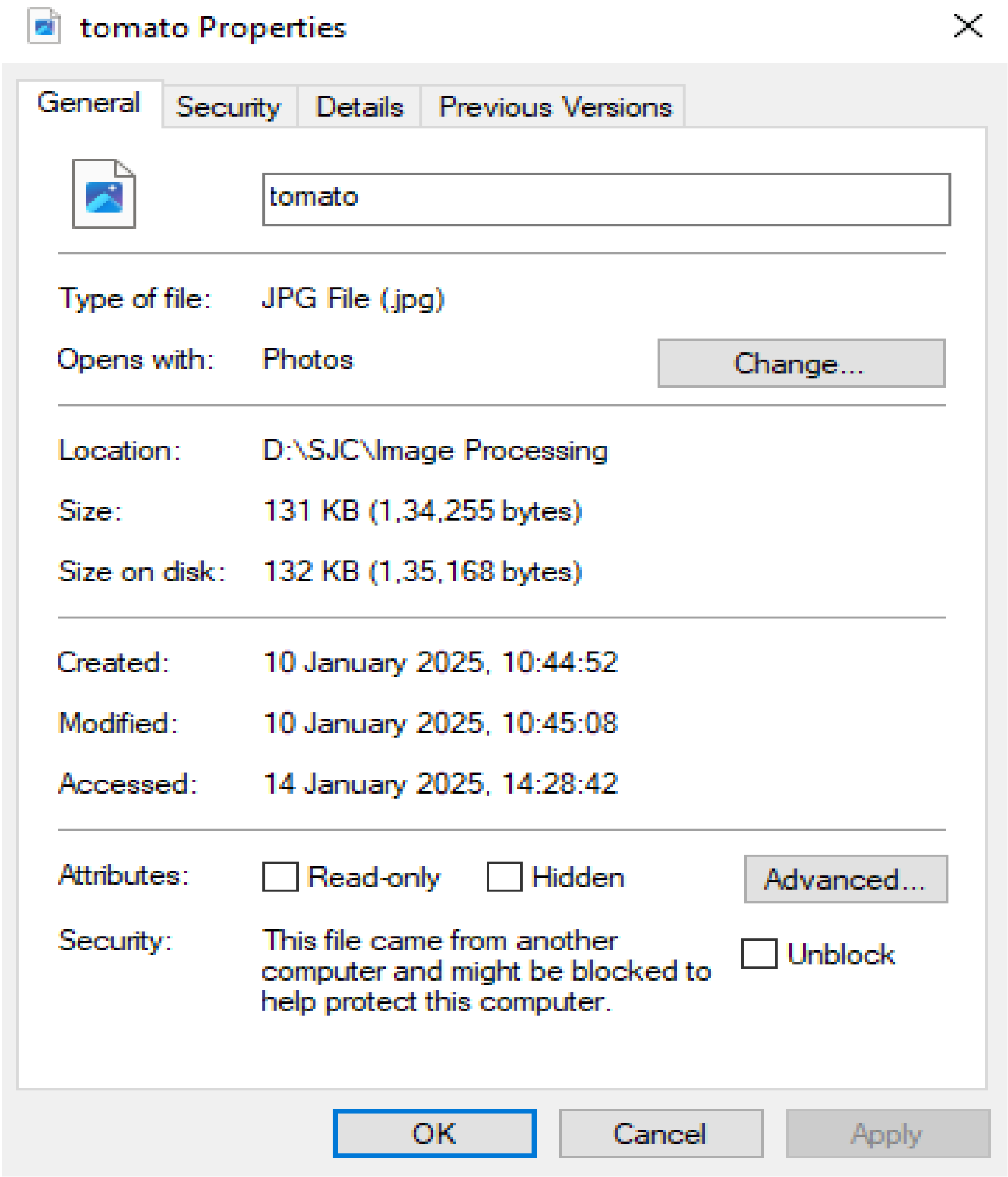
#ff0000

rgb(255, 0, 0)

hsl(0, 100%, 50%)

100%	#ffffff
95%	#ffe6e6
90%	#ffcccc
85%	#ffb3b3
80%	#ff9999
75%	#ff8080
70%	#ff6666
65%	#ff4d4d
60%	#ff3333
55%	#ff1a1a
50%	#ff0000
45%	#e60000
40%	#cc0000
35%	#b30000
30%	#990000
25%	#800000
20%	#660000
15%	#4d0000
10%	#330000
5%	#1a0000
0%	#000000




tomato Properties

tomato

Type of file: JPG File (.jpg)

Opens with: Photos

Location: D:\SJC\Image Processing

Size: 131 KB (1,34,255 bytes)

Size on disk: 132 KB (1,35,168 bytes)

