Compsci 101, Loops and Patterns

Owen Astrachan
Kristin Stephens-Martinez
February 6, 2018
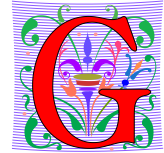
G is for …

- Google
  - How to find the answer to everything
- Global Variable
  - Accessible everywhere, don't do this at home?
- GIGO
  - Garbage In, Garbage Out
- Git
  - Working Together or Solo

PF The Week

- Loops over Lists and Files
  - Accumulator Idiom: initialize and update
  - How do process data? Designing Software
  - Idiomatic programming
  - For each compared to For index

- Turtles
  - APIs, loops, indexing, creativity

Administrivia

- What's due this week?
  - APT Quiz, Lab

- What's coming next week?
  - Assignment 2, APT 2, Midterm

- What grades have been recorded?
  - Updated soon

Revisit/Review Vowel Removal

- Let's explore this: "Dk Cmptr Scnc s s mch fn!"
  - How do we create this sentence?
  - Input is sentence. Output has vowels removed

- First we do with paper and pencil. Always
  - Do not skip this step!
  - Always solve by hand before coding!

- How do we translate ideas into code?

From Idea to code

- Wtht vwls cn y stll rd this sntnc?
- Accumulator idiom: change variable in a loop
  - Variable accumulates new value based on old
  - Understanding "gets" the assignment operator

```
def noVowels(word):
    ret = ""
    for ch in word:
        if "aeiou".count(ch) == 0:
            ret = ret + ch
    return ret
```

More than one way to …

- Boolean
  - not
  - True
  - False

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

```
def noVowels(word):
    ret = ""
    for ch in word:
        if isvowel(ch) == False:
            ret = ret + ch
    return ret
```

Two, Two, Two Accumulator Loops

- https://goo.gl/6PvDv5
- Let's look at all the pieces in action
  - Examine loop and accumulators carefully

- Initialize, Update, Finalize
  - Why initialize to empty string?
  - Why does update use +, concatenation?
  - Why is there a leading space in answer ☹

Processing Data

- How do we find the longest word in .. Any text?
- How do we find the word that occurs the most?
- How is this related to how Google Search works?

- Text files can be viewed as sequences
  - Sequences of lines
  - Each line is a string
  - Some clean-up because of '\n'

One line at a time

- Simplest and reasonably efficient Python idiom
  - https://trinket.io/python3/0e04f63c31
  - Open, loop, close, return/process
- File as sequence
  - One line at-a-time

```
1 - def lineCount(fname):
2       """
3         return # lines in file fname
4       """
5       f = open(fname)
6       lc = 0
7 -     for line in f:
8           lc = lc + 1
9
10      f.close()
11      return lc
```

- Asymmetry
  - Open v Close

File Objects

- A file is an object, like a string
  - Functions applied to object: **len("word")**
  - For files we have **open("data.txt")**
  - What is returned? Integer value, file object

- Often methods (aka function) applied to object
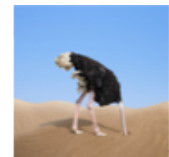 - **st.lower(), st.count("e")**
  - **f.readlines(),f.read(), f.close()**

Aside: The Object Concept

- Sometimes it helps to know how things "work"
  - Sometimes it's wonderful to be oblivious

- Python variables are references
  - Label that refers to object
  - Label is small, object is big

- Example to follow

PythonTutor or Console

- https://goo.gl/QjD39s

```
Python 3.6
1  a = ["pig", "cow", "dog", "bat"]
2  b = a
3  print(len(a))
4  print(len(b))
5  a.append("ant")
6  print(b)
```

```
Global frame          list
   a                   0      1      2      3
   b                  "pig"  "cow"  "dog"  "bat"
```

Compsci 101, Spring 2018, Loops and Patterns   13

Immutable built-in Types

- In python string, int, float, boolean .. Immutable
  - Once created cannot change
  - These are objects in Python3!!
- PythonTutor gets this wrong

```
val = 0
bee = val
val = val + 20
```

- Objects don't change
  - Labels/names change

```
val = "apple"
val = val + "sauce"
bee = val + "sauce"
```
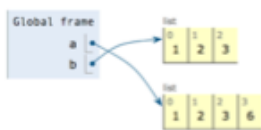
Compsci 101, Spring 2018, Loops and Patterns   14

Scourge of the Mutable

- https://goo.gl/gkFvtx

```
Python 3.6
1  a = [1,2,3]
2  b = a
3  a = a + [6]
4  print(b)
```

```
Frames          Objects
Global frame     list
   a             1  2  3
   b
                 list
                 1  2  3  6
```

```
a = [1,2,3]
b = a
a = a + [6]
print(b)
```

```
a = [1,2,3]
b = a
a.append(6)
print(b)
```

