

PFTT (plan for this Thursday)

- **What is a Python program?**
 - In the context of what we do in Compsci 101
 - In a neuroscience lab, at a web start-up, ...
 - What does "what is a program" even mean?
- **High-level and low-level Python constructs**
 - Variables and constants:
 - Names, types, and values
 - Operators and functions on Python types
- **Different approaches to code in Compsci101**

Compsci 101.2, Fall 2015

4.1

Start with Code Detective/Analysis

- **Use your skill, intuition, and deductive reasoning experience to answer questions about code that may be unfamiliar**

<http://bit.ly/101fall15-0903-1>

Compsci 101.2, Fall 2015

4.2

Results of Code Analysis

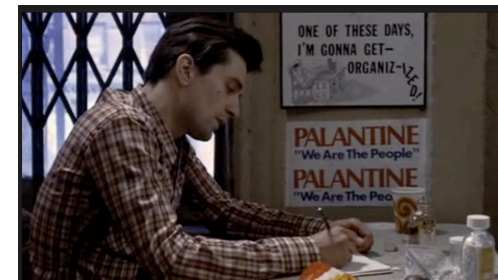
- **For details on plurals:** <http://bit.ly/1N49u6b>
- **How did we call pluralize many times?**
 - Loop. What is an alternative?
- **What does the 'if' statement do?**
 - Selects a code block to execute (more next week)
- **If you have a question? Write and run code!**

Compsci 101.2, Fall 2015

4.3

Organization matters

- <https://www.youtube.com/watch?v=1ve5713c19g>



Compsci 101.2, Fall 2015

4.4

APT organization, Code organization

- You've written the BMI.py APT
 - Where is that module? How do you test it?
 - PyDev console, but then must import it
 - Adding print statements in BMI.py to test
- Putting sentences together in order...
 - "Once upon a time..." "It was the best of times..." "Aujord'hui ma maman est morte"
- Putting code together in order
 - Takes judgment and experience

Writing Functions, Calling Functions

- After writing BMI.py, testing it (snarf)

➢ http://www.sutterhealth.org/health/bmi_calculator.html

```
import BMI

def getAdvice(name):
    print "hello",name,"how tall are you (in inches)?",
    inches = input()
    print "how much do you weigh (in pounds)",
    pounds = input()
    bmi = BMI.calculate(pounds,inches)

    if (bmi < 18.5):
        return "underweight"
    if (bmi < 24.9):
        return "healthy"
    if (bmi < 29.9):
        return "overweight"
    return "obese"
```

Examining Functions Closely

- Names of parameters in BMI.calculate?
 - What about order of parameters?
- Names of values passed to BMI.calculate?
 - Could be variables, constants: arguments
- Who wrote `math.sqrt(x)`?
 - What is name of parameter? Essential to call?
 - What is type of parameter? Essential to call?

Writing Code and Deploying Code

- You've written code to solve an APT
 - Written a .py module, how do you run it?
 - Use a Python interpreter, *must call function*
- The APT testing framework calls your code
 - Hollywood principle
 - "Don't call us, we'll call you"
 - https://en.wikipedia.org/wiki/Hollywood_principle
 - Like developing and using an API, someone writes the code, someone calls the code
 - `urllib2.urlopen` (<http://nytimes.com>)

Return to the Barnyard and Farm

- Back to an example from last time
 - Organizing code in a program
 - Refactoring code in a working program
- Once a program works, sometimes you're not done!
 - What does "it works" even mean?
 - What about version 2.0?
 - What about making it "better": perfect is the enemy of good. Good enough works!!!!

Toward creating functions

- New meets old
 - <https://www.youtube.com/watch?v=0IM-NyN06rA>

Old MacDonald had a farm, Ee-igh, Ee-igh, oh!
And on his farm he had a pig , Ee-igh, Ee-igh, oh!
With a oink oink here
And a oink oink there
Here a oink there a oink everywhere a oink oink
Old MacDonald had a farm, Ee-igh, Ee-igh, oh!

Creating Parameterized Function

What differs? Variable or Parameter

Old MacDonald had a farm, Ee-igh, Ee-igh, oh!
And on his farm he had a **horse**, Ee-igh, Ee-igh, oh!
With a **neigh neigh** here
And a **neigh** neigh there
Here a **neigh** there a **neigh** everywhere a **neigh neigh**
Old MacDonald had a farm, Ee-igh, Ee-igh, oh!

Old MacDonald had a farm, Ee-igh, Ee-igh, oh!
And on his farm he had a **pig** , Ee-igh, Ee-igh, oh!
With a **oink oink** here
And a **oink oink** there
Here a **oink** there a **oink** everywhere a **oink oink**
Old MacDonald had a farm, Ee-igh, Ee-igh, oh!

Abstracting over code: functions

- <http://goo.gl/DfcPgI>
- See snarf for class work as well
- These functions do not return values, they print
 - Illustrates problem decomposition, but ...
 - Normally have each function return a value
 - Normally use the return value in function call



Part of <http://goo.gl/DfcPgl> (and snarf)

```
def eieio():
    print "Ee-igh, Ee-igh, oh!"

def refrain():
    print "Old MacDonald had a farm,",
    eieio()

def had_a(animal):
    print "And on his farm he had a", animal, ", ",
    eieio()
```

Lots of commas

Anatomy and Dissection of Print

- Print generates output to a console, window, ...
 - Depends on how program invoked
 - Basically used for: help with debugging and creating output for copy/paste, view

```
print "hello", x, "what's up", y
```

- Space inserted between comma-separated items
 - Can use string concatenation, "hello"+str(x)
 - If print statement ends with comma, no newline
 - Print anything that has a string representation...

Abstraction over barnyards

- In OldMacPrint we have pig() and fox() ...
 - What's the same in these? What's different?
 - Capture differences in parameters/variables
- Create new function:
 - def verse(animal, noise)
- Look at pig() and fox() create new function
 - Call: verse("horse", "neigh")
 - Call: verse("cow", "moo")

<http://bit.ly/101fall15-901-2>

Vocabulary, grammar, rules: Python

- Naming
 - The power of abstraction and parameterization
 - What is abstraction?
 - What are parameters? What has them?
- 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: +, *, [], [:], [::]

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!



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.

Expressions, Operators, Types

- Why is `3+5*4` different than `(3+5)*4`?
 - 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!

Functions: abstractions over code

- **Naming something gives you power**
 - How do you read a file into a string?
 - What is length of a string? Of a list?
- **We can write and call functions**
 - Re-use and/or modify
 - Store in module, import and re-use functions
 - Import standard modules and use functions from them
- **Functions can (should?) return a value**
 - We've seen len return an int, what about file.read()?
 - Other functions return Strings, floats, or other types

Value Expert

- **Answer these questions**

<http://bit.ly/101fall15-0903-2>