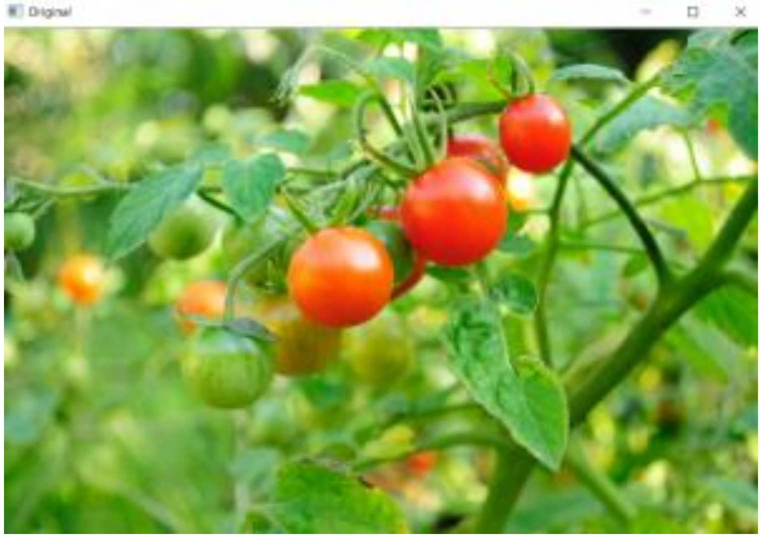
Created: 10 January 2025, 10:44:52

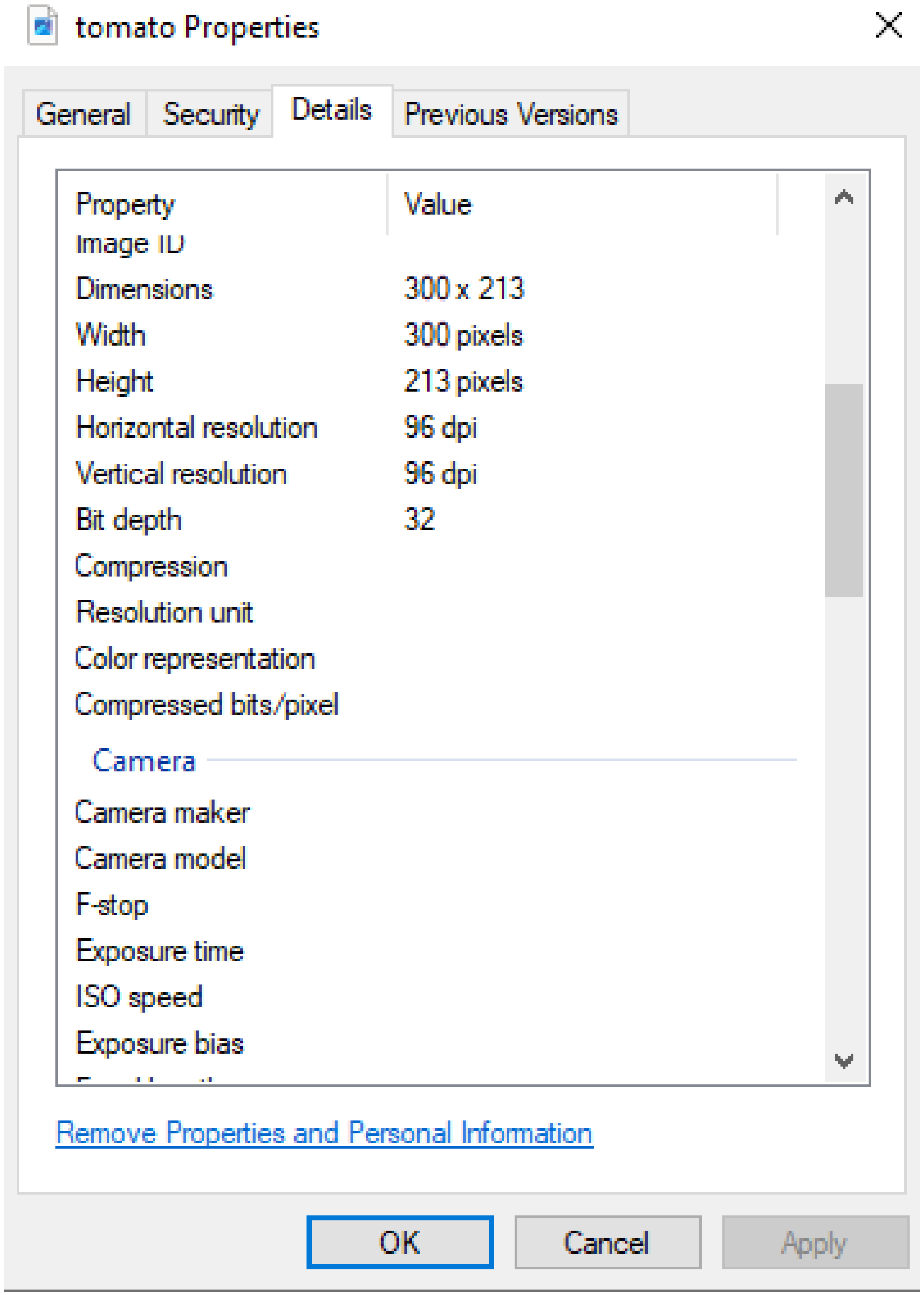
Modified: 10 January 2025, 10:45:08

Accessed: 14 January 2025, 14:28:42

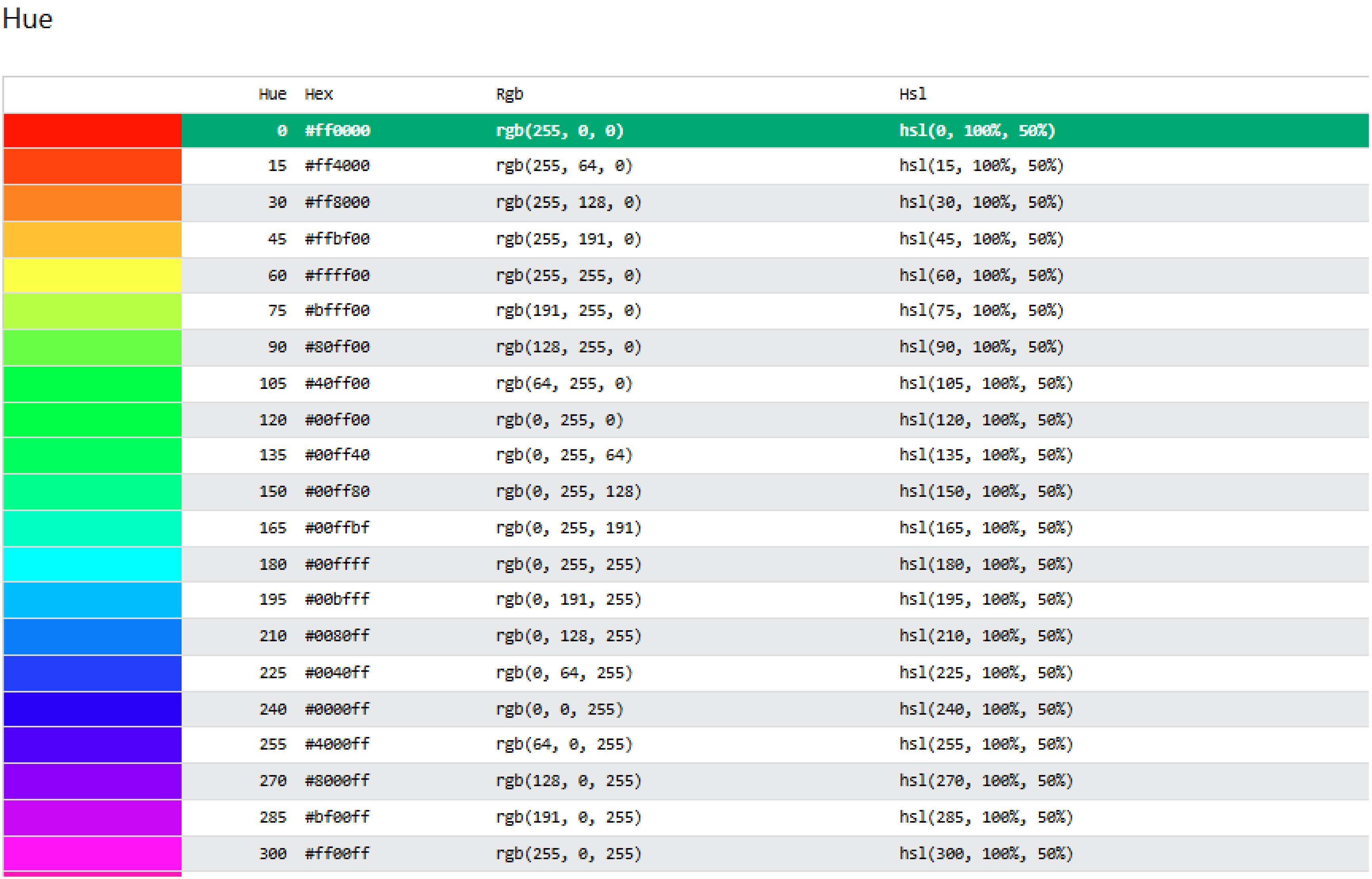
Attributes: □ Read-only □ Hidden

Security: This file came from another computer and might be blocked to help protect this computer. □ Unblock





