Week 7 - Wednesday

COMP 1800

Last time

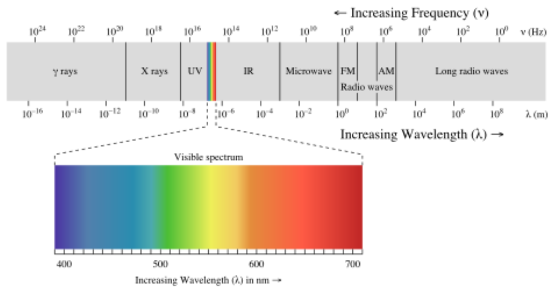
- What did we talk about last time?
- **`while`** examples
- List comprehensions
- Reading data from the Internet

Questions?

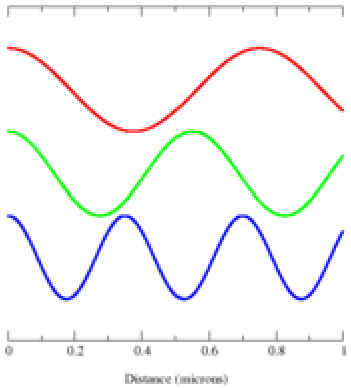
Assignment 5

Color

Light

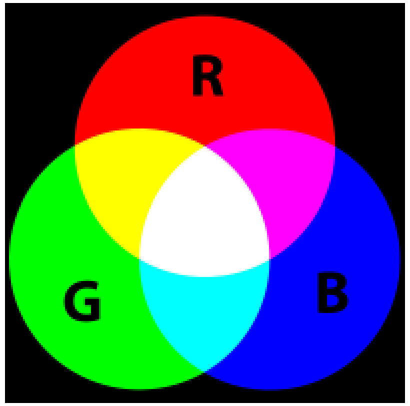


- Visible light is a form of electromagnetic radiation
- We could think of it as a wave with a specific frequency
- Instead, color theorists have discovered that we can represent most visible colors as a combination of a small number of set colors



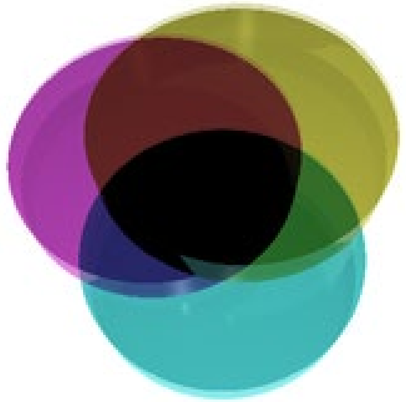
RGB

- One system for representing color is **RGB**
- With **Red**, **Green**, and **Blue** components, you can combine them to make most visible colors
- Combining colors is an additive process:
  - With no colors, the background is black
  - Adding colors never makes a darker color
  - Pure **Red** added to pure **Green** added to pure **Blue** makes **White**
- **RGB** is a good model for computer screens



CMYK

- **CMYK** stands for **Cyan**, **Magenta**, **Yellow**, and **Key** (Black)
- **CMYK** is another color system, but it's a **subtractive** system
  - With no colors, the background is white
  - Adding colors never makes a lighter color
  - Pure **Cyan** added to pure **Magenta** added to pure **Yellow** makes **Black**
- **CMYK** is useful for printing, not for computer screens



Installing necessary libraries

- The image manipulation we want to do with Python requires libraries to be installed
- Open the command line appropriate for your OS and type:

```
> pip install pillow
> pip install cImage
>
```

- The **>** isn't something you type, it's just a way to show it's on the command line
- On your own machine, you will only need to do this once
- On lab computers, you might have to do it every time you log in

Differences from the book

- The newest version of **cImage** (installed from the command line) uses the namespace **image** instead of **cImage**
- Everywhere the book says **cImage**, say **image** instead
- There are a few other small differences, notably the title of the window
- Book version:

```python
from cImage import *
window = ImageWin('Window Name', 800, 600)
```

- Our version:

```python
from image import *
window = ImageWin(800, 600, 'Window Name')
```

Pixels

- All computer images are made up of **pixels**
    - Short for **picture elements**
- Each pixel is a single color
- The smaller the pixels, the more realistic the image

