Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Introduction to Python
Modules, Conditionals, Looping

Christopher Barker

UW Continuing Education / Isilon

June 27, 2012

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Table of Contents

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Review of Previous Class

- Values and Types
- Expressions
- Intro to functions
- Scopes and functions

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Homework review

Homework Questions?

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Functions as Objects

Passing a function as an argument

```
def simple():
    print "I'm simple"

def run_twice(fun):
    fun()
    fun()

run_twice( simple )
vs.
run_twice( simple() )
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Note about lectures

Hopefully, you are reading the book(s)

I'm not going to be comprehensive–

I'll focus on overview and "gotchas"

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Notes on reading

Checking for type – factorial example

tuple unpacking in function arguments –

Pythagoras example

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Code Structure

Python is all about namespaces – the "dots"

```
name.another_name
```

the "dot" indicates looking for a name in the namespace of the given object.
could be:

- name in a module
- module in a package
- attribute of an object
- method of an object

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

indenting and blocks

Indenting determines blocks of code

```
something:
    some code
    some more code
    another block:
        code in
        that block
```

But you need the colon too...

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

indenting and blocks

You can put a one-liner after the colon:

```
In [167]: x = 12

In [168]: if x > 4: print x
12
```

Only do this if it makes it more readable...

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

spaces and tabs

An indent can be:

- Any number of spaces
- A tab
- tabs and spaces:
  - A tab is eight spaces (always!)
  - Are they eight in your editor?

Use four spaces – really!

(PEP 8)

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Various Brackets

Bracket types:

- parentheses ( )
    - tuple literal: (1,2,3)
    - function call: fun(arg1, arg2)
    - grouping: (a + b) * c
- square brackets [ ]
    - list literal: [1,2,3]
    - sequence indexing: a_string[4]
- curly brackets { }
    - dictionary literal: {"this":3, "that":6}
    - (we'll get to those...)

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

tuples and commas..

Tuples don't NEED parentheses...

```
In [161]: t = (1,2,3)
In [162]: t
Out[162]: (1, 2, 3)

In [163]: t = 1,2,3
In [164]: t
Out[164]: (1, 2, 3)

In [165]: type(t)
Out[165]: tuple
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

tuples and commas..

Tuples do need commas...

```
In [156]: t = ( 3 )

In [157]: type(t)
Out[157]: int

In [158]: t = (3,)
In [159]: t
Out[159]: (3,)

In [160]: type(t)
Out[160]: tuple
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

modules and packages

A module is simply a namespace

A package is a module with other modules in it

The code in the module is run when it is imported

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

importing modules

```
import modulename

from modulename import this, that

import modulename as a_new_name
```

(demo)

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

importing from packages

```
import packagename.modulename

from packagename.modulename import this, that

from package import modulename
```

(demo)

http://effbot.org/zone/import-confusion.htm

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

importing from packages

```
from modulename import *
```

Don't do this!

("Namespaces are one honking great idea...")

(wxPython and numpy example...)

Except *maybe* math module

(demo)

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

import

If you dont know the module name before execution.

`__import__(module)`

where `module` is a Python string.

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

modules and packages

The code in a module is NOT re-run when imported
again – it must be explicitly reloaded to be re-run

```
import modulename
```

```
reload(modulename)
```

(demo)

```
import sys
print sys.modules
```

(demo)

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

LAB

Experiment with importing different ways:

```
import math
dir(math) # or, in ipython -- math.<tab>
math.sqrt(4)

import math as m
m.sqrt(4)

from math import *
sqrt(4)
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

LAB

Experiment with importing different ways:

```
import sys
print sys.path

import os
print os.path
```

You wouldn't want to import * those – check out

```
os.path.split()
os.path.join()
```

Lightning Talk

Lightning Talk: Cliff

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Truthiness

What is true or false in Python?

- The Booleans: `True` and `False`
- "Something or Nothing"

`http://mail.python.org/pipermail/python-dev/2002-April/`
`022107.html`

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Truthiness

determining Truthiness:

`bool(something)`

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Boolean Expressions

False

- None
- False
- zero of any numeric type, for example, 0, 0L, 0.0, 0j.
- any empty sequence, for example, '', (), [] .
- any empty mapping, for example, {}.
- instances of user-defined classes, if the class defines a
  __nonzero__() or __len__() method, when that method
  returns the integer zero or bool value False.

http://docs.python.org/library/stdtypes.html

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Boolean Expressions

avoid

```
if xx == True:
```

use

```
if xx:
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Boolean Expressions

"Shortcutting"

```
                  if x is false,
x or y                return y,
                      else return x


                  if x is false,
x and y               return  x
                      else return y


                  if x is false,
not x                 return True,
                      else return False
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Boolean Expressions