Compsci 101, Spring 2018, Loops and Patterns   15

WOTO

http://bit.ly/101spring18-feb6-1

Object
Computer science

In computer science, an object can be a variable, a data structure, a function, or a method, and as such, is a location in memory having a value and referenced by an identifier. Wikipedia

Compsci 101, Spring 2018, Loops and Patterns   16

Brian Fox

- See Wikipedia: http://bit.ly/brianfox2018
  - Bash Shell, Stallman, Wells Fargo, more
- See LifeHacker: http://bit.ly/brianfox-hack
  - Learned Logo at 8, wrote it at 21 for Apple!
  - Now? Open Voting

There's nothing that I am better at than everyone else, except being me. There's no secret to being me. Follow your interests and work hard at them. Then you will play bass better, program better, cook better, ride motorcycles better, or anything else that you really want to do.

2/6/2018 — Compsci 101, Spring 2018, Loops and Patterns — 17

Text File Processing Idioms

- See module **FileStuff.py** in download for today
  - Newline '\n' is read, call **.strip()**
  - Break line into "words", call **.split()**

- Process the list returned by **.split()**
  - May need to convert strings to int or float or …

- The **for line in f:** idiom is efficient
  - Contrast list returned by **f.readlines()**

2/6/2018 — Compsci 101, Spring 2018, Loops and Patterns — 18

YALALA

- Yet Another Look At Loop and Accumulate
  - Initialize, Update, Finalize
  - Before loop, In Loop, After Loop

- Do we care about efficiency with lists? It depends!
  - Quick look at **ListChanger.py**
  - Differences **+[val]** and **.append(val)**
  - Total Storage created in **buildPlus**? Why?
    - This is what we study in detail in Compsci 201

2/6/2018 — Compsci 101, Spring 2018, Loops and Patterns — 19

How was this created?

2/6/2018 — Compsci 101, Spring 2018, Loops and Patterns — 20

Turtle Programming

- Must import turtle module
  - Must create window/Screen and exit on click
  - Must create turtles to use, name/type/value
  - Review Turtle commands and concepts

- See Snowpeople.py, ColorMyWorld, and Spiro.py for some ideas
  - Color, Position, Leaving Turtle where started
  - Many more commands than this

Turtle Concepts

- http://bit.ly/turtle_tutorial for more, and book
  - What are key concepts in Spiro.py?

```python
7  import turtle
8
9  def draw():
10     colors = ['red', 'purple', 'blue', 'green', 'yellow', 'orange']
11     ola = turtle.Turtle()
12     ola.speed(0)
13     for x in range(360):
14         ola.pencolor(colors[x % 6])
15         ola.width(x / 100 + 1)
16         ola.forward(x)
17         ola.left(59)
18
19  if __name__ == '__main__':
20      win = turtle.Screen()
21      draw()
22      win.exitonclick()
```
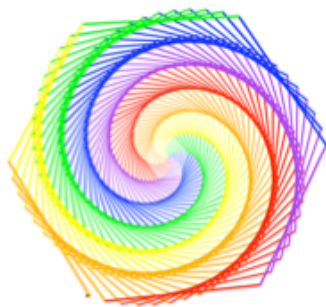
Run Turtle, Run

Repetition with Turtles

- Sometimes rather than looping over a sequence o values you want to repeat # times
  - Do this 4 times
  - Do that 250 times

- The Python range function helps
  - The function call range(x,y) starts at x and goes up to but doesn't include y.
  - Sequence that provides access to int values

Turtle Concepts

- Create a screen so you can ..
  - Exit On Click
  - Some other Screen Functions
- Create a turtle so you can …
  - Move and draw using the turtle
- Drawing Concepts
  - Pen [up and down] Fill
  - Color
  - Position

2/6/2018　　Compsci 101, Spring 2018, Loops and Patterns　　25

Range Examples

- Access all the values in a list to print them
  - Use the "for each" idiom
  - Use an index to access i^{th} element
- Names?

```
xx = ["ant", "bat", "cat", "dog"]
for a in xx:
    print(a)
```

```
xx = ["ant", "bat", "cat", "dog"]
for i in range(len(xx))
    print(xx[i])
```

2/6/2018　　Compsci 101, Spring 2018, Loops and Patterns　　26

WOTO

http://bit.ly/101spring18-feb6-2

- Why doesn't query Logo Turtle work?

SEA TURTLE

2/6/2018　　Compsci 101, Spring 2018, Loops and Patterns　　27