CompSci 101
Introduction to Computer Science

Oct 31, 2017

Prof. Rodger

Announcements

- Next Reading and RQ due Thursday
- Assignment 6 due Thursday
- APT 5 due tonight, APT 6 due Nov 7
- APT Quiz2 Sun-Wed next week
- Lab this week - images

- Today:
  – Nested loops, tuples, images

ACM Programming Contest Need Volunteers Saturday, Nov 11 at Duke

- Over 120 teams, 8 university sites
- Team:
  – 3 people, 1 computer
  – 8 problems, 5 hours
- Need volunteers to deliver printouts, etc
  – 8:15am-12:30 OR 11:20am-6pm
  – Get tshirt and meals!

world finals acm icpc April 15-20 icpc 2018 Beijing

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

Bit.ly/101f17-1031-1

```
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’]

mo, bo

ted, jes

- To solve
  - for name in A:

```
for name in B:
    Check length
        print pair
```

Bit.ly/101f17-1031-2

```
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
```

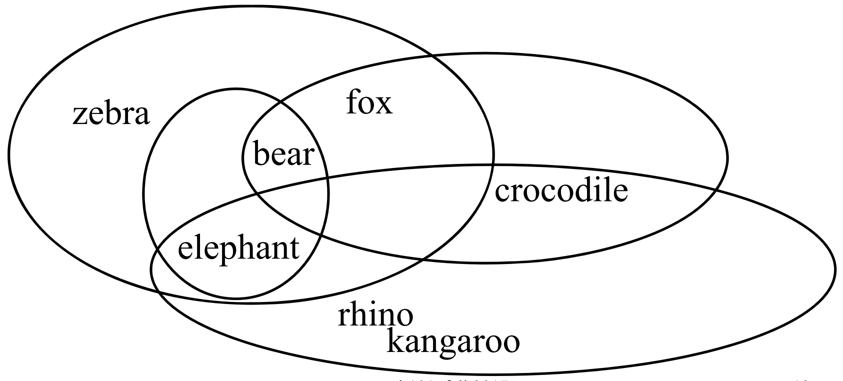
APT - UniqueZoo

```
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

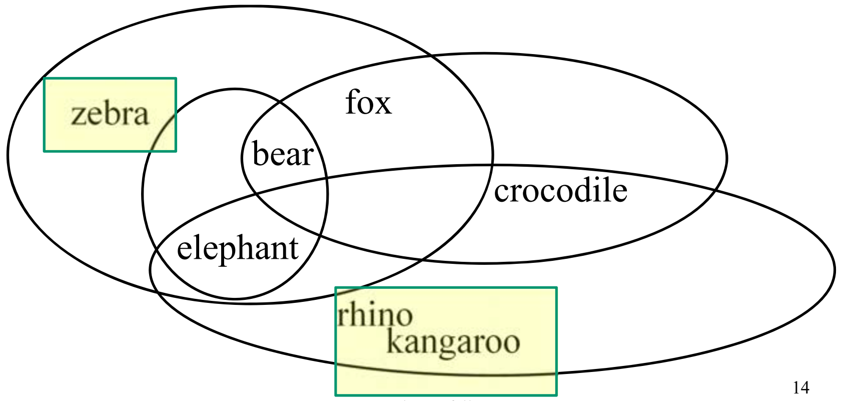
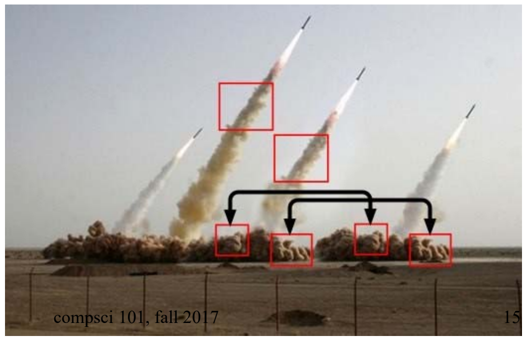


Image Processing

- What's real, what's Photoshopped
  - http://bit.ly/1Kj0Kn6 from 2008
  - 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 8

- 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
  - https://en.wikipedia.org/wiki/RGB_color_model
  - 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!

npx = (255-pix[0],255-pix[1],255-pix[2])

(r,g,b) = pix
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!

