CSE 107

Python for image processing

The Python Language

- Python is:
  - High level
  - General purpose
  - Interpreted (versus compiled)
- It has a simple syntax, emphasizing code readability

Getting Python

- (Mostly) pre-installed on macOS and Linux.
  - But might not be latest version.
- On Windows, download from python.org.
- Should be on lab computers.
- If you have a laptop, make sure you have it installed.
- Detailed introduction, such as
  - https://www.w3schools.com/python/python_intro.asp

Run your python code

- Command line:
  - Terminal
  - iTerm (for Mac OS)
  - cmd,Powershell,WSL (Windows)
- IDE:
  - IDLE
  - VS Code
  - PyCharm
  - Jupyter Notebook
  - …

Set up your conda env (Mac OS)

Mac OS

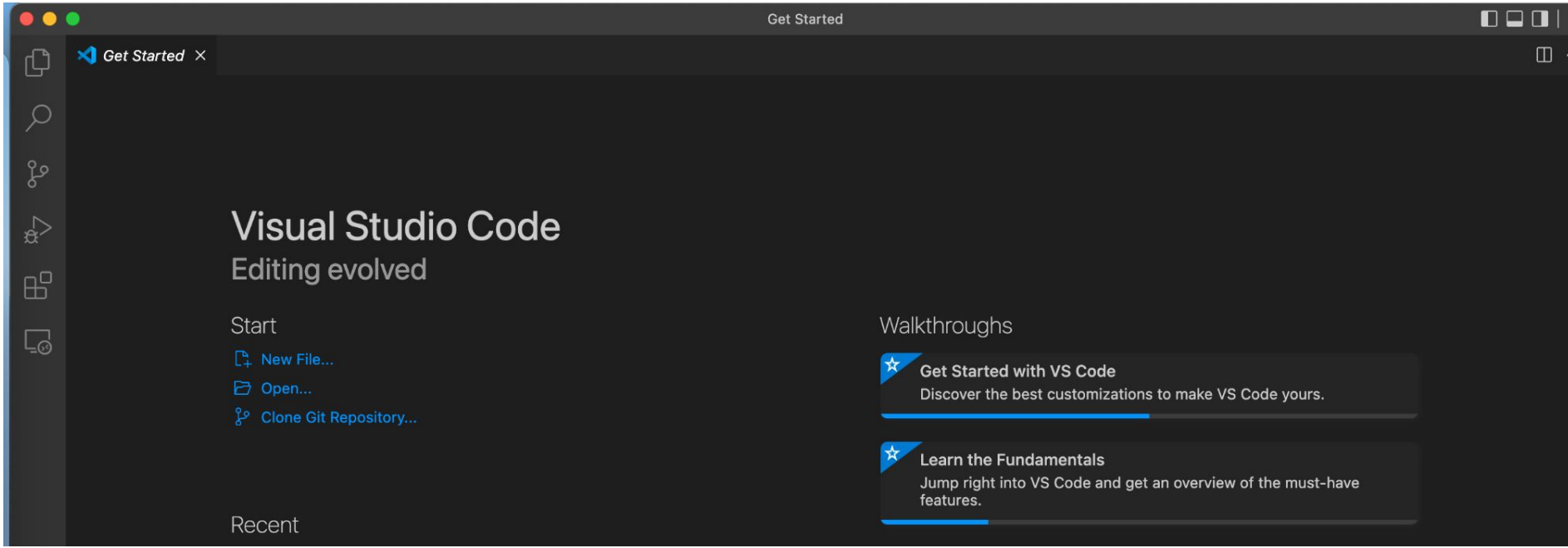
1. Download and install anaconda:
   a. Mac OS: https://docs.conda.io/projects/conda/en/latest/user-guide/install/macos.html
2. Install required packages:
   a. https://anaconda.org/conda-forge/matplotlib
3. Check your env with `conda list`

Windows

Download the installer from anaconda.com and install.

