CompSci 101
Introduction to Computer Science

EARTHQUAKE!!!!!!!

Sept 19, 2017

Prof. Rodger

Announcements

- Reading and RQ7 due next time
- Assignment 3 due Thursday
- APT 2 due today, APT 3 out
- APT Quiz 1 – Sunday -Tuesday night
  - Up for about 3 days, you pick 3 hours to do it
  - Practice quiz out by Friday
- Today
  - Calculating info about earthquake data
  - Really doing? (Functions, if, strings, lists)

Lab 4 this week

- Lists – indexing and splicing (like strings!)
- Processing a data file
  - Putting each line from file in a different form
  - Writing functions to calculate information about the data

Stuck on solving a problem? Don’t know where to start?

- Use the 7 step process!

Problem Solving to Code
7 Step Process

1. Work small examples by hand
2. Write down what you did in words (algorithm)
3. Find Patterns (generalize algorithm)
4. Work another example by hand (does your algorithm work? If not, go back to 2)
5. Translate to code
6. Test several cases
7. Debug failed test cases

APT: Last Name First

Problem Statement

Sabrina needs to be able to reorganize names into the last name first and she wants to abbreviate any middle names with the first letter and a period. She respects middle names that are a single letter and does **not** abbreviate them.

Write function `modify` that given a name returns the name with the last name first, followed by a comma, followed by the first name (if any), followed by the first letter of each remaining/middle name with a period after each letter. If a middle name is a single letter, do not abbreviate it/follow it by a period.

Specification

```
filename: LastNameFirst.py

def modify(name):
    """
    return the name with the last name first, followed by
    a comma, followed by the first name (if any), followed
    by the first letter of each remaining name with a
    period after each letter.
    name has at least one word.
    """

    # you write code here
```

2.
```
name = "Prince"

returns "Prince"

There is only one name.
```

3.
```
name = "Thomas Narten"

returns "Narten, Thomas"

There is no middle name.
```

4.
```
name = "Elizabeth Rosemond Hilton Wilding Todd Fisher Burton Warner Fortensky Taylor"

returns "Taylor, Elizabeth R. H. W. T. F. B. W. F."

All the middle names are abreviated.
```

Step 1) Work small example by hand

- **name** is “Moe Jess Bo Lu Yue”
- **first** is “Moe”
- **las**t is “Yue”
- **middle** is “Jess Bo Lu”
  - “Jess” gives us “J.”
  - “Bo” gives us “B.”
  - Join together “J. B.”
  - “Lu” gives us “L.”
  - Join together “J. B. L.”
- Last, First Middle: “Yue, Moe J. B. L.”

Step 2) Describe in words what you did

- Name is: “Moe Jess Bo Lu Yue”
- Determine first name: “Moe”
- Determine last name: “Yue”
- Determine all middle names: “Jess Bo Lu”
  - – Look at first word in middle: “Jess”
  - – newMiddle is “J.”
  - – Look at second word: “Bo”
  - – NewMiddle is “J. B.”
  - – Look at third word: “Lu”
  - – NewMiddle is “J. B. L.”
- Put together last, first and newMiddle

Step 3) Find Patterns (Generalize)? Don’t see it? Work another example

- Name is: “Sue Mo Lucy Lo So Fa Ti”
- Firstname is “Sue”, Lastname is “Ti”
- Middle is “Mo Lucy Lo So Fa”
  - “Mo”, newMid is “M.”,
  - “Lucy”, newMid is “M. L.”
  - “Lo”, newMid is “M. L. L.”
  - “So”, newMid is “M. L. L. S.”
  - “Fa”, newMid is “M. L. L. S. F.”
- Put together: “Ti, Sue M. L. L. S. F.”

Step 3) Find Patterns (Generalize)

- Name is: “Moe Jess Bo Lu Yue”
- Firstname is first word
- Lastname is last word
- Middle is string of all the middle words
- Initialize newMiddle
- For each word in middle:
  - Add first letter of word, period and blank to newMiddle
- Build new string:
  - lastname, firstname newMiddle
- Return answer

Step 4) Work another example by hand using your algorithm

- Name = “Jo Flo Bo Yup”
- Firstname = “Jo”
- Lastname = “Yup”
- Middle = “Flo Bo”
- newMid = “”
- For word in Middle:
  - newMid = “F.” (first time thru loop)
  - newMid = “F. B.” (second time thru loop)
- Answer = “Yup, Jo F. B.”