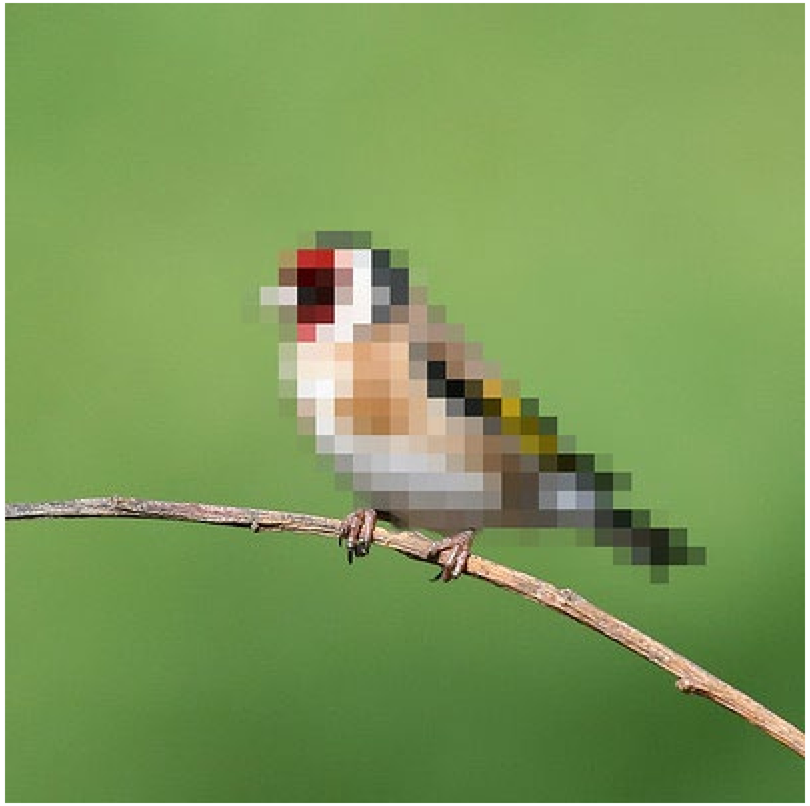


Image by Rego Korosi
https://www.flickr.com/photos/korosirego/4592913123/

`Pixel` class

- The **`Pixel`** class is a way for Python to keep track of colors, using an **RGB** model
- This class is one of many we will be using from the **`image`** module
- To use this library, you need to type:

  `from image import *`
- Each **`Pixel`** object represents one of 16,777,216 different colors with a value between 0-255 for **Red**, **Green**, and **Blue**

Example colors

Color	Red	Green	Blue
Black	0	0	0
Red	255	0	0
Green	0	255	0
Blue	0	0	255
Orange	255	165	0
Gray	128	128	128
Cyan	0	255	255
Magenta	255	0	255
Yellow	255	255	0
White	255	255	255

To use Pixel

- To create a custom color:

```
color = Pixel(255,165,0) # orange
green = color.getGreen()
```

- Create colors using **`Pixel`** to specify RGB values
- Get individual values using:
  - `getRed()`
  - `getGreen()`
  - `getBlue()`

Luminance

- If the R, G, B values happen to be the same, the color is a shade of gray
  - 255, 255, 255 = White
  - 128, 128, 128 = Gray
  - 0, 0, 0 = Black
- To convert a color to a shade of gray, use the following formula:
  - value = .3R + .59G + .11B
  - Then, the color will be (value, value, value)
- Based on the way the human eye perceives colors as light intensities

Image Class

Purpose of the `Image` class

- Because images have complex file types, some educators wrote the **`Image`** class
- It's a simple interface for doing routine things with an image
  - Loading/saving an image
  - Getting the height and width of an image
  - Changing the pixels of an image
  - Drawing the image

Image methods

Method	Use
FileImage(file)	Creates an **Image** object from a file name
EmptyImage(width, height)	Creates a blank **Image** of size **width** by **height**
getWidth()	Return the width of the image
getHeight()	Return the height of the image
getPixel(x, y)	Return the **Pixel** which is the color at (**x**,**y**)
setPixel(x, y, pixel)	Set the **Pixel** object at (**x**,**y**) to **pixel**
save(file)	Save the **Image** to the file with the given file name

Drawing an Image

- To see an image, we have to make a window and then draw the image on it
- We make a window with following constructor:

```
ImageWin(width, height, title)
```

  - The title variable is a string giving the name of the window
  - The width and height variables are integers giving the width and the height of the new window in pixels
  - This is the one where the title is in a different place than the book example

Drawing an Image example

- The following creates an **`Image`** object from a file called **`picture.jpg`**
- Then, we create an **`ImageWin`** window object to display it
- The **`Image`** object draws itself on the window
- To make the window easier to work with, we call its **`exitOnClick()`** method so that it closes when we click on it

```
picture = FileImage('picture.jpg')
window = ImageWin(picture.getWidth(),picture.getHeight(),
     'Picture')
picture.draw(window)
window.exitOnClick()
```

Nested loops

- We can put loops inside of other loops
- Doing so is useful when we want to perform a repeated task as part of another repeated task
- Example:
    - Loop over every column in an image
        - For each column, loop over every row
- Code:

```
for x in range(picture.getWidth()):
     for y in range(picture.getHeight()):
          # do something
```

Photo negative

- We can make the negative of a photo
- Algorithm:
  - Loop over every column of the image
    - Loop over every row of the column
      - Make a new pixel whose red, green, and blue are 255 – red, 255 - green, and 255 – blue
      - Put the pixel into the image in the same location