VS Code

Download: https://code.visualstudio.com/download



EXPLORER

CSE107

- intro.py
- my_gray_puppy.jpg
- my_puppy.jpg
- puppy.jpg

Create folder and files here

puppy.jpg | myfunction.m | intro.py | my_puppy.jpg | my_gray_puppy.jpg

intro.py

```python
import numpy as np
from numpy import asarray

# Create two-dimensional data structures (images) and setting their values to zero
a = np.zeros(shape=
a[0,0] = 2        #
a[1:3, 0:4] = [[0,1

# loop matrix wit
row, col = a.shape
for r in range(row)
    for c in range(
        if a[r,c] !
            print('


##################
# Matrix arithmetic
##################

# 1. dot product: I
a = np.array([[1, 2
b = np.array([[5, 6
print("dot:", np.do

# 2. trasnpose: swi
a = np.array([[1, 2
print("\nWith np.tr
print(np.transpose(

print("\nWith ndarr
print(a.transpose()

# 3. Mean: Returns
a.mean()

# 4. Return the max
a.max()

# 5. Return the min
a.min()


##################################################################
# Basic image operation
##################################################################
# import pillow
from PIL import Image, ImageOps
```

```
(base) yuxin1211@yuxin1211-mbp cse107 % pwd
/Users/yuxin1211/workspace/cse107
(base) yuxin1211@yuxin1211-mbp cse107 % ls
intro.py                my_gray_puppy.jpg       my_puppy.jpg            puppy.jpg
(base) yuxin1211@yuxin1211-mbp cse107 % python intro.py
current value is:  2.0
current value is:  1.0
current value is:  2.0
current value is:  3.0
current value is:  2.0
current value is:  3.0
current value is:  4.0
current value is:  5.0
dot: [[21 24 27]
 [47 54 61]]

With np.transpose(a) function
[[1 3 5]
 [2 4 6]]

With ndarray.transpose() method
[[1 3 5]
 [2 4 6]]
previous image size is:  (1920, 1330)
modified image size is:  (256, 256)
(256, 256, 3)
current shape is:  (256, 256)
current pixel value is greater than 200:  251
current pixel value is greater than 200:  252
current pixel value is greater than 200:  255
current pixel value is greater than 200:  251
current pixel value is greater than 200:  255
current pixel value is greater than 200:  255
current pixel value is greater than 200:  251
```

1. Open terminal
2. Go to this folder
3. Type command: python xxxxx.py

OUTLINE
PROJECTS
TABLE METADATA
UNCOMMITTED CHANGES
REMOTE HOSTS [MS]
WORKING SETS

Working with Image in Python

- Python Image Manipulation Tools You Can Try:
  - Numpy
  - PIL/ Pillow
  - *OpenCV*
  - *Scikit-Image*
  - …

Numpy

NumPy is the fundamental package needed for scientific computing with Python.

```
>>> import numpy as np
>>> a1D = np.array([1, 2, 3, 4])
>>> a2D = np.array([[1, 2], [3, 4]])
>>> a3D = np.array([[[1, 2], [3, 4]], [[5, 6], [7, 8]]])
>>>
>>> print(a1D)
[1 2 3 4]
>>> print(a2D)
[[1 2]
 [3 4]]
>>> print(a3D)
[[[1 2]
  [3 4]]

 [[5 6]
  [7 8]]]
>>>
```

```
import numpy as np
from numpy import asarray

# read image to numpy array
data = asarray(im_resize)

# print shape of your numpy array
print(data.shape)
```

Matrix operations

```
# Create two-dimensional data structures (images) and setting their values to zero
a = np.zeros(shape=(6, 6))
a[0,0] = 2        # assign value by index
a[1:3, 0:4] = [[0,1,2,3], [2,3,4,5]]  # assign value by slicing

# loop a matrix with 2 for loops
row, col = a.shape
for r in range(row):
    for c in range(col):
        if a[r,c] != 0:   # using if statement to check the value
            print('current value is: ', a[r,c])
```

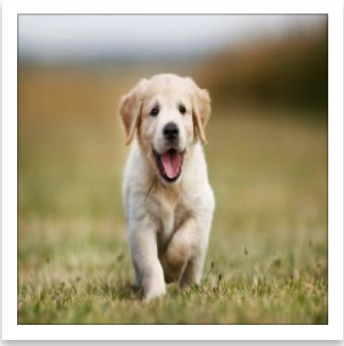
Matrix arithmetic

```
# 1. dot product: It is the sum of the products of the corresponding elements in the two matrices.
a = np.array([[1, 2], [3, 4]])
b = np.array([[5, 6, 7], [8, 9, 10]])
print("dot:", np.dot(a, b))

# 2. trasnpose: switching its rows with its columns.
a = np.array([[1, 2], [3, 4], [5, 6]])
print("\nWith np.transpose(a) function")
print(np.transpose(a))

print("\nWith ndarray.transpose() method")
print(a.transpose())

# 3. Mean: Returns the average of the matrix elements along the given axis.
a.mean()

# 4. Return the maximum value along an axis.
a.max()

# 5. Return the minimum value along an axis.
a.min()
```