IT WORKS!

Step 5) Translate to Code

- Firstname is first word

- Lastname is last word

- Middle is string of all the middle words

Step 5) Translate to Code

- Initialize newMiddle
- For each word in middle:
  - Add first letter, period and blank to newMiddle
- Build new string:
  - lastname, firstname newMiddle
- return answer

Step 6) Test Several Cases

- Does our algorithm work for?
- Name = “Ronald McDonald”
- Name = “Simon”
- Name = “Felicia Mary Moffet”

- Need to handle special cases

Step 7) Debug Failed Test Cases

- How do you debug? Some tips
  - Isolate where the problem is
  - You think your code is correct, but is it?
  - Print out the value of variables.
  - Break code apart and print
  - Print, print, print
  - Identify your output
  - OR put function in Python tutor and call it on an example

Break Code apart so you can print parts of it

```
return foo(y, 3, calculate(y, j))
```

INSTEAD:

```
temp = calculate(y,j)
print “temp is”, temp
temp2 = foo(y, 3, temp)
print “temp2 is”, temp2
return temp2
```

Comment out prints for debugging later with #

```
#print “temp” is temp
```

Given a string of words, this function should return a new string with ‘s’ removed at the end of every word
bit.ly/101f17-0919-1

```
def removePlurals(phrase):
    answer = ""
    alist = phrase.split()
    for word in alist:
        if word[-1] == "s":
            answer = word[:-1] + " "
         else:
            answer = word + " "
    return answer.strip()
```

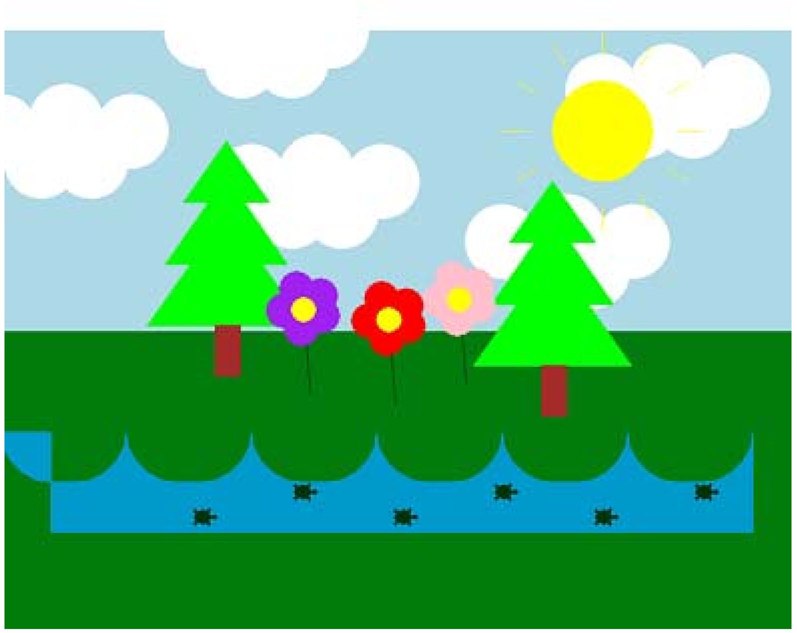
Lynn Conway

See Wikipedia and lynnconway.com

- Joined Xerox Parc in 1973
  - Revolutionized VLSI design with Carver Mead
- Joined U. Michigan 1985
  - Professor and Dean, retired '98
- NAE '89, IEEE Pioneer '09
- Helped invent dynamic scheduling early '60s IBM
- Transgender, fired in '68

Assignment 3

- Turtles
  - Creative



- Earthquakes
  - Data from last 30 days around the world
  - Example - Find the largest earthquake

EarthQuake
bit.ly/101f17-0919-2

```python
def quakesMystery(letter, quakes):
    loclist = []
    for item in quakes:
        p = getParts(item)[-1]
        if letter in p:
            loclist = loclist + [item]
    return loclist
```

```
eqList = ['earthquake, 2.4, 14km WSW of Willow, Alaska',
          'earthquake, 0.9, 4km NNW of The Geysers, California',
          'earthquake, 2.1, 13km ESE of Coalinga, California',
          'earthquake, 4.1, 8km E of Maltignano, Italy',
          'earthquake, 1.4, 17km N of Meadow Lakes, Alaska',
          'earthquake, 0.7, 28km SW of Hawthorne, Nevada',
          'earthquake, 1.8, 41km W of Anchorage, Alaska']
```