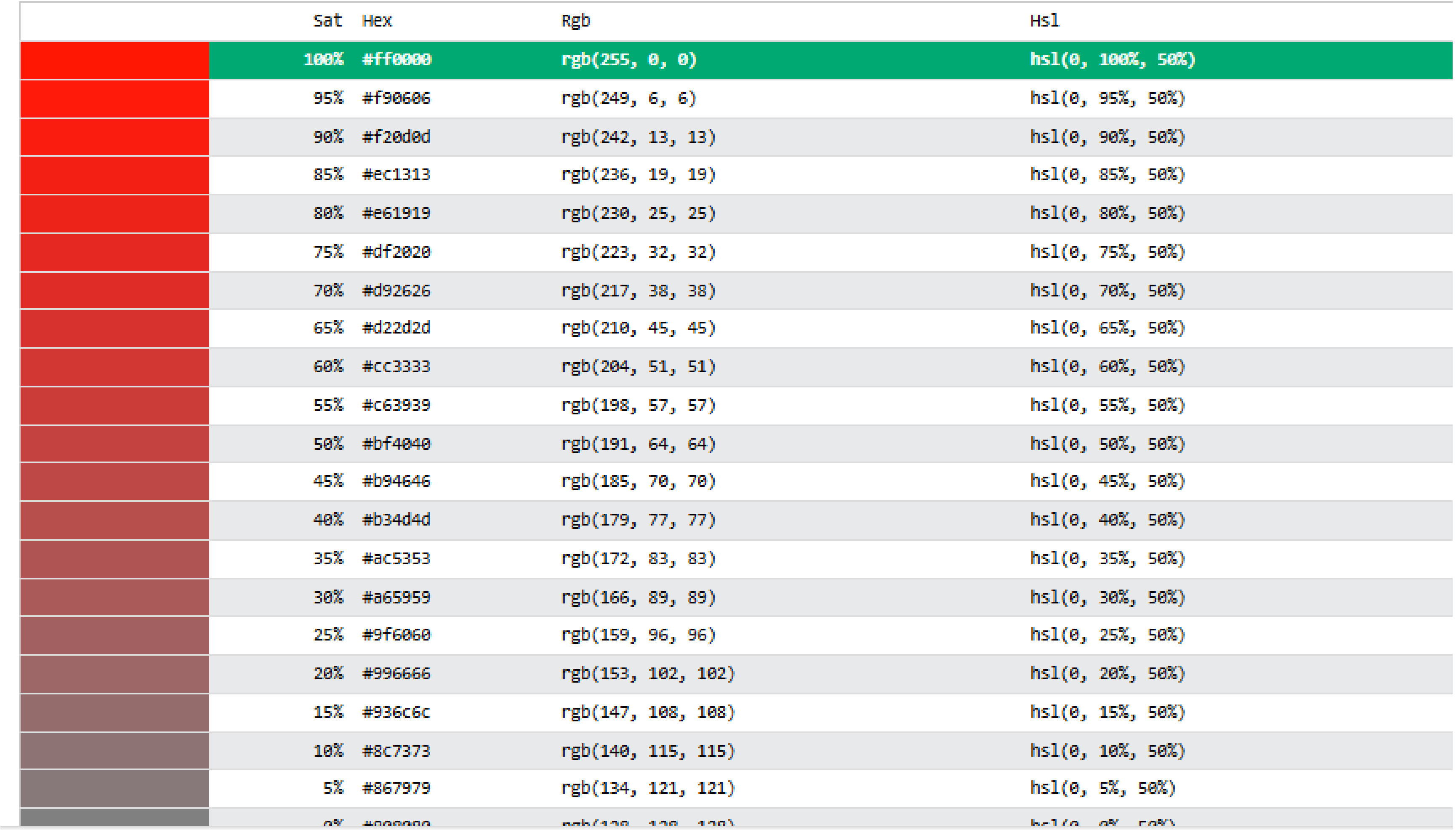
Property	Value
Image ID	
Dimensions	300 x 213
Width	300 pixels
Height	213 pixels
Horizontal resolution	96 dpi
Vertical resolution	96 dpi
Bit depth	32
Compression	
Resolution unit	
Color representation	
Compressed bits/pixel	
Camera	
Camera maker	
Camera model	
F-stop	
Exposure time	
ISO speed	
Exposure bias	



