

A Little Python – Part 2

Introducing Programming with Python

Data Structures, Program Control

Outline

- Python and the System
- Data Structures
 - Lists, Dictionaries
- Control Flow
 - if, for, while

Python Path

- Python environment variable
 - Where to find python packages
- PYTHONPATH
 - cshell
 - `setenv PYTHONPATH "/home/dwmc/development/python"`
 - bash
 - `PYTHONPATH="/home/dwmc/development/python"`
 - `export PYTHONPATH`

Python Modules

- System or Distribution Modules
 - Come pre-installed
 - Installed with `easy_install` or `pip`
- User modules
 - Python searches the PYTHONPATH directories
 - In the order specified
 - Module generally a directory name, have `__init__.py` file

Import Python Modules

- Access a module with the “import” command

- Seen an example of this last time

- Import

```
>>> import math
>>> import random
>>> import aflag
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ImportError: No module named aflag
```

- Importing user modules is the same.

Import Variant

- Import treats package variables, methods

```
import random
print random.randint(3,100)
```

- There is a way to import using a “from” clause

```
from random import randint
print randint(3,100)
```

- There are different ways to use “from”

Some Important Modules

<code>import sys</code>	Access to system features
<code>import os</code>	Python operating system hooks
<code>import urllib</code>	Manipulate URLs, web access
<code>import random</code>	Random number generator

Data Structures

- Lists
- Dictionaries
- Tuples (not covering these)

Data Structures - Lists

- List a collection of arbitrary items

```
l = []
l = [1, 'two', 3, 4]
l = ['a', ['b', 'c']]
l[i]
l[j][k]
l[x:y]
l.append('abc'), l.sort(), l.pop(), l.remove(3)
del l[k]
l[2] = 'abc'
```

- Try a few of these

Example - lists

```
>>> l1 = []
>>> l2 = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
>>> print l1
[]
>>> print l2
['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
>>> print l2[2]
c
>>> print l2[4:7]
['e', 'f', 'g']
>>> print l2[:3]
['a', 'b', 'c']
>>> print l2[7:]
['h', 'i', 'j']
```

Example - lists

```
>>> l1 = []
>>> l2 = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
>>> print l1
[]
>>> print l2
['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
>>> del l2[5]
>>> print l2
['a', 'b', 'c', 'd', 'e', 'g', 'h', 'i', 'j']
>>> print l2.pop()
j
>>> print l2
['a', 'b', 'c', 'd', 'e', 'g', 'h', 'i']
>>> l2.append(45)
>>> print l2
['a', 'b', 'c', 'd', 'e', 'g', 'h', 'i', 45]
>>> l2.append(l1)
>>> print l2
['a', 'b', 'c', 'd', 'e', 'g', 'h', 'i', 45, []]
>>> print len(l2)
10
```

Data Structures - Dictionaries

- Key/Value stores, arbitrary items
 - Keys are immutable, but values can be changed

```
d = {}
d = {'size':4, 'name':"bob", 'list':[1,2,3], 'dict':{}}
d['size']
d['list'][0]
d.keys()
d.values()
del d['name']
```

- Try a few of these

Example - dictionaries

```
>>> d2 = {'size':4, 'name':"bob", 'list':[1,2,3,'a','b'], 'dict':
{'name':"booboo", 'value':123}}
>>> print d2
{'dict': {'name': 'booboo', 'value': 123}, 'list': [1, 2, 3, 'a', 'b'], 'name':
'bob', 'size': 4}
>>> print d2['size']
4
>>> print d2['list'][0]
1
>>> print d2['dict']['name']
booboo
>>> print d2.keys()
['dict', 'list', 'name', 'size']
>>> print d2.values()
[{'name': 'booboo', 'value': 123}, [1, 2, 3, 'a', 'b'], 'bob', 4]
>>> d2['gob']="This is a string."
>>> print d2
{'dict': {'name': 'booboo', 'value': 123}, 'gob': 'This is a string.', 'list':
[1, 2, 3, 'a', 'b'], 'name': 'bob', 'size': 4}
>>> print len(d2)
5
```

Control Flow

- Conditional Tests
 - If
- Looping
 - while
 - for

Conditional - if

■ Conditional Branch

```
if <test_condition>:  
    statements  
elif <test_condition>:  
    statements  
else:  
    statements
```

Example - if

```
def iftest(goo):  
    if goo == 1:  
        print "Found a digit"  
    elif goo=="one":  
        print "Found one string"  
    else:  
        print "Not sure"  
  
>>> goop = "one"  
>>> iftest(goop)  
Found one string  
>>> iftest(1)  
Found a digit  
>>> iftest("two")  
Not sure  
>>> iftest(one)  
Traceback (most recent call last):  
  File "<stdin>", line 1, in <module>  
NameError: name 'one' is not defined
```


Looping - while

- Conditional Loop

```
while <test_condition>:  
    statements
```

- Special looping statements

```
break  
continue
```

Example - while

```
def loopTest1(c=0,b=100):  
    i = 0  
    while i<c:  
        print i  
        if i==b:  
            print "break"  
            break  
        i += 1  
  
>>> loopTest(5)  
0  
1  
2  
3  
4  
>>> loopTest(5,2)  
0  
1  
2  
break
```

Looping - for

- Iterator looping

```
for <target> in <object>:  
    statement
```

- Special looping statements

```
break  
continue
```

- Particularly good for iterating through lists

Example - for

```
l2 = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']  
  
def loopTest2(l=[]):  
    if l:  
        for item in l:  
            print "Got item: \"%s\""%(str(item))  
    else:  
        print "List parameter was empty"
```

Example - for

```
>>> print l2
['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
>>> loopTest2()
List parameter was empty
>>> loopTest2(l2)
Got item: "a"
Got item: "b"
Got item: "c"
Got item: "d"
Got item: "e"
Got item: "f"
Got item: "g"
Got item: "h"
Got item: "i"
Got item: "j"
```

Assignment

- Write 2 short programs
 1. Write a procedure that accepts one parameter (count) and generates a list of (count) random integers, between 0 and 1000, and puts those integers into a list, and returns the list.
 2. Write a procedure that takes four parameters (lastname, firstname, score, grade) and returns a new dictionary item with those four items.