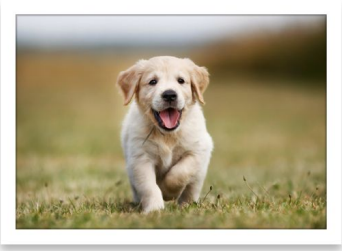
Pillow

Pillow is the friendly PIL fork. PIL is the Python Imaging Library, adds image processing capabilities to your Python interpreter.

```python
# import pillow
from PIL import Image, ImageOps

# open a image from path
im = Image.open('puppy.jpg')

# show the image
im.show()

# resize the image
new_size = (256, 256)
im_resize = im.resize(new_size)

# save the image to target path
im_resize.save('my_puppy.jpg')

# print image size
print("previous image size is: ", im.size)
print("modified image size is: ", im_resize.size)
```



my_puppy.jpg



puppy.jpg

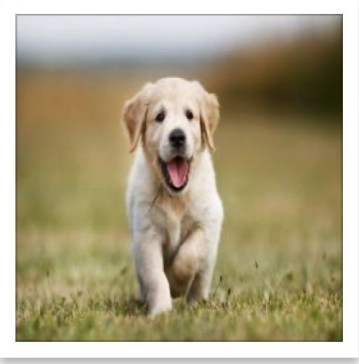
When I run this from terminal:

```
(base) yuxin1211@yuxin1211-mbp cse107 % python intro.py
previous image size is:  (1920, 1330)
modified image size is:  (256, 256)
```

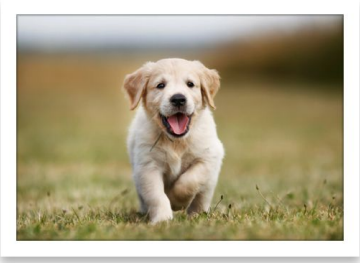
Convert RGB image to Grayscale

```
# convert image to gray scale
im_gray = ImageOps.grayscale(im_resize)
im_gray_path = 'my_gray_puppy.jpg'
im_gray.save(im_gray_path)
```

my_gray_puppy.jpg



my_puppy.jpg



puppy.jpg

Image processing with pixels

```
# loop all the pixels of the image
im_gray_pixels = asarray(Image.open(im_gray_path))
print("current shape is: ", im_gray_pixels.shape)

rows, cols = im_gray_pixels.shape

# get all the pixel values using the index
for row in range(rows):
    for col in range(cols):
        # get the current pixel value
        current_pixel_value = im_gray_pixels[row, col]
        # Manipulating your pixel values
        # for example: print pixel values that are greater than 200
        if current_pixel_value > 200:
            print("current pixel value is greater than 200: ", current_pixel_value)
```

Function

```python
def myFirstFunction(a,b):
    print(a+b)

myFirstFunction(46,64)
myFirstFunction('46','64')
```

```
(base) boris@gigapc:/mnt/c/Users/boris/Desktop/CSE107$ python intro.py
110
4664
```