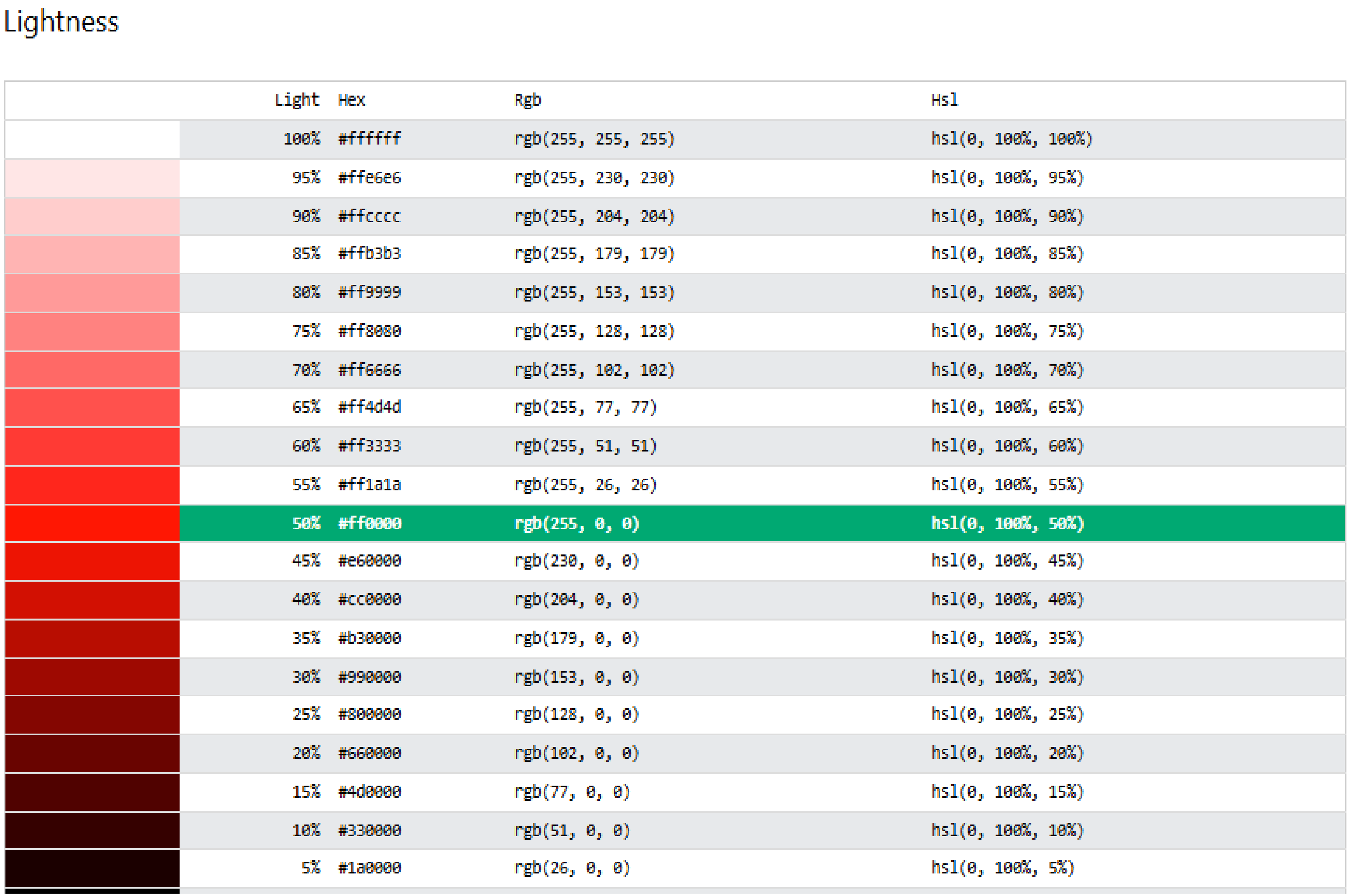
Hue

	Hue	Hex	Rgb	Hsl
🟥	0	#ff0000	rgb(255, 0, 0)	hsl(0, 100%, 50%)
🟧	15	#ff4000	rgb(255, 64, 0)	hsl(15, 100%, 50%)
🟧	30	#ff8000	rgb(255, 128, 0)	hsl(30, 100%, 50%)
🟨	45	#ffbf00	rgb(255, 191, 0)	hsl(45, 100%, 50%)
🟨	60	#ffff00	rgb(255, 255, 0)	hsl(60, 100%, 50%)
🟩	75	#bfff00	rgb(191, 255, 0)	hsl(75, 100%, 50%)
🟩	90	#80ff00	rgb(128, 255, 0)	hsl(90, 100%, 50%)
🟩	105	#40ff00	rgb(64, 255, 0)	hsl(105, 100%, 50%)
🟩	120	#00ff00	rgb(0, 255, 0)	hsl(120, 100%, 50%)
🟩	135	#00ff40	rgb(0, 255, 64)	hsl(135, 100%, 50%)
🟩	150	#00ff80	rgb(0, 255, 128)	hsl(150, 100%, 50%)
🟩	165	#00ffbf	rgb(0, 255, 191)	hsl(165, 100%, 50%)
🟦	180	#00ffff	rgb(0, 255, 255)	hsl(180, 100%, 50%)
🟦	195	#00bfff	rgb(0, 191, 255)	hsl(195, 100%, 50%)
🟦	210	#0080ff	rgb(0, 128, 255)	hsl(210, 100%, 50%)
🟦	225	#0040ff	rgb(0, 64, 255)	hsl(225, 100%, 50%)
🟦	240	#0000ff	rgb(0, 0, 255)	hsl(240, 100%, 50%)
🟪	255	#4000ff	rgb(64, 0, 255)	hsl(255, 100%, 50%)
🟪	270	#8000ff	rgb(128, 0, 255)	hsl(270, 100%, 50%)
🟪	285	#bf00ff	rgb(191, 0, 255)	hsl(285, 100%, 50%)
🟪	300	#ff00ff	rgb(255, 0, 255)	hsl(300, 100%, 50%)

Saturation



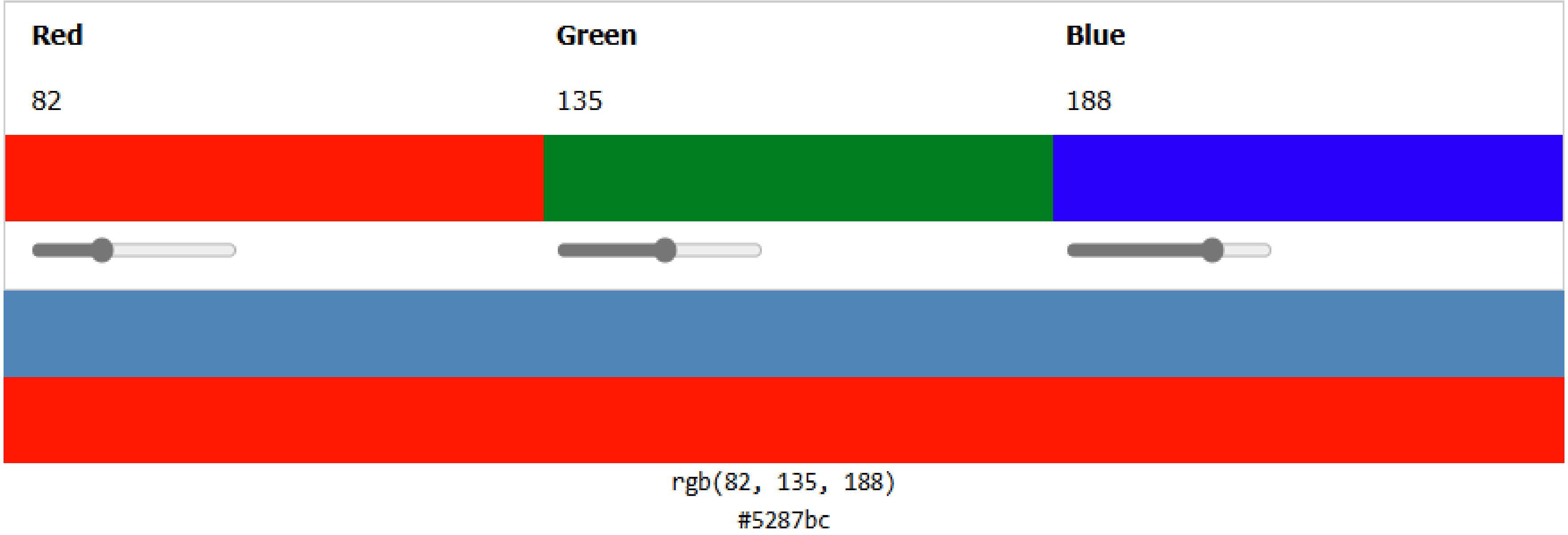
	Sat	Hex	Rgb	Hsl
🟥	100%	#ff0000	rgb(255, 0, 0)	hsl(0, 100%, 50%)
🟥	95%	#f90606	rgb(249, 6, 6)	hsl(0, 95%, 50%)
🟥	90%	#f20d0d	rgb(242, 13, 13)	hsl(0, 90%, 50%)
🟥	85%	#ec1313	rgb(236, 19, 19)	hsl(0, 85%, 50%)
🟥	80%	#e61919	rgb(230, 25, 25)	hsl(0, 80%, 50%)
🟥	75%	#df2020	rgb(223, 32, 32)	hsl(0, 75%, 50%)
🟥	70%	#d92626	rgb(217, 38, 38)	hsl(0, 70%, 50%)
🟥	65%	#d22d2d	rgb(210, 45, 45)	hsl(0, 65%, 50%)
🟥	60%	#cc3333	rgb(204, 51, 51)	hsl(0, 60%, 50%)
🟥	55%	#c63939	rgb(198, 57, 57)	hsl(0, 55%, 50%)
🟥	50%	#bf4040	rgb(191, 64, 64)	hsl(0, 50%, 50%)
🟥	45%	#b94646	rgb(185, 70, 70)	hsl(0, 45%, 50%)
🟥	40%	#b34d4d	rgb(179, 77, 77)	hsl(0, 40%, 50%)
🟥	35%	#ac5353	rgb(172, 83, 83)	hsl(0, 35%, 50%)
🟥	30%	#a65959	rgb(166, 89, 89)	hsl(0, 30%, 50%)
🟥	25%	#9f6060	rgb(159, 96, 96)	hsl(0, 25%, 50%)
🟥	20%	#996666	rgb(153, 102, 102)	hsl(0, 20%, 50%)
🟥	15%	#936c6c	rgb(147, 108, 108)	hsl(0, 15%, 50%)
🟥	10%	#8c7373	rgb(140, 115, 115)	hsl(0, 10%, 50%)
🟥	5%	#867979	rgb(134, 121, 121)	hsl(0, 5%, 50%)
⬛	0%	#808080	rgb(128, 128, 128)	hsl(0, 0%, 50%)



