PFTD: Vocabulary and stories

- Thinking about problems first: motivate language
  - Instead of looking at language first, look at problems
  - Motivate Python types and control by applications
- What's one of the steps in writing a search engine?
  - Python in the news? http://bit.ly/y7yeXX
  - One step in ranking a webpage: who links to me?
- Running computational experiments
  - Simulations for: finance, weather, biology, ...
  - Repetition with randomness: statistical analysis

Some steps in a search engine

- What's the HTML at duke.edu?

```
<li id="nav-academics"><a href="http://academics.duke.edu">Academics</a></li>
<li id="nav-medical"><a href="http://www.dukemedicine.org/">Medical</a></li>
<li id="nav-research"><a href="http://research.duke.edu">Research</a></li>
<li id="nav-global"><a href="http://global.duke.edu">Global</a></li>
<li id="nav-arts"><a href="http://arts.duke.edu">Arts</a></li>
<li id="nav-libraries"><a href="http://library.duke.edu">Libraries</a></li>
<li id="nav-athletics"><a href="http://goduke.com">Athletics</a></li>
<li id="nav-giving"><a href="http://giving.duke.edu">Giving to Duke</a></li>
```

  - http://library.duke.edu
- How do we find this URL? How do we find all?
  - How do you do it, how do we get Python to do it?

What is a web-page?

- It depends
  - From whose point of view?
  - Sequence of characters? images?
  - What does rendering software/engine do?
  - What does web-spider/crawler do
- What is a file?
  - What have we done to a file in Python?
  - How did we do this?
- What functions/operations exist for strings, files, ...

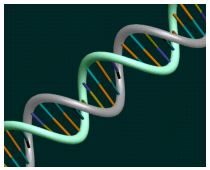
Sequences, functions, and loops

- What is a string? Sequence of characters
  - What functions are native/built-in to Python
  - What is a return value, different from print
  - How do we use strings in Totem.py?
- Loops, Lists, Strings : processing data from URL
  - Loop over sequence: string, file, list, "other"
  - Process each element, sometimes selectively
  - Toward understanding the power of lists
- Accumulation as a coding pattern

Understanding cgratio APT

- How do you count 'c' and 'g' content of a string?
  - What's going on below? What does the code do?

```
def cgcount(strand):
    cg = 0
    for nuc in strand:
        if nuc == 'c' or nuc == 'g':
            cg = cg + 1
    return cg
```



- Types in the code above
  - What is a string, what is an int

Counting words: accumulation

- Anatomy of assignment and accumulation
  - var = "hello", y = 7
  - What do these do? Memory?
  - Reading assignment statement

- Accumulation

```
var = 0
for x in data:
    if x == "a":
        var = var + 1
```

- RHS, assign to LHS

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

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

Coding Interlude

- What's going on with CountAppearances?
  - How do you count digits in a number
  - How do you count occurrences in a string?
- What can be tested with/in if statement
  - Boolean conditions: <, <=, ==, !=, =>, >
  - Boolean operators: not, and, or
  - True, False
- What's the purpose of an APT

From high- to low-level Python

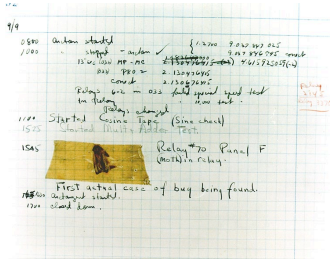
```
def reverse(s):
  r = ""
  for ch in s:
    r = ch + r
  return r
```

- Create version on the right using dissassembler
  `dis.dis(code.py)`

```
 7           0 LOAD_CONST               1 ('')
             3 STORE_FAST               1 (r)

 8           6 SETUP_LOOP              24 (to 33)
             9 LOAD_FAST                0 (s)
            12 GET_ITER
        >>  13 FOR_ITER                16 (to 32)
            16 STORE_FAST               2 (ch)

 9          19 LOAD_FAST                2 (ch)
            22 LOAD_FAST                1 (r)
            25 BINARY_ADD
            26 STORE_FAST               1 (r)
            29 JUMP_ABSOLUTE           13
        >>  32 POP_BLOCK

10      >>  33 LOAD_FAST                1 (r)
            36 RETURN_VALUE
```


Bug and Debug

- software 'bug'
- Start small
  - Easier to cope
- Judicious 'print'
  - Debugger too



- Verify the approach being taken, test small, test frequently
  - How do you 'prove' your code works?

Making choices at random

- Why is making random choices useful?
  - How does modeling work? How does simulation work?
  - Random v Pseudo-random, what's used?
  - Online gambling?
- Python random module/library: import random
  - Methods we'll use: `random.random()`, `random.randint(a,b)`, `random.shuffle(seq)`, `random.choice(seq)`, `random.sample(seq,k)`, `random.seed(x)`
- How do we use a module?