EarthQuake (cont)

```
def quakesMystery2(quakes):
    temp = ""
    for item in quakes:
        one = getParts(item)[-1]
        if len(one) > len(temp):
            temp = one
    return temp
```

```
eqList = ['earthquake, 2.4, 14km WSW of Willow, Alaska',
          'earthquake, 0.9, 4km NNW of The Geysers, California',
          'earthquake, 2.1, 13km ESE of Coalinga, California',
          'earthquake, 4.1, 8km E of Maltignano, Italy',
          'earthquake, 1.4, 17km N of Meadow Lakes, Alaska',
          'earthquake, 0.7, 28km SW of Hawthorne, Nevada',
          'earthquake, 1.8, 41km W of Anchorage, Alaska']
```

How to tackle the earthquake part of assignment?

- Write one function at a time
- TEST IT, make sure it works! Super important!!
- Then move to next function
- **Example**: Write a function named **getParts**.
- getParts("earthquake, 1.3, 81km SSW of Kobuk, Alaska") would return
- [1.3, "earthquake", "81km SSW of Kobuk, Alaska"]

Calling your Earthquake functions

Assume eqList is a list of all the earthquakes, each earthquake is a string in the correct format.

```
quakes1 = bigQuakes(3.0, eqList)
quakes2 = locationQuakes(“Alaska”, quakes1)
printQuakes(quakes2, 5)
```

What can you say about quakes2?

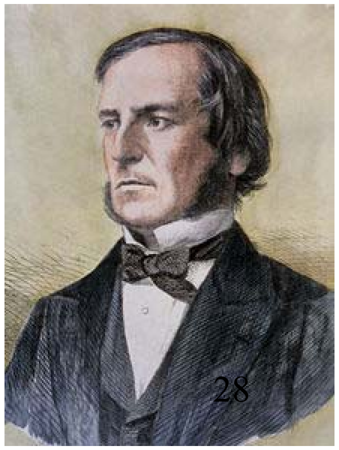
Type, value

Python if statements and Booleans

- In python we have if: else: elif:
  - Used to guard or select block of code
  - If guard is True then code block, else other

- What type of expression used in if/elif tests?
  - ==, <=, <, >, >=, !=, and, or, not, in
  - Value of expression must be either True or False
  - Type is bool - George Boole, Boolean,

- Examples with if
  - String starts with vowel
    (useful for APT Emphasize)



Four versions of isVowel?
bit.ly/101f17-0919-3

A

```
def isVowel(ch):
   if ch =='e':
      return True
   if ch == 'a':
      return True
   if ch == 'i':
      return True
   if ch == 'o':
      return True
   if ch == 'u':
      return True
   return False
```

B

```
def isVowel(ch):
   c = "aeiou".count(ch)
   if c > 0:
       return True
```

C

```
def isVowel(ch):
   return "aeiou".count(ch) > 0
```

D

```
def isVowel(ch):
   if ch in "aeiou":
       return True
   else:
       return False
```

Anatomy of a Python String

- String is a sequence of characters
  - Functions we can apply to sequences: len, slice [:], others
  - Methods applied to strings [specific to strings]
    - st.split(), st.startswith(), st.strip(), st.lower(), …
    - st.find(), st.count()
- Strings are *immutable* sequences
  - Characters are actually length-one strings
  - Cannot change a string, can only create new one
    - What does upper do?
  - See resources for functions/methods on strings

- *Iterable*: Can loop over it, *Indexable*: can slice it

Counting vowels in a string

- Accumulating a count in an int is similar to accumulating characters in a string

```
def vowelCount(word):
    value = 0
    for ch in word:
        if isVowel(ch):
            value = value + 1
    return value
```

- Alternative version of adding:

```
value += 1
```

Incremental + : numbers and strings

- Wtht vwls cn y stll rd ths sntnc?
  - Create a no-vowel version of word
  - Examine each character, if it's not a vowel …
  - Pattern of building a string

```
def noVowels(word):
    ret = ""
    for ch in word:
        if not isVowel(ch):
            ret = ret + ch
    return ret
```

APT Emphasize

- Use a helper function! isVowel(ch)