Lightness

	Light	Hex	Rgb	Hsl
⬜	100%	#ffffff	rgb(255, 255, 255)	hsl(0, 100%, 100%)
🟥	95%	#ffe6e6	rgb(255, 230, 230)	hsl(0, 100%, 95%)
🟥	90%	#ffcccc	rgb(255, 204, 204)	hsl(0, 100%, 90%)
🟥	85%	#ffb3b3	rgb(255, 179, 179)	hsl(0, 100%, 85%)
🟥	80%	#ff9999	rgb(255, 153, 153)	hsl(0, 100%, 80%)
🟥	75%	#ff8080	rgb(255, 128, 128)	hsl(0, 100%, 75%)
🟥	70%	#ff6666	rgb(255, 102, 102)	hsl(0, 100%, 70%)
🟥	65%	#ff4d4d	rgb(255, 77, 77)	hsl(0, 100%, 65%)
🟥	60%	#ff3333	rgb(255, 51, 51)	hsl(0, 100%, 60%)
🟥	55%	#ff1a1a	rgb(255, 26, 26)	hsl(0, 100%, 55%)
🟥	**50%**	**#ff0000**	**rgb(255, 0, 0)**	**hsl(0, 100%, 50%)**
🟥	45%	#e60000	rgb(230, 0, 0)	hsl(0, 100%, 45%)
🟥	40%	#cc0000	rgb(204, 0, 0)	hsl(0, 100%, 40%)
🟥	35%	#b30000	rgb(179, 0, 0)	hsl(0, 100%, 35%)
🟥	30%	#990000	rgb(153, 0, 0)	hsl(0, 100%, 30%)
🟥	25%	#800000	rgb(128, 0, 0)	hsl(0, 100%, 25%)
🟥	20%	#660000	rgb(102, 0, 0)	hsl(0, 100%, 20%)
🟥	15%	#4d0000	rgb(77, 0, 0)	hsl(0, 100%, 15%)
🟥	10%	#330000	rgb(51, 0, 0)	hsl(0, 100%, 10%)
🟥	5%	#1a0000	rgb(26, 0, 0)	hsl(0, 100%, 5%)

RGB (Red, Green, Blue)



Red	Green	Blue
82	135	188
🟥	🟩	🟦

rgb(82, 135, 188)

#5287bc

➢Regions and Boundaries:

- A subset R of pixels in an image is called a Regionof the image if R is a connected set.
- The boundaryof the region R is the set of pixels in the region that have one or more neighbors that are not in R.
- If R happens to be entire Image?

➢Distance measures:

Given pixels p, q and z with coordinates (x, y), (s, t), (u, v)respectively, the distance function D has following properties:a.

Aspect ratio

- In image processing, aspect ratio refers to the proportional relationship between the width and height of an image.
- It is an important parameter that affects the visual appearance and suitability of an image for different applications, such as printing, display, and analysis.

Arithmetic Operation Between Images

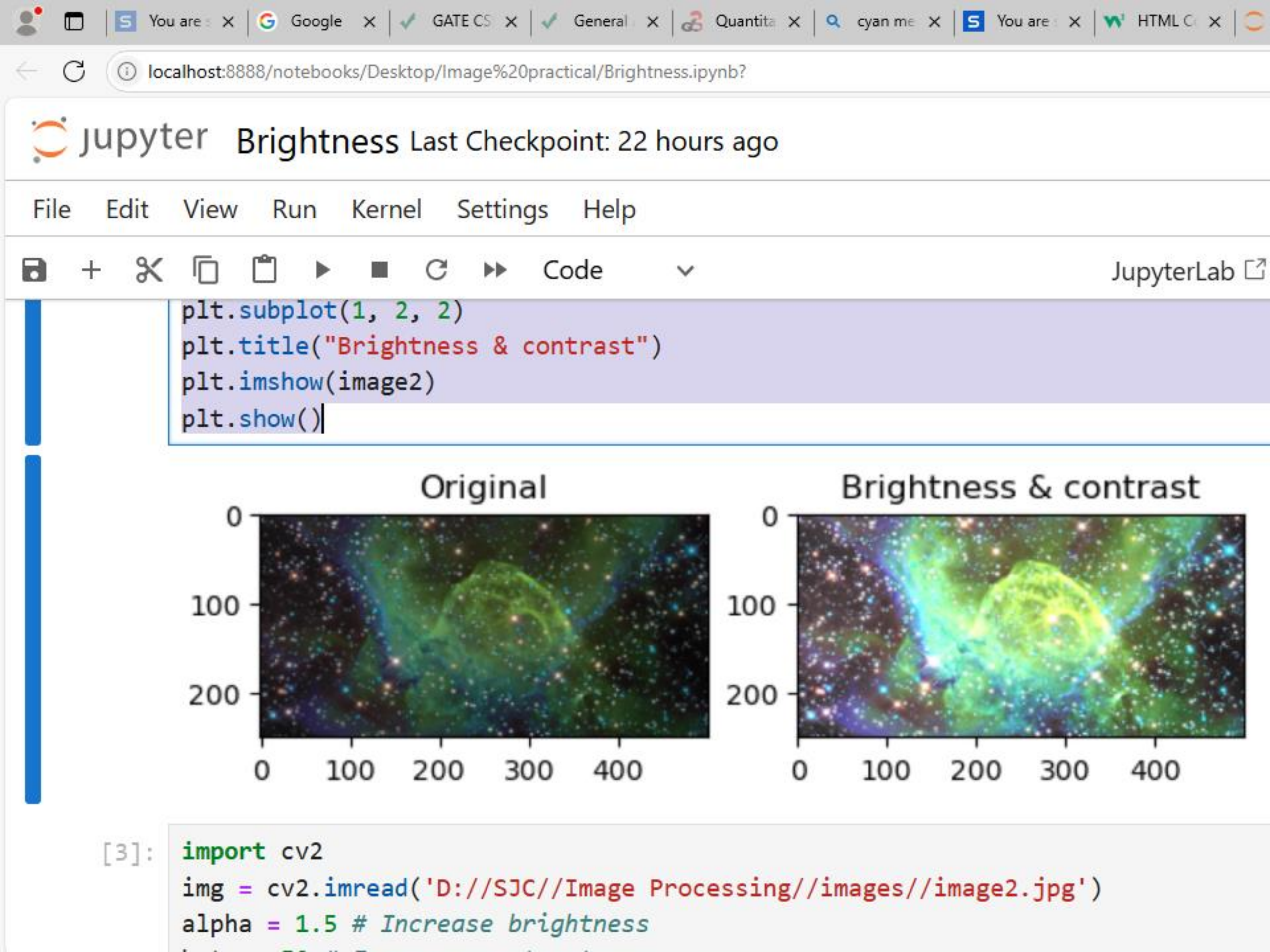
- There are Array Operations which are carried out between corresponding pixels pairs. The four arithmetic operations are denoted as

- $A(x,y) = f(x,y)+g(x,y)$
- $S(x,y) = f(x,y)-g(x,y)$
- $p(x,y) = f(x,y)*g(x,y)$
- $D(x,y) = f(x,y)/g(x,y)$

- These all arithmetic operations are performed between corresponding pixels pairs.

