PFTWeek

● Introduce new Python concepts
  ➢ Control: if, elif, else, for
  ➢ Data: Strings, Lists
    • Operators on data: slicing, methods, functions
  ➢ Variables and constants:
    • Names, types, and values

● Review Organization and Problem-Solving
  ➢ Defining functions, calling functions
  ➢ Return types, print, None
Variables, Types, Values

- **Variable is a name associated with "storage"**
  - Assign a value: `x = 5`
  - Print value of variable: `print x`
  - Use variable in expression: `y = x * 55`

- **String is a type and has a value**
  - Assign: `x = "hello"`
  - Print value of variable: `print x`
  - Use in expression
    - `print len(x)`
    - `print x + " world"`

- **There are more types, this is a start!**
Vocabulary, grammar, rules: Python

● **Naming**
  - The power of abstraction and parameterization
  - What are parameters? What has them (Python, World)?

● **Types**
  - What's used in computing? What's used in Python?
  - Determine names of types in Python, use type(..)

● **Expressions and operators in Python**
  - Arithmetic: +, −, *, /, %, **, …
  - Boolean: <, ==, >, and, . . .
  - String: +, *, [ ], [:], [: :]
Types and values in Python

● **Numbers are important, but not everything is a ...**
  > What is a number? In mathematics, in Python, in Java,
  > Integers, floating-point numbers, complex numbers, ...
  > • We will worry about types, not speed or storage
    (though these are a concern sometimes)
  > • 1, 2, 3 compared to 3.1415, 1.75, math.pi
  > • 5/2 compared to 5.0/2 compared to 5//2

● **Strings are sequences of characters, "python.org"**
  > Somewhere these are converted to numbers: 0's and 1's
  > No real need to know this now.
  > Strings are immutable: make new ones, can't change them
Expressions, Operators, Types

- Why is $3+5\times4$ different than $(3+5)\times4$?
  - Where can you find information about precedence?

- Why is $5/3$ different than $5.0/3$?
  - What will happen in Python 3? Accommodate in 2.7?

- What happens when operators go bad?
  - What is "apple" + 3? What is "apple" + "pi"?
  - What is "apple" * 3? What is "apple" * "pi"?

- What is a variable in Python?
  - Name, Type, Value
Observations about String literals

● Sometimes the details are tricky
  ➢ "I " + "love " + 'Python'
  ➢ "I " + "love " + '"Python"
  ➢ "I " + "love " + "'Python'"

● When in doubt, use parentheses
  ➢ What is "a" + "b" * 3
  ➢ What is "a" "b" * 3
Names, Types, Values Revisited

name = "/data/poe.txt"
ff = open(name)
st = ff.read()
words = st.split()
print "# words in", name, "=" , len(words)

● What are the names in the code above?
  ➢ Why are names important?

● What are the types in the code above?
  ➢ How do we get Python to help us answer this question?

● How do we re-use this code more generally?
  ➢ The power of names! The power of functions!
Slicing and Indexing

- The Python types str (String) and list both support indexing and slicing

\[ s = "blue devils at duke" \]

0123456789012345678

- \( s[1], \ s[2], \ s[0], \ s[50] \)
- \( s[-1], \ s[-4] \)
- \( s[5:11], \ s[15:], \ s[:4], \ s[:] \)
- \( s[5:10:2], \ s[:], \ s[::-1] \)
Value Expert

● Answer these questions

Grace Murray Hopper (1906-1992)

- “third programmer on world's first large-scale digital computer”
  - US Navy: Admiral
    “It's better to show that something can be done and apologize for not asking permission, than to try to persuade the powers that be at the beginning”
    https://www.youtube.com/watch?v=1-vcErOPofQ

- ACM Hopper award given for contributions before 35
  2010: Craig Gentry: http://www.youtube.com/watch?v=qe-zmHoPW30
  2011: Luis von Ahn
  2013: Pedro Felzenszwab
  2014: Sylvia Ratnasamy

- How do you solve this problem?
  - First steps: are there simple cases that can be solved immediately?
    - What are these for the pancake problem?
  - Sometimes it helps to know if you are on track, should you use Python to check your paper and pencil work?

- Get specific, solve for 5, not N
  - Fix one parameter, vary the other
  - Identify the cases and continue
Three pancakes in a two-cake pan...

- **Number of cakes in the system**
  - First 5 minutes

- **Number of cakes in the system**
  - Second 5 minutes
Three pancakes in a two-cake pan…

- **Number of cakes in the system**
  - Third 5 minutes

- **How many minutes to cook all three pancakes?**
Algorithmic Problem/Program Testing

- Complete this form about Pancakes and one other APT (read the other one from form)

How to teach pancake flipping

- [http://www.youtube.com/watch?v=W_gxLKSsSIE](http://www.youtube.com/watch?v=W_gxLKSsSIE)
  - For longer, more complex robotic tasks
    - [http://www.youtube.com/watch?v=4usoE981e7I](http://www.youtube.com/watch?v=4usoE981e7I)

- **Back to specifics:**
  - Capacity = 5
  - Numcakes = 1,2,...5?
  - Numcakes = 6,7,8,9,10?
  - Numcakes = 11,12,13,14,15?

- **Is five special? 4? 3? 2?**
Language and Problems in Context

● Convert Spanish Wikipedia page to English
  ➢ How do we convert HTML to text?

● How do you determine if 2040 is a leap year?
  ➢ Any specified year is a leap year?

● How do we make an image larger, more red, ...
  ➢ What is an image? How do read it? Convert it? Access it?

● Make "Jones, Howard" from "Howard Jones"
What years are leap years?

  - But not 1900, not 2100, yes 2400!
  - Yes if divisible by 4, but not if divisible by 100 unless divisible by 400! (what?)

- There is more than one way to skin a cat, but we need at least one way

```python
def is_leap_year(year):
    if year % 400 == 0:
        return True
    if year % 100 == 0:
        return False
    if year % 4 == 0:
        return True
    return False
```
Three versions of \texttt{is\_vowel}

\begin{verbatim}
def is_vowel(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

def is_vowel(ch):
    c = "aeiou".count(ch)
    if c > 0:
        return True
    else:
        return False

def is_vowel(ch):
    return "aeiou".count(ch) > 0
\end{verbatim}
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, 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 == bool, George Boole, Boolean,

● Examples with if
  ➢ String starts with vowel (useful for APT Emphasize)
Eclipse Interlude

- **Finishing the Pancake problem**
  - Translating problem-solving ideas to code
  - Control with if/elif: arithmetic with / and %
Lessons: special cases, abstractions

● There are special cases in many, many problems
  ➢ Identifying them is important
  ➢ Abstracting them away when possible is important
  ➢ Example: Pancake APT
    • What happens when everything fits in the pan?
    • Can there be a pan with no capacity?

● Solve problems by hand, pencil, brain
  ➢ Can you do pancakes for any pan-size and number? If not, can’t write code!
Software Dreams

● Translating ideas into (Python) code
  ➢ Create interesting “heads”, “totem poles” ?
  ➢ Create software for face recognition? Gait?
  ➢ Create "five four" from "four five"?
  ➢ Create "SCUBA" from "self contained underwater breathing apparatus"

● Master the syntax of the language?
  ➢ Organization of program constructs
  ➢ Knowledge of libraries
  ➢ Practice and experience!
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**
From high- to low-level Python

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

● Create version on the right using disassembler

```
dis.dis(code.py)
```

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

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

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

10    33 LOAD_FAST    1 (r)
36     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?