Photo negative code

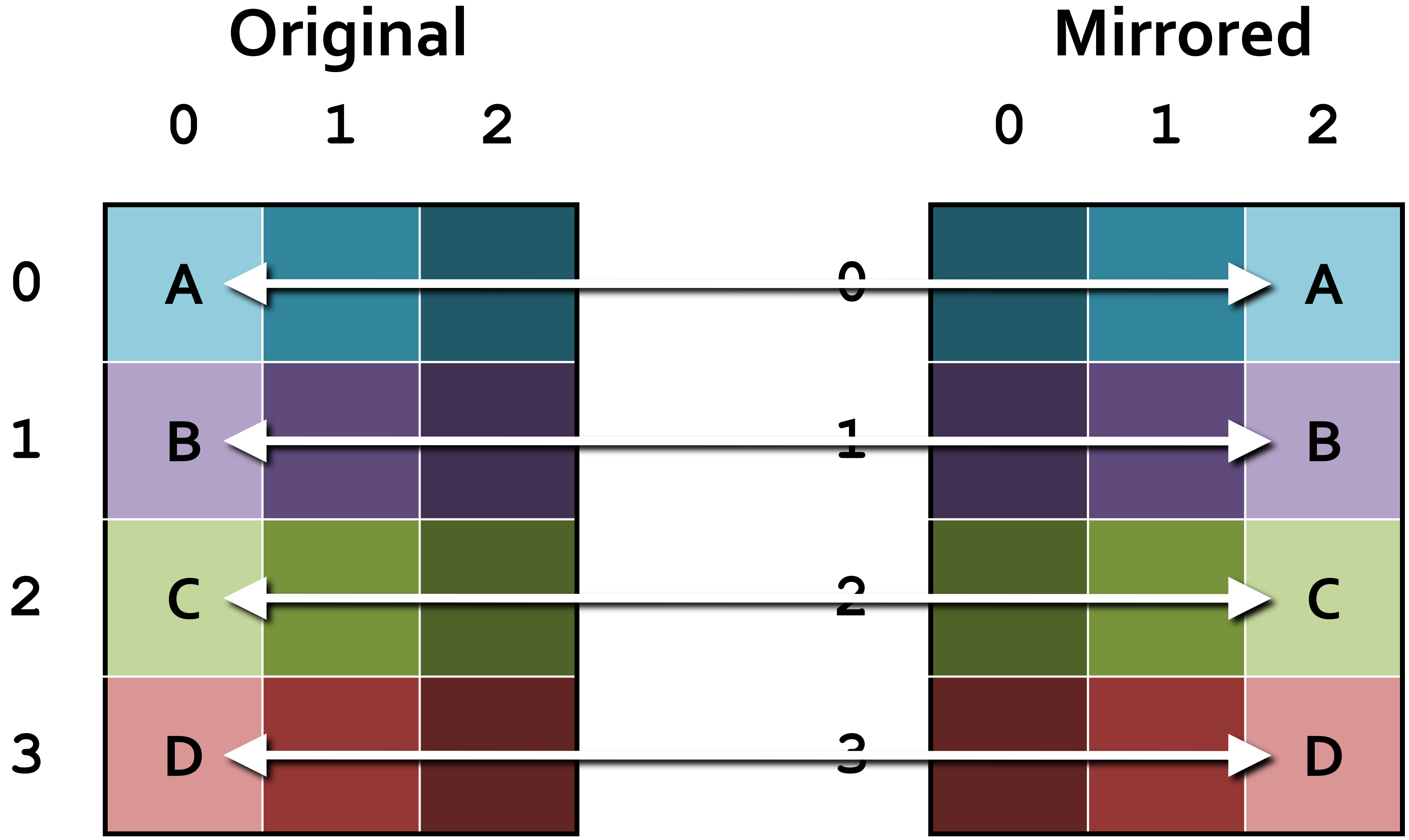
- Here's the code for the algorithm from the previous slide

```
for x in range(picture.getWidth()):
     for y in range(picture.getHeight()):
          pixel = picture.getPixel(x, y)
          red = 255 - pixel.getRed()
          green = 255 - pixel.getGreen()
          blue = 255 - pixel.getBlue()
          picture.setPixel(x, y, Pixel(red, green, blue))
```

Horizontal mirror

- We could make a mirror image of an image, flipping the left and right sides
  - Like how you look in a Zoom call … or a mirror
- Moving from left to right in the original image, copy each column, storing each column from right to left in the new image

Horizontal mirror example



Original

	0	1	2
0	A		
1	B		
2	C		
3	D		

Mirrored

	0	1	2
0			A
1			B
2			C
3			D

Horizontal mirror in code

- What would the code for mirroring look like?

```
# the picture to be mirrored
picture = FileImage(file)

mirrored =
  EmptyImage(picture.getWidth(),picture.getHeight() )

for x in range(picture.getWidth()):
  for y in range(picture.getHeight()):
    mirrored.setPixel(picture.getWidth() - x - 1, y,
  picture.getPixel(x, y))
```

Quiz

Upcoming

Next time…

- Namespaces
- Work time for Assignment 5

Reminders

- Read section 6.4
- **Finish Assignment 5**
  - **Due Friday before midnight!**