Compsci 101
Images, Tuples

Yesenia Velasco
Susan Rodger
March 21, 2023
P is for …

• Python
  • Whatever you want it to be? Language!!!
• Parameter
  • When an argument becomes a variable
• Power Cycle
  • Not the last resort. But works
• P2P
  • From networking to collaboration
Cynthia Rudin

- Duke CompSci Professor
  - Univ Buffalo, BS Mathematical Physics, BA Music Theory
  - Princeton, PhD.
- Works in interpretable machine learning, which is crucial for responsible and trustworthy AI
- Winner of Squirrel AI Award for AI for the Benefit of Humanity – 1 million
  - Detecting crime series
  - Con Edison NYC – underground electrical distribution network

She uses AI’s power to help society.
Announcements

• Assign 4 due Thursday, March 23
  • Assign 4 Sakai Quiz due tonight!
• Prelab 8 – do before lab this week
• Assign 5 and Apt 5 out on Thursday

• Exam 2 coming back soon
• APT Quiz 2 starts at end of next week
  • March 30-April 3
PFTD

• Exam 2
• Images
• Classes and Objects
• Tuples sprinkled about
Images

What is Photoshop?
Image Processing

• Convert image into format for manipulating the image
  • Visualization, Sharpening, Restoration, Recognition, Measurement, more
  • Resizing, Red-eye Removal, more
  • CrashCourse: Navigating Digital Info
Image Library

- **PIL: Python Image Library -> Pillow**
  - To install run the command below in a terminal
    - Terminal in PyCharm, not “Python Console”
    - `pip install Pillow`
      - If that doesn’t work try:
        - `Python3 -m pip install Pillow`

- Library has extensive API, far more than we need
  - Concepts often apply to every image library
  - Realized in Python-specific code/functions
Color Models

• Cameras, Displays, Phones, JumboTron: RGB
  • Additive Color Model: Red, Green, Blue
  • https://en.wikipedia.org/wiki/RGB_color_model

• Contrast Printers and Print which use CMYK
  • Subtractive: Cyan, Magenta, Yellow, Key/Black
An image is made up of Pixels

- A pixel is a square of color
Images and Pixels

- **Image is a collection of pixels**
  - Organized in rows: number of rows is image height
  - Each row has the same length: image width

- **Pixels addressed by \((x, y)\) coordinates**
  - Upper-left \((0,0)\), Lower-right \((\text{width}-1, \text{height}-1)\)
  - Typically is a single \((x, y)\) entity: tuple

- **Remember: Tuple is immutable, indexed sequence**
  \((a, b, c)\)
Each pixel has a location in Image
Each pixel has an RGB color

- Duke has three Duke blues
- Duke Athletics RGB: (0, 48, 145)
- Two for academics

<table>
<thead>
<tr>
<th>BLUE (DUKE ATHLETICS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PANTONE: PMS 287 C</td>
</tr>
<tr>
<td>HEX COLOR: #003087;</td>
</tr>
<tr>
<td>RGB: (0, 48, 135)</td>
</tr>
<tr>
<td>CMYK: (100, 75, 2, 18)</td>
</tr>
<tr>
<td>BUY MATCHING PAINT</td>
</tr>
</tbody>
</table>

DUKE ROYAL BLUE
HEX COLOR: #00539B
RGB: (0, 83, 155)
CMYK: (100, 53, 2, 16)

DUKE NAVY BLUE
HEX COLOR: #012169;
RGB: (1, 33, 105)
CMYK: (100, 85, 5, 22)
SimpleDisplay.py

• Access to PIL and Image module
  • What type is img?
  • https://pillow.readthedocs.io/en/latest/

```python
from PIL import Image

if __name__ == '__main__':
    img = Image.open("images/bluedevil.png")
    img.show()
    print("type is:", type(img))
    print("width %d height %d" % (img.width, img.height))
```

OUTPUT:
String formatting with % operator

- Use formatted string with % in string to show where to put values
  - Followed by % and tuple of values
  - %d is for an int
  - %f is for a float
  - %.xf is to specify x digits past the decimal
  - %s is for a string or something that could be shown as a string
String Formatting Examples

name = "Xiao"
age = 19
print("%s is %d years old" % (name, age))
alist = [6, 7.8643, 2]
print("%f is a list %s" % (alist[1], alist))
print("fav in %s is %.2f" % (alist, alist[1]))

OUTPUT:
WOTO-1 Images
What is a class in Python?

- Class ≈ module ≈ library (for this CS101)
- Class – Also blueprint/Factory for creating objects
  - We've used int, float, str
  - `<class 'int'>`, `<class 'list'>`
  - Everything is a class in Python3

- Objects are created from a class
  - x = [5, 6, 7]
  - b = “Moe”
  - c = “Charlotte”
Types

```
print(type(6))
print(type([1,1]))
print(type('blue'))
print(type((6,[7])))

img = Image.open("images/bluedevil.png")
print(type(img))

img = Image.open("images/eastereggs.jpg")
print(type(img))
```
What is a class in Python?