Important Points

- If the result is a floating point number, round off its value
- If the result is above the pixel range, select the max range value
- If the result is below the pixel range, select the min range value
- If the result is infinity, write it as zero

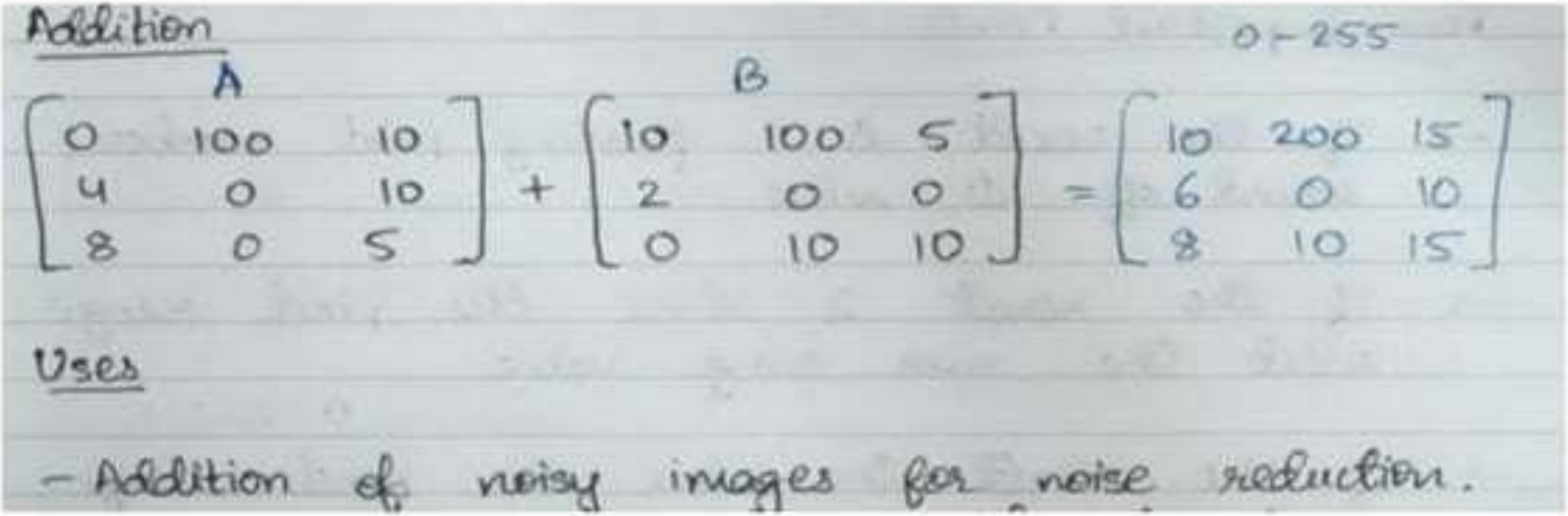


```
plt.subplot(1, 2, 2)
plt.title("Brightness & contrast")
plt.imshow(image2)
plt.show()
```

[3]:

```
import cv2
img = cv2.imread('D://SJC//Image Processing//images//image2.jpg')
alpha = 1.5 # Increase brightness
```

Addition



Addition

0 - 255

$$\overset{A}{\begin{bmatrix} 0 & 100 & 10 \\ 4 & 0 & 10 \\ 8 & 0 & 5 \end{bmatrix}} + \overset{B}{\begin{bmatrix} 10 & 100 & 5 \\ 2 & 0 & 0 \\ 0 & 10 & 10 \end{bmatrix}} = \begin{bmatrix} 10 & 200 & 15 \\ 6 & 0 & 10 \\ 8 & 10 & 15 \end{bmatrix}$$

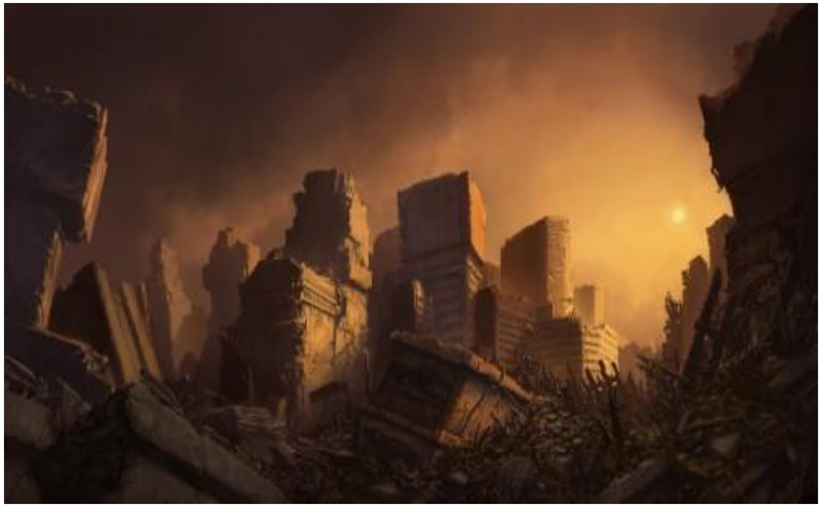
Uses

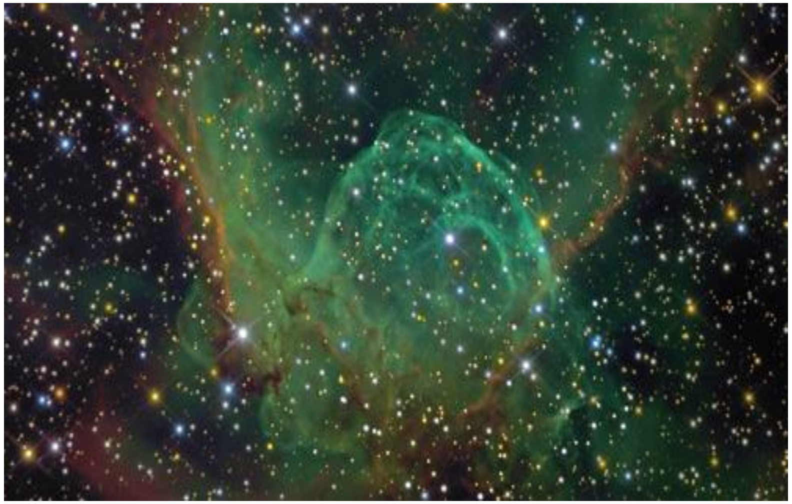
- Addition of noisy images for noise reduction.