Stringing them together

a or b or c or d

a and b and c and d

The first value that defines the result is returned

(demo)

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Boolean returns

From CodingBat

```
def sleep_in(weekday, vacation):
    if weekday == True and vacation == False:
        return False
    else:
        return True
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Boolean returns

From CodingBat

```
def sleep_in(weekday, vacation):
    return not (weekday == True and vacation == False)
```

or

```
def sleep_in(weekday, vacation):
    return (not weekday) or vacation
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

bools are ints?

bool types are subclasses of integer

```
In [1]: True == 1
Out[1]: True

In [2]: False == 0
Out[2]: True
```

It gets weirder!

```
In [6]: 3 + True
Out[6]: 4
```

(demo)

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

LAB

re-write a couple CodingBat exercises, returning the direct
boolean results

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

if

Making Decisions...

```
if a:
    print 'a'
elif b:
    print 'b'
elif c:
    print 'c'
else:
    print 'that was unexpected'
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

if

Making Decisions...

```
if a:
    print 'a'
elif b:
    print 'b'

## versus...

if a:
    print 'a'
if b:
    print 'b'
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

switch?

No switch/case in Python

use use `if..elif..elif..else`

(or a dictionary, or subclassing....)

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Sequences

Sequences are ordered collections

They can be indexed, sliced, iterated over,...

They have a length: `len(sequence)`

Common sequences (Duck Typing)

- strings
- tuples
- lists

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Indexing

square brackets for indexing: []

Indexing starts at zero

```
In [98]: s = "this is a string"

In [99]: s[0]
Out[99]: 't'

In [100]: s[5]
Out[100]: 'i'
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Indexing

Negative indexes count from the end

```
In [105]: s = "this is a string"

In [106]: s[-1]
Out[106]: 'g'

In [107]: s[-6]
Out[107]: 's'
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Slices

Slicing: Pulling a range out of a sequence

```
sequence[start:finish]
```

indexes for which:

```
start <= i < finish
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Slices

```
In [121]: s = "a bunch of words"
In [122]: s[2]
Out[122]: 'b'

In [123]: s[6]
Out[123]: 'h'

In [124]: s[2:6]
Out[124]: 'bunc'

In [125]: s[2:7]
Out[125]: 'bunch'
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Slices

the indexes point to the spaces between the items

```
X   X   X   X   X   X   X   X
|   |   |   |   |   |   |   |
0   1   2   3   4   5   6   7
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Slices

Slicing satisfies nifty properties:

```
len( seq[a:b] ) == b - a
```

```
seq[a:b] + seq [b:c] == seq
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Slicing vs. Indexing

Indexing returns a single element

```
In [86]: l
Out[86]: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

In [87]: type(l)
Out[87]: list

In [88]: l[3]
Out[88]: 3

In [89]: type( l[3] )
Out[89]: int
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Slicing vs. Indexing

Unless it's a string:

```
In [75]: s = "a string"

In [76]: s[3]
Out[76]: 't'

In [77]: type(s[3])
Out[77]: str
```

there is no single character type

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Slicing vs. Indexing

Slicing returns a sequence:

```
In [68]: l
Out[68]: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

In [69]: l[2:4]
Out[69]: [2, 3]
```

Even if it's one element long

```
In [70]: l[2:3]
Out[70]: [2]

In [71]: type(l[2:3])
Out[71]: list
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Slicing vs. Indexing

Indexing out of range produces an error

```
In [129]: s = "a bunch of words"
In [130]: s[17]
----> 1 s[17]
IndexError: string index out of range
```

Slicing just gives you what's there

```
In [131]: s[10:20]
Out[131]: ' words'

In [132]: s[20:30]
Out[132]: ''
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Multiplying and slicing

from CodingBat: Warmup-1 – front3

```python
def front3(str):
  if len(str) < 3:
    return str+str+str
  else:
    return str[:3]+str[:3]+str[:3]
```

or

```python
def front3(str):
    return str[:3] * 3
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Slicing

from CodingBat: Warmup-1 – `missing_char`

```
def missing_char(str, n):
  front = str[0:n]
  l = len(str)-1
  back = str[n+1:l+1]
  return front + back

def missing_char(str, n):
    return str[:n] + str[n+1:]
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

LAB

- stuff with sequences and slicing

http://www.upriss.org.uk/python/PythonCourse.html

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Lightning Talk

Lightning Talk: Myke

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

for loops

looping through sequences

```
for x in sequence:
    do_something_with_x
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

for loops

```
In [170]: for x in "a string":
   .....:         print x
   .....:
a

s
t
r
i
n
g
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

range

looping a known number of times..

