Announcements

• Next Reading and RQ due Thursday
• Assignment 5 due Thursday
  – Next Assignment out after week
• APT 5 due tonight, APT 6 due March 23
• Lab this week - images

• Today:
  – Nested loops, tuples, images
  – Focus on problem solving with sets
Finish notes from last time
Problems — snarf setExample.py

• Given a list of strings that have the name of a course (one word), followed by last names (one word each) of people in the course:
  1. Find total number of people taking any course
  2. Find number of people taking just one course

["econ101 Abroms Curtson Williams Smith", "history230 Black Wrigley Smith", ... ]

Process data – create lists of strings of names for each course
Data for example


TO easier format to work with:


compsci101 fall16
Set Picture of Data

COMPSCI101
Abroms
Li
Ye
Lin

MATH101
Green
Noell
Wei
Yavatkar
Delong

HISTORY230
Wrigley

FRENCH1
Wills
Lee
Olson

ECON101
Curtson
Williams

Smith
Black

Li

compsci101 fall16
People in CompSci 101

- Abroms
- Li
- Smith
- Ye
- Lin

People in Math 101

- Green
- Noell
- Wei
- Yavatkar
- Delong

People in Econ 101

- Curtson
- Williams

People in History 230

- Wrigley
- Lee
- Olson

People in French 1

- Wills
People Taking both Math And CompSci

Intersection

ECON101

COMPSCI101

MATH101

FRENCH1

HISTORY230

Smith

Ye

Li

Abroms

Lin

Ye

Wrigley

Green

Noell

Wei

Yavatkar

Delong

Black

Lin

Wills

Lee

Olson

Intersecting students taking both Math and CompSci.
Part 1 — processList

• Given a list of strings that have the name of a course (one word), followed by last names of people in the course:
  – Convert list into lists of strings of names for each course

[ "econ101 Abroms Curtson Williams Smith",
  "history230 Black Wrigley Smith",
  ... ]

Part 2 – peopleTakingCourses

• Given a list of lists of names, each list represents the people in one course:
  – Find total number of people taking any course
  – peopleTakingCourses should return unique list of names

• Small Example

[[‘Abroms’, ‘Curtson’, ‘Williams’, ‘Smith’],
[‘Black’, ‘Wrigley’, ‘Smith’]]

Answer is 6 unique names
Total Number Is 17 unique names

People taking Courses - Union

ECON101
Curtsone, Williams

COMPSCI101
Abroms, Li

MATH101
Ye, Lin, Black, Green, Noell, Wei, Yavatkar, Delong

FRENCH1
Wrigley, Wills, Lee, Olson

HISTORY230
Smith, Black, Wrigley
Next, find the number of people taking just one course
To solve this problem

• First let’s write a helper function
Part 3 — unionAllSetsButMe

- Given example, a list of sets of strings, and the index of one of the sets, return the union of all the sets but that one

```python
example = [set(['a', 'b', 'c']), set(['b', 'c', 'd', 'g']), set(['e', 'd', 'a'])]
unionAllSetsButMe(example, 1) is set(['a', 'b', 'c', 'e', 'd'])
```
Part 4 — peopleTakingOnlyOneCourse

Given a list of lists of strings of names representing people from courses
  – Find number of people taking just one course

[[‘Abroms’, ‘Curtson’, ‘Williams’, ‘Smith’],

4
People taking Only one course

ECON101

Smith

COMPSCI101

Abroms

Ye

Lin

MATH101

Green

Noell

Wei

Delong

Yavatkar

HISTORY230

Wrigley

FRENCH1

Wills

Lee

Olson

People taking Only one course

Li

People taking Only one course

Black

Williams

Curtson

Li

People taking Only one course

Ye

Lin

People taking Only one course

Green

Noell

Wei

Delong

Yavatkar

People taking Only one course

Wills

Lee

Olson

People taking Only one course
APT - UniqueZoo

filename: UniqueZoo.py

def numberUnique(zoos):
    """
    Parameter zoos is a list of strings, each string is the types of animals the zoo has, separated by blanks. Return the number of zoos that have at least one unique animal that does not appear at any other zoo """

    # you write code here

• How do you solve this problem?
• How is it similar to the problem we just solved
Example Data for UniqueZoo

["zebra bear fox elephant","bear crocodile fox", "rhino elephant crocodile kangaroo", "elephant bear"]
UniqueZoo – two zoos have unique animals

- zebra
- fox
- bear
- elephant
- crocodile
- rhino
- kangaroo
Problem: Given list of words, find word with most vowels

• Example:
  – Given ['dog', 'cat', 'gerbil', 'elephant']
  – 'elephant' has 3 vowels, the most

• To solve – nested loops:
  – Loop over words in list
    • For each word: Loop over characters in word
def wordWithMostVowels(words):
    maxcnt = 0
    maxword = ""
    cnt = 0
    for word in words:
        for letter in word:
            if isVowel(letter):
                cnt += 1
        if cnt > maxcnt:
            maxcnt = cnt
            maxword = word
    return maxword
Problem — Given two lists of names, print a list of pairs of names in which the two names are the same length