Pixels in an image
Background is black, 12x9

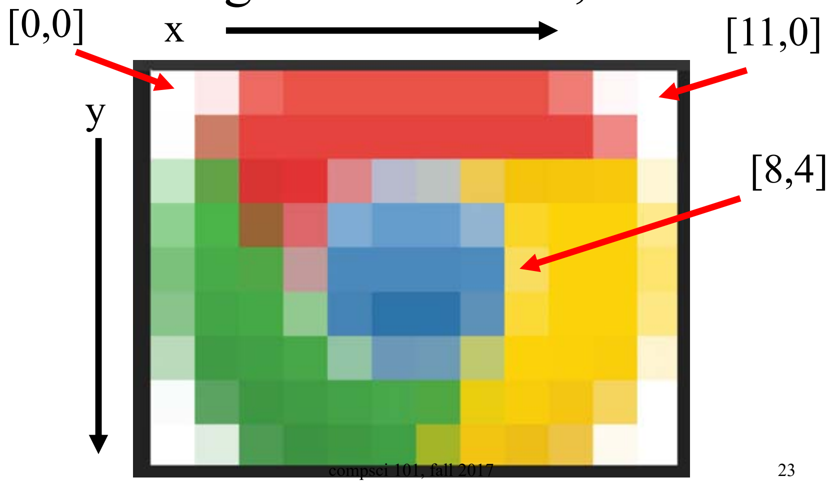


Image Library: open, modify, save

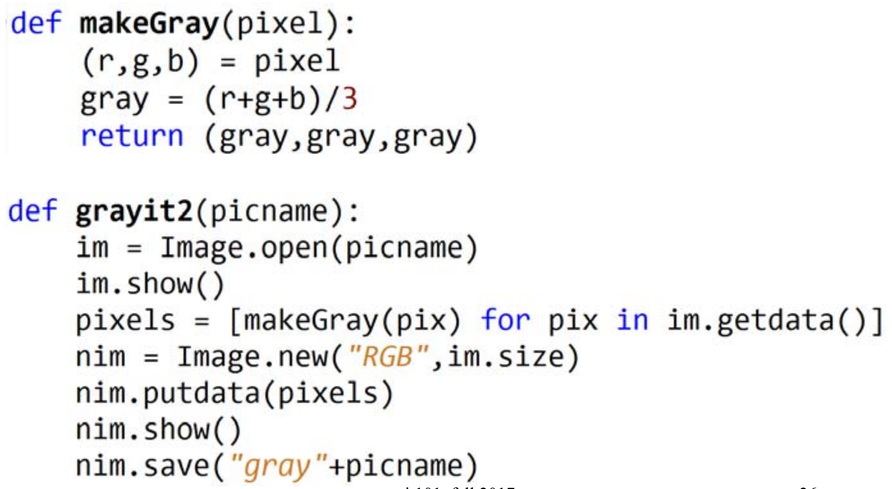
- `Image.open` can open most image files
  - .png, .jpg, .gif, and more
  - Returns an image object, so store in variable of type Image instance
  - Get pixels with `im.getdata()` or `im.load()`
- `Image.new` can create a new image, specify color model "RGB" and size of image
  - Add pixels with `im.putdata()`

- 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
bit.ly/101f17-1031-3



```
def makeGray(pixel):
    (r,g,b) = pixel
    gray = (r+g+b)/3
    return (gray,gray,gray)

def grayit2(picname):
    im = Image.open(picname)
    im.show()
    pixels = [makeGray(pix) for pix in im.getdata()]
    nim = Image.new("RGB",im.size)
    nim.putdata(pixels)
    nim.show()
    nim.save("gray"+picname)
```

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

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

```
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

```
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!

```
[2*n for n in range(10000)]
(2*n for n in range(10000))
```

- See this in PyDev console