```
In [171]: for i in range(5):
    .....:     print i
    .....:
0
1
2
3
4
```

(you don't need to do anything with i...

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

range

range defined similarly to indexing

```
In [183]: range(4)
Out[183]: [0, 1, 2, 3]

In [184]: range(2,4)
Out[184]: [2, 3]

In [185]: range(2,10,2)
Out[185]: [2, 4, 6, 8]
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

indexing?

Python only loops through a sequence – not like C,
Javascript, etc...

```
for(var i=0; i<arr.length; i++) {
    var value = arr[i];
    alert(i =") "+value);
}
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

indexing?

Use range?

```
In [193]: letters = "Python"

In [194]: for i in range(len(letters)):
    .....:      print letters[i]
    .....:
P
y
t
h
o
n
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

indexing?

More Pythonic – for loops through sequences

```
In [196]: for l in letters:
    .....:     print l
    .....:
P
y
t
h
o
n
```

Never index in normal cases

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

enumerate

If you need an index – `enumerate`

```
In [197]: for i, l in enumerate(letters):
   .....:     print i, l
   .....:
0 P
1 y
2 t
3 h
4 o
5 n
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

multiple sequences – zip

If you need to loop though parallel sequences – zip

```
In [200]: first_names = ['Fred', 'Mary', 'Jane']

In [201]: last_names = ['Baker', 'Jones', 'Miller']

In [203]: for first, last in zip(first_names, last_names):
   .....:     print first, last
   .....:
Fred Baker
Mary Jones
Jane Miller
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

xrange

range creates the whole list

xrange is a generator – creates it as it's needed –

a good idea for large numbers

```
In [207]: for i in xrange(3):
   .....:     print i
0
1
2

(Python 3 – range == xrange)
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

for

`for` does NOT create a name space:

```
In [172]: x = 10

In [173]: for x in range(3):
   .....:     pass
   .....:

In [174]: x
Out[174]: 2
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

LAB

- Look up the % operator. What do these do?
  10 % 7 == 3
  14 % 7 == 0

- Write a program that prints the numbers from 1 to 100 inclusive. But for multiples of three print "Fizz" instead of the number and for the multiples of five print "Buzz". For numbers which are multiples of both three and five print "FizzBuzz" instead.

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

while

`while` is for when you don't know how many loops
you need

Continues to execute the body until condition is not
True

```
while a_condition:
    some_code
    in_the_body
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

while

while is more general than for – you can always
express for as while, but not always vice-versa.

while is more error-prone – requires some care to
terminate

loop body must make progress, so condition can
become False

potential error: infinite loops

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

while vs. for

```
letters = 'Python'
i=0
while i < len(letters):
    print letters[i]
    i += 1

vs.

letters = 'Python'
for c in letters:
    print c
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

while

Shortcut: recall – 0 or empty sequence is `False`

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

break

break ends a loop early

```
x = 0
while True:
    print x
    if x > 3:
        break
    x = x + 1
In [216]: run for_while.py
0
1
2
3
4
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

break

same way with a `for` loop

```
name = "Chris Barker"
for c in name:
    print c,
    if c == "B":
        break
print "I'm done"

C h r i s   B
I'm done
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

continue

continue skips to the start of the loop again

```
print "continue in a for loop"
name = "Chris Barker"
for c in name:
    if c == "B":
        continue
    print c,
print "\nI'm done"

continue in a for loop
C h r i s   a r k e r
I'm done
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

continue

`continue` works for a `while` loop too.

```
print "continue in a while loop"
x = 6
while x > 0:
    x = x-1
    if x%2:
        continue
    print x,
print "\nI'm done"

continue in a while loop
4 2 0
I'm done
```

Christopher Barker   Intro to Python: Week 2

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

else again

else block run if the loop finished naturally – no
break

```
print "else in a for loop"
x = 5
for i in range(5):
    print i
    if i == x:
        break
else:
    print "else block run"
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Command Line Input

```
input and raw_input
input evaluates the input
In [265]: val = input("a message> ")
a message> 4.5

In [266]: type(val)
Out[266]: float
```

raw_input gives you the plain string

```
In [265]: val = input("a message> ")
a message> 4.5

In [266]: type(val)
Out[266]: float
```

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

LAB

```
def count_them(letter):
```

- prompts the user to input a letter
- counts the number of times the given letter is input
- prompts the user for another letter
- continues until the user inputs "x"
- returns the count of the letter input

```
def count_letter_in_string(string, letter):
```

- counts the number of instances of the letter in the string
- ends when a period is encountered
- if no period is encountered – prints "hey, there was no period!"

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

example

Example: `re-run-latex.py` script

Review/Questions
Python Code Structure
Modules and packages
Boolean Expressions
Conditionals
Sequences
Looping

Homework

-