Call function from another file

EXPLORER

OPEN EDITORS

CSE107

- __pycache__
- intro.py
- myfunc.py

intro.py | myfunc.py

myfunc.py > divideNumbers

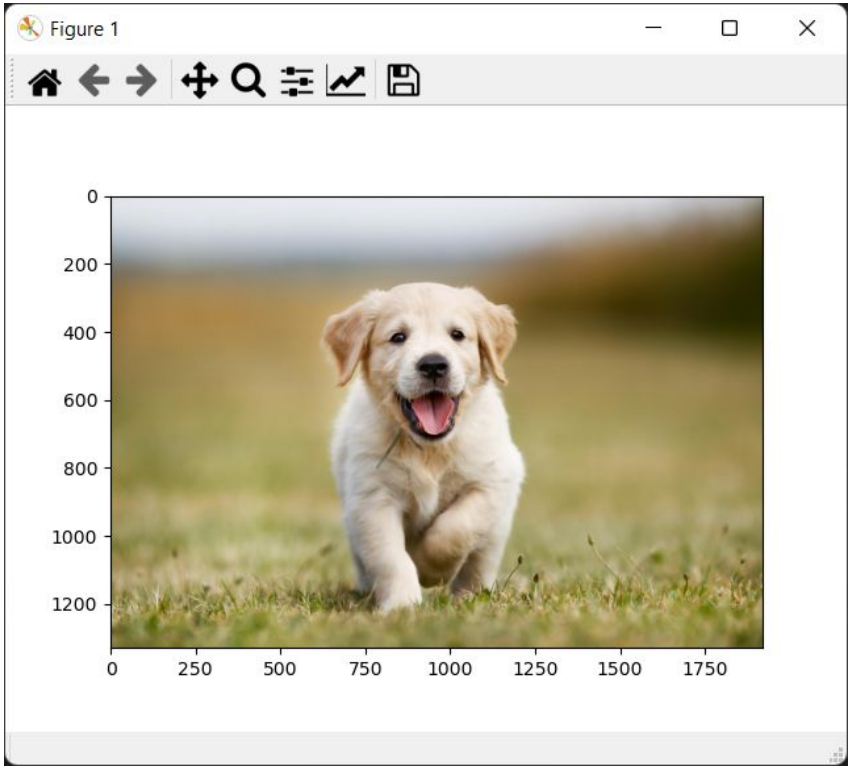
```python
# myfunc.py

def addNumbers(a, b):
    print("Sum is ", a + b)

def subtractNumbers(a, b):
    print("Difference is ", a-b)

def multiplyNumbers(a, b):
    print("Product is ", a * b)

def divideNumbers(a, b):
    print("Division is ", a / b)

def modulusNumbers(a, b):
    print("Remainder is ", a % b)
```

```python
import myfunc
from myfunc import subtractNumbers,divideNumbers

# call function
myfunc.addNumbers(456,654)
subtractNumbers(654,321)
divideNumbers(462,7984)

#cannot call multiplyNumbers directly at this time since it's not imported
#multiplyNumbers(546,7)
myfunc.multiplyNumbers(546,7)
```

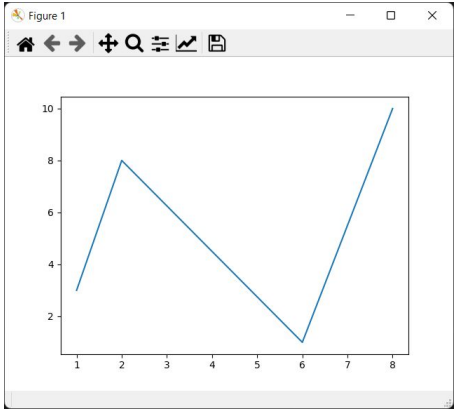
```
(base) boris@gigapc:/mnt/c/Users/boris/Desktop/CSE107$ python intro.py
Sum is  1110
Difference is  333
Division is  0.057865731462925854
Product is  3822
```

Plotting images using Matplotlib

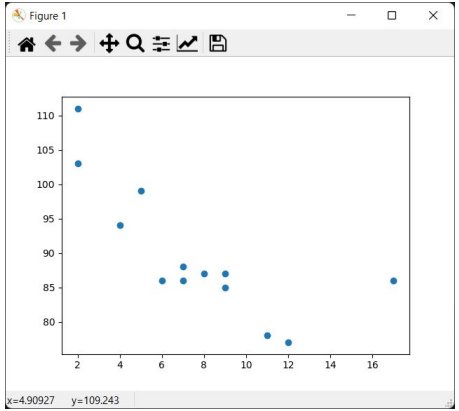
```
#display image using plt.imshow
im = Image.open('puppy.jpg')
plt.imshow(im)
plt.show()
```



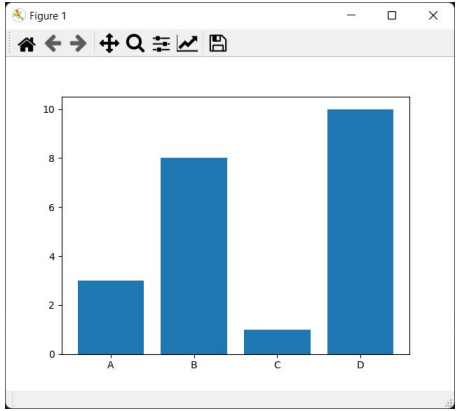
More plotting



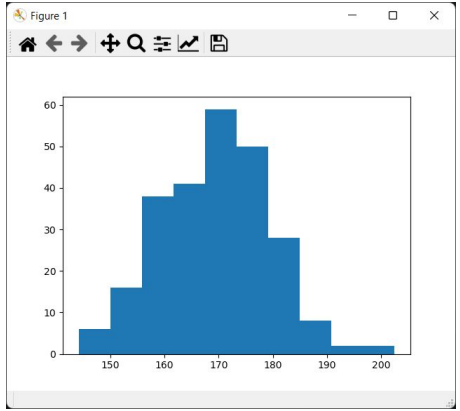
Lines



Scatter



Bars



Histogram

More usage and examples please refer to W3School and Matplotlib documentation