• Use .dot notation to access object's innards
  • word = "Hello"
    • word is an object from the String class
  • word.lower()
    • .lower() is a function, but don’t call it that!
    • Function that goes with a class is called a method
    • .lower() is a method from the String class
  • img.width is an attribute aka field/property
    • Note there are no ()’s, like a variable
Image Library Basics

• Library can create/open images in different formats, e.g., .png, .jpg, .gif, ...

• Images have properties: width, height, type, color-model, and more (variables associated with class)
  • Functions and fields access these properties, e.g., `im.width`, `im.format`, and more

• Pixels are formed as triples \((255,255,255), (r,g,b)\)
  • In Python these are tuples: immutable sequence
Types

```python
img = Image.open("images/bluedevil.png")
print(img.format)
```

```python
img = Image.open("images/eastereggs.jpg")
print(img.format)
```
WOTO-2 Classes
Demo: Convert Color to Gray

Process each pixel
Convert to gray
```python
if __name__ == '__main__':
    img = Image.open("images/eastereggs.jpg")
    start = time.process_time()
    gray_img = grayByPixel(img, True)
    #gray_img = grayByData(img, True)
    end = time.process_time()
    img.show()
    gray_img.show()
    print("Time = %1.3f" % (end-start))
```
grayByPixel Function

def grayByPixel(img, debug=False):
    width = img.width
    height = img.height
    new_img = img.copy()
    if debug:
        print("creating %d x %d image" % (width, height))
    for x in range(width):
        for y in range(height):
            (r, g, b) = img.getpixel((x, y))
            grays = getGray(r, g, b)
            new_img.putpixel((x, y), grays)
    return new_img
getGray function

def getGray(r, g, b):
    gray = int(0.21*r + 0.71*g + 0.07*b)
    return (gray, gray, gray)
def grayByPixel(img, debug=False):
    width = img.width
    height = img.height
    new_img = img.copy()
    if debug:
        print("creating %d x %d image" % (width, height))
    for x in range(width):
        for y in range(height):
            (r, g, b) = img.getpixel((x, y))
            grays = getGray(r, g, b)
            new_img.putpixel((x, y), grays)
Revisiting nested Loops

• What is printed here? y varies first
  • Value of x as inner loop iterates?

```python
>>> for x in range(5):
    ...
    for y in range(3):
        ...
        print(x, y)
```

```
0 0
0 1
0 2
1 0
1 1
1 2
2 0
2 1
2 2
3 0
3 1
3 2
4 0
4 1
4 2
```
def grayByPixel(img, debug=False):
    width = img.width
    height = img.height
    new_img = img.copy()
    if debug:
        print("creating %d x %d image" % (width, height))
    for x in range(width):
        for y in range(height):
            (r, g, b) = img.getpixel((x, y))
            grays = getGray(r, g, b)
            new_img.putpixel((x, y), grays)
Accessing Individual Pixels is Inefficient

• Accessing each one one-at-a-time is inefficient
  • Python can do better "under the hood"

• PIL provides a function `img.getdata()`
  • Returns list-like object for accessing all pixels
  • Similar to how file is a sequence of characters
  • Symmetry: `img.putdata(sequence)`
Processing all Pixels at Once

• Treat `img.getdata()` as list, it's not quite a list
  • Iterable: object use in “for ... in ...” loop

```python
def grayByData(img, debug=False):
    pixels = [getGray(r, g, b) for (r, g, b) in img.getdata()]
    new_img = Image.new("RGB", img.size)
    new_img.putdata(pixels)
```
def grayByData(img, debug=False):
    pixels = [getGray(r,g,b) for (r,g,b) in img.getdata()]
    new_img = Image.new("RGB", img.size)
    new_img.putdata(pixels)
    if debug:
        print("created %d x %d gray image" % (img.width, img.height))
    return new_img
## Summary of Image functions

- Many, many more

<table>
<thead>
<tr>
<th>Image function/method</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>im.show()</td>
<td>Display image on screen</td>
</tr>
<tr>
<td>im.save(&quot;foo.jpg&quot;)</td>
<td>Save image with filename</td>
</tr>
<tr>
<td>im.copy()</td>
<td>Return copy of im</td>
</tr>
<tr>
<td>im.getdata()</td>
<td>Return iterable pixel sequence</td>
</tr>
<tr>
<td>im.load()</td>
<td>Return Pixel collection indexed by tuple (x,y)</td>
</tr>
</tbody>
</table>
WOTO-4 More on Images