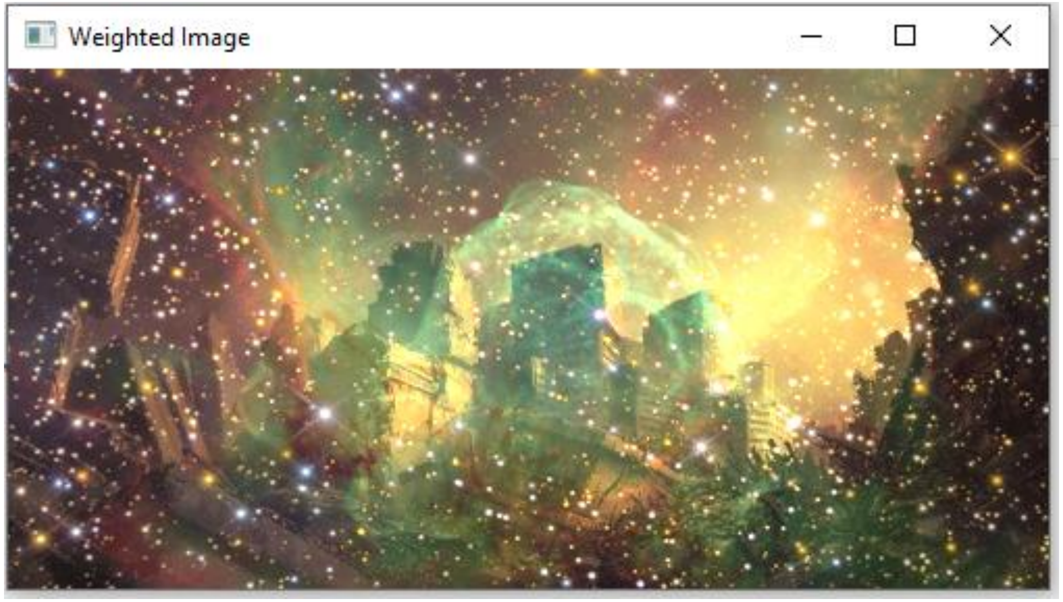
Uses:

- Changing Image Background
- Watermark Images

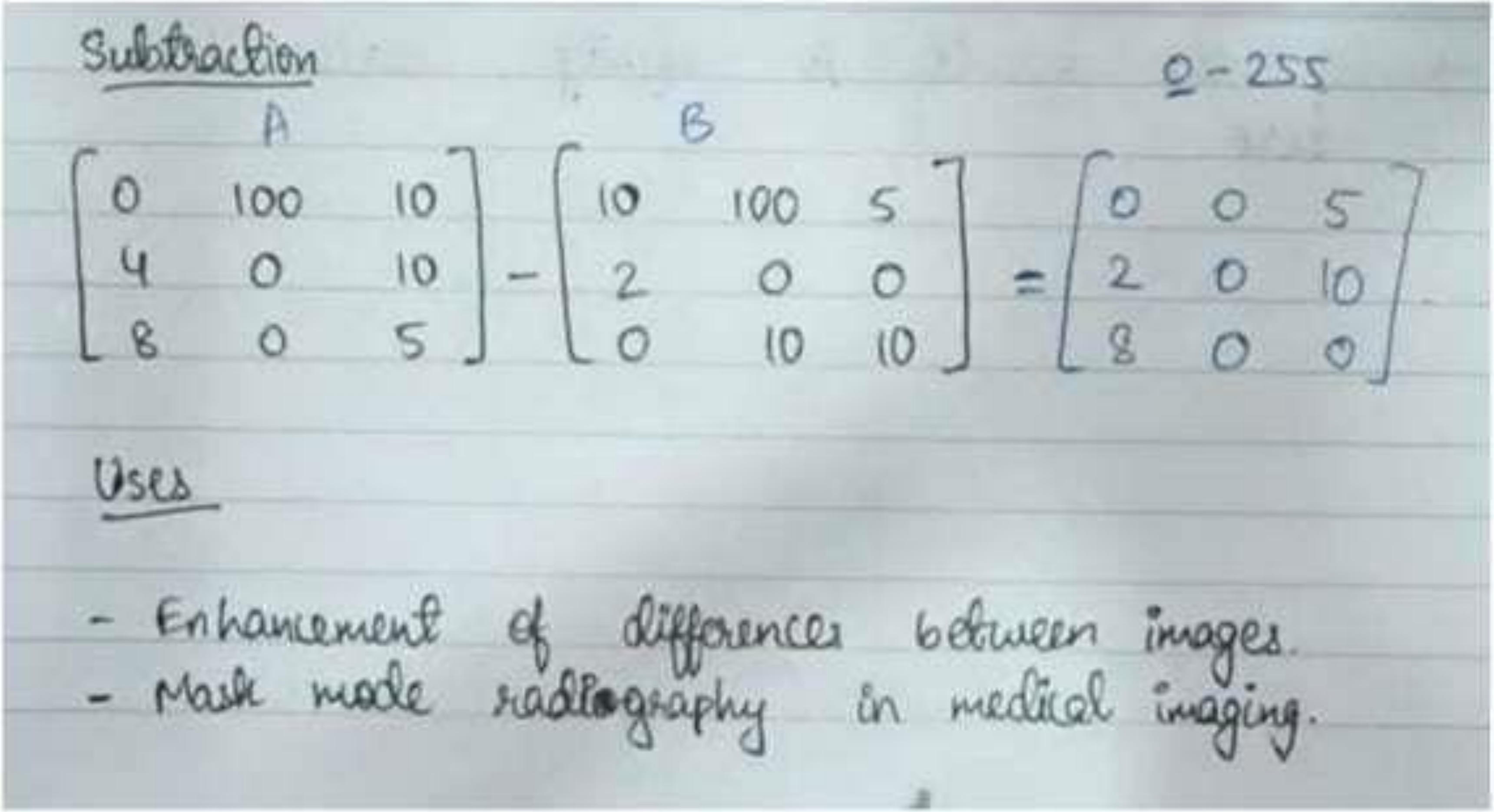
- import cv2
- first_img = cv2.imread("D://SJC//Image Processing//images//iamge1.jpg")
- second_img = cv2.imread("D://SJC//Image Processing//images//image2.jpg")
- weightedSum=cv2.add(first_img , second_img)
- cv2.imshow('Weighted Image', weightedSum)
  cv2.waitKey(0)cv2.destroyAllWindows()