• A = ['mo', 'ted', 'bill']
• B = ['billie', 'jes', 'bo']

• To solve
  – for name in A:
    for name in B:
      Check length
      print pair
          mo, bo
          ted, jes
for aname in A:
    for bname in B:
        if len(aname) == len(bname):
            print aname + ", " + bname
print
for bname in B:
    for aname in A:
        if len(aname) == len(bname):
            print aname + ", " + bname
Tuples

• Like a list, but cannot change them
  – Define them with “,”
    (5, 7, 8) or 5, 7, 8

• Use most list operations on them
  – they are a type of list
  – But immutable

• Examples
Example

\[
x = (4, 6, 8) \\
y = 9, 5, 6 \\
print x \\
print y \\
print x[1] \\
print y[1] \\
y[0] = 2 \\
z = ([5,6], [7,8])
\]

\[
print z \\
z[0][1] = 12 \\
print z \\
z[0].append(4) \\
print z \\
z[0].remove(5) \\
z[0].remove(12) \\
z[0].remove(4) \\
print z
\]
Image Processing

• What's real, what's Photoshopped
  – Learn more at http://bit.ly/1Psi0hG, we'll do very basic stuff in class and lab, next assignment too!
Example: convert color to gray scale

Process each pixel
Convert to gray
Example: convert blue to green

Process each pixel
Convert blue ones to green

Is this like red-eye removal?
Lab 7

• You’ll create new images
  – Invert
  – Solarize
  – Darken
  – Brighten
  – etc
Need new concepts and Image library

• Red, Green, Blue color model
  – Triples of (R,G,B) are processed as Python tuples.
  – *Let's study tuples!*

• Images can be very big, what's 4K display?
  – 4,096 x 2,160 = 8,847,360 pixels, 8Mb at least
  – Creating huge lists takes up memory
  – Sometimes only need one pixel at-a-time
  – *Let's study generators!*
Need new concepts and Image library

• Red, Green, Blue color model
  – Additive model, each pixel specified by $(r,g,b)$ triple, values of each between 0-255
  – White is $(255,255,255)$ and Black is $(0,0,0)$

• Images stored as sequence of $(r,g,b)$ tuples, typically with more data/information too
  – 256 values, represented as 8 bits, $2^8 = 256$
  – 32 bits per pixel (with alpha channel)
  – In Python we can largely ignore these details!
Image library: Two ways to get pixels

- Each pixel is a **tuple** in both models
  - Like a list, indexable, but **immutable**
  - pix = (255, 0, 0)
    - What is pix?, pix[0]? What is pix[5]?  

- Invert a pixel: by subscript or named tuple
  - Access by assignment to variables!
  
  \[
  \text{npx} = (255 - \text{pix}[0], 255 - \text{pix}[1], 255 - \text{pix}[2])
  \]

\[
(r, g, b) = \text{pix} \\
\text{npx} = (255 - r, 255 - g, 255 - b)
\]
Let's look at GrayScale.py

• Key features we see
  – Import Image library, use API by example
  – Image.open creates an image object

• Image functions for Image object im
  – im.show(), displays image on screen
  – im.save("xy"), saves with filename
  – im.copy(), returns image that's a copy
  – im.load(), [x,y] indexable pixel collection
  – im.getdata(), iterable pixel collection

• Let's look at two ways to process pixels!
Image Library: open, modify, save

- **Image.open** can open most image files
  - .png, .jpg, .gif, and more
  - Returns an image object, so store in variable of type Image instance
  - Get pixels with `im.getdata()` or `im.load()`

- **Image.new** can create a new image, specify color model "RGB" and size of image
  - Add pixels with `im.putdata()`

- These belong to Image package
**im.getdata(), accessing pixels**

- Returns something *like* a list
  - Use: `for pix in im.getdata():`
  - Generates pixels on-the-fly, can't slice or index unless you use `list(im.getdata())`
  - Structure is called a Python generator!
  - Saves on storing all pixels in memory if only accessed one-at-a-time

- See usage in GrayScale.py, note how used in list comprehension, like a list!
Questions about Image Code

bit.ly/101s17-0307-5
Alternate : Still Tuples and Pixels

• The `im.getdata()` function returns list-like iterable
  – Can use in list comprehension, see code
  – Use `.putdata()` to store again in image

```python
pixels = [makeGray(pix) for pix in im.getdata()]
```

```python
def makeGray(pixel):
    r,g,b = pixel
    gray = (r+g+b)/3
    return (gray,gray,gray)
```
Making Tuples and Generators

• Overuse and abuse of parentheses
  – To create a tuple, use parentheses
    
    ```python
    for pix in im.getdata():
        (r,g,b) = pix
        npx = (255-r,255-g,255-b)
    ```
  
  – To create a generator use parentheses as though creating a list comprehension!
    
    ```python
    [2*n for n in range(10000)]
    (2*n for n in range(10000))
    ```

• See this in PyDev console