Weighted Image

Subtraction



Subtraction

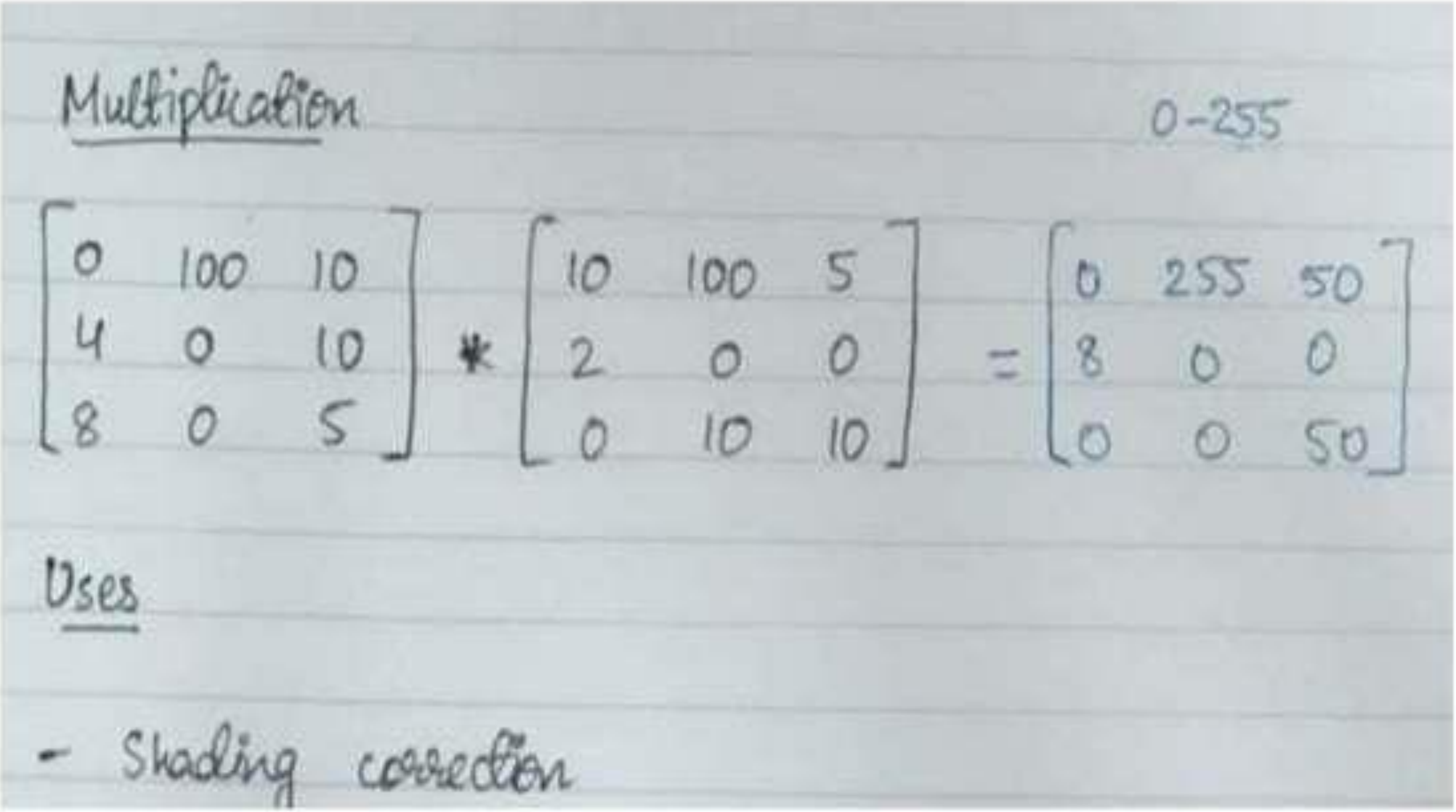
$\underline{0}$ - 255

$$\overset{A}{\begin{bmatrix}0 & 100 & 10\\4 & 0 & 10\\8 & 0 & 5\end{bmatrix}} - \overset{B}{\begin{bmatrix}10 & 100 & 5\\2 & 0 & 0\\0 & 10 & 10\end{bmatrix}} = \begin{bmatrix}0 & 0 & 5\\2 & 0 & 10\\8 & 0 & 0\end{bmatrix}$$

Uses

- Enhancement of differences between images.
- Mask mode radiography in medical imaging.

Multiplication



Multiplication

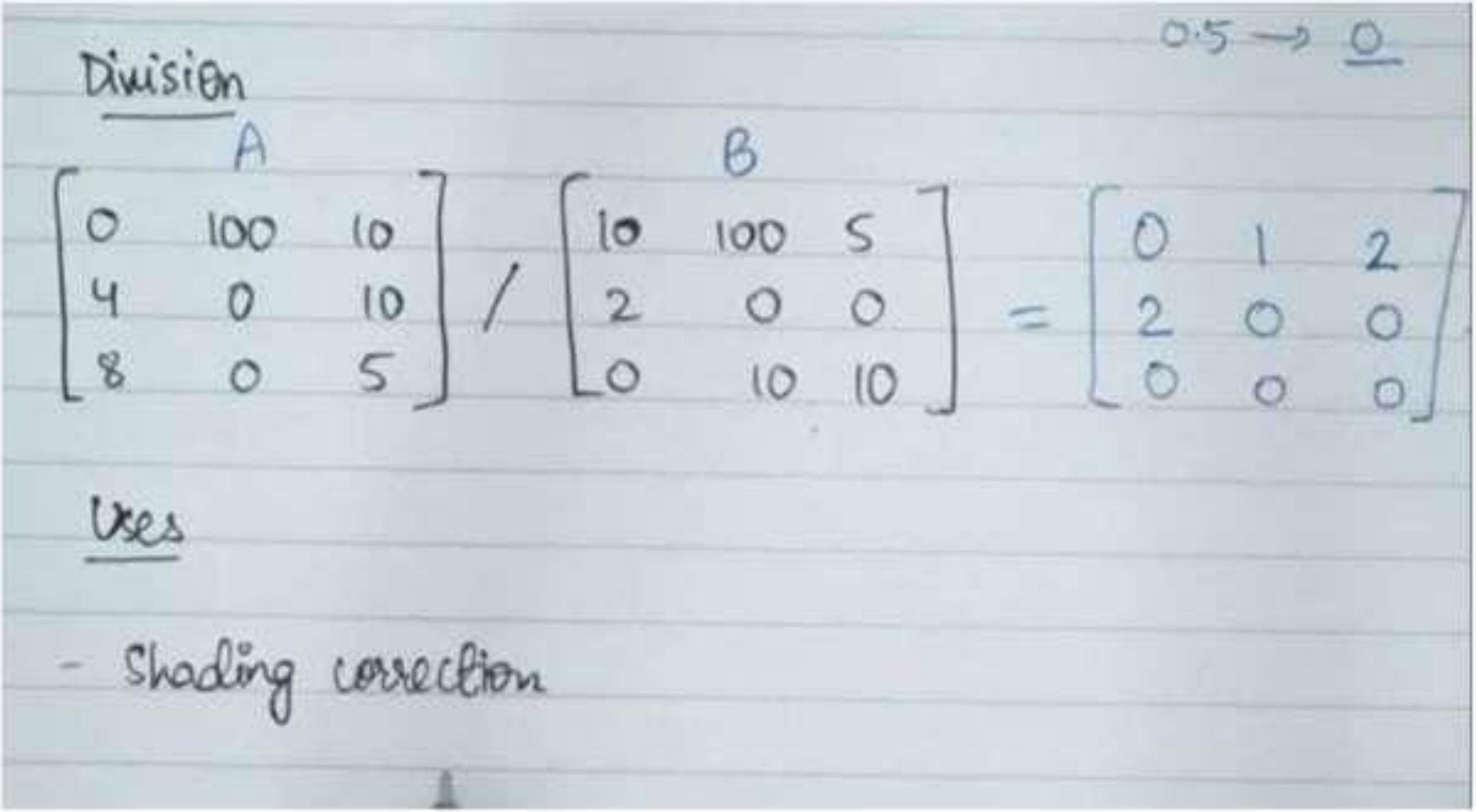
0-255

$$\begin{bmatrix} 0 & 100 & 10 \\ 4 & 0 & 10 \\ 8 & 0 & 5 \end{bmatrix} * \begin{bmatrix} 10 & 100 & 5 \\ 2 & 0 & 0 \\ 0 & 10 & 10 \end{bmatrix} = \begin{bmatrix} 0 & 255 & 50 \\ 8 & 0 & 0 \\ 0 & 0 & 50 \end{bmatrix}$$

Uses

- Shading correction

Division



$0.5 \rightarrow \underline{0}$

<u>Division</u>

$$\overset{A}{\begin{bmatrix} 0 & 100 & 10 \\ 4 & 0 & 10 \\ 8 & 0 & 5 \end{bmatrix}} / \overset{B}{\begin{bmatrix} 10 & 100 & 5 \\ 2 & 0 & 0 \\ 0 & 10 & 10 \end{bmatrix}} = \begin{bmatrix} 0 & 1 & 2 \\ 2 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

<u>Uses</u>

- Shading correction