

More on Functions; Some File I/O

UW CSE 190p

Summer 2012

Review

```
def myfunc(n):  
    total = 0  
    for i in range(n):  
        total = total + i  
    return total  
  
print myfunc(4)
```

What is the result?

Exercise: Control Flow

```
def c_to_f(c):  
    print "c_to_f"  
    return c / 5.0 * 9 + 32
```

```
def make_message(temp):  
    print "make_message"  
    return "The temperature is "+str(temp)
```

```
for tempc in [19,22,21]:  
    tempf = c_to_f(tempc)  
    message = make_message(tempf)  
    print message
```

```
c_to_f  
make_message  
The temperature is 66.2  
c_to_f  
make_message  
The temperature is 71.6  
c_to_f  
make_message  
The temperature is 69.8
```

Loop Gotcha: Indentation

```
def c_to_f(c):  
    print "c_to_f"  
    return c / 5.0 * 9 + 32
```

```
def make_message(temp):  
    print "make_message"  
    return "The temperature is "+str(temp)
```

```
for tempc in [19,22,21]:  
    tempf = c_to_f(tempc)  
    message = make_message(tempf)  
    print message
```

```
c_to_f  
make_message  
c_to_f  
make_message  
c_to_f  
make_message  
The temperature is 69.8
```

Function Gotcha: Local Variables

```
def c_to_f(c):  
    result = c / 5.0 * 9 + 32  
    return result  
  
tempf = c_to_f(19)  
print result
```

A mistake! What happens here?

See examples in the book:

<http://thinkcspy.appspot.com/build/functions.html#variables-and-parameters-are-local>

Local variables

```
def c_to_f(c):  
    result = c / 5.0 * 9 + 32  
    return result  
  
tempf = c_to_f(19)  
print result
```

Local scope

Global scope

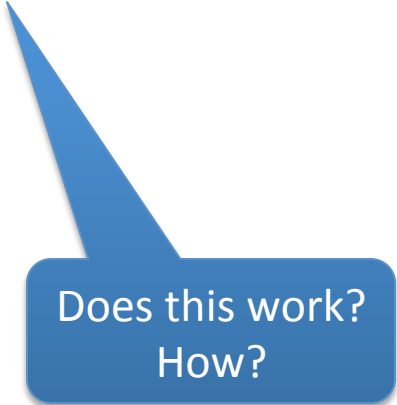
English definition:

scope (n): extent or range of view, outlook, application, operation, effectiveness, etc.:

Local Scope and Global Scope

```
def normal_pressure (pressure) :  
    result = pressure - pressure_at_sea_level  
    return result
```

```
pressure_at_sea_level = 7  
print normal_pressure (16)
```



Does this work?
How?

Local Scope and Global Scope

- When Python encounters a variable, it
 - first checks to see if the variable is defined in the **local scope**
 - then checks to see if the variable is defined in the **global scope**

But: try to avoid this feature.

Keep global variables global, and local variables local

Local Scope and Global Scope

```
def normal_pressure(pressure) :  
    pressure_at_sea_level = 7  
    result = pressure - pressure_at_sea_level  
    return result  
  
print normal_pressure(16)
```

Better

Confusing Variables

```
def atm_to_mbar (pressure) :  
    return pressure * 1013.25
```

```
def mbar_to_mmHg (pressure) :  
    return pressure * 0.75006
```

```
pressure = 1.2 # in atmospheres  
pressure = atm_to_mbar (pressure)  
pressure = mbar_to_mmHg (pressure)  
print pressure
```

A Better Way

```
def atm_to_mbar(pressure):  
    return pressure * 1013.25
```

```
def mbar_to_mmHg(pressure):  
    return pressure * 0.75006
```

```
in_atm = 1.2  
in_mbar = atm_to_mbar(in_atm)  
in_mmHg = mbar_to_mmHg(in_mbar)  
print in_mmHg
```

Much more clear!!

Function Gotcha: Return Value

```
def c_to_f(c):  
    print c / 5.0 * 9 + 32
```

No return value!?!

```
print c_to_f(19)
```

Anything wrong here?

*Good practice: Always include a
return statement!*

Q: Can functions call other functions?

```
def atm_to_mbar(pressure):  
    return pressure * 1013.25  
  
def mbar_to_mmHg(pressure):  
    return pressure * 0.75006  
  
def atm_to_mmHg(pressure):  
    in_mbar = atm_to_mbar(pressure)  
    in_mmHg = mbar_to_mmHg(in_mbar)  
    return in_mmHg  
  
print atm_to_mmHg(5)
```

Local Scope and Global Scope revisited

- When Python encounters a variable, it
 - first checks to see if the variable is defined in the **local scope**
 - then checks to see if the variable is defined in the **next most outer scope**
 - then checks to see if the variable is defined in the **next most outer scope**
 - ...
 - then checks to see if the variable is defined in the **global scope**

Function Engineering

- Breaking down a program into functions is the fundamental activity of programming!
- How do you decide when to use a function?
 - One rule from the last lecture: DRY
 - Whenever you are tempted to copy and past code, don't!
- Now how do you design a function?

Function Engineering

- How do you design a function?
- Step 1: Write the program as if the function you want already existed

```
print "This is the temperature in Farenheit: ", tempf
tempc = f_to_c(tempf)
print "This is the temperature in Celsius: ", tempc
```

Always start this way!

Function Engineering

- How do you design a function?
- Step 2: Describe the inputs and output. Refer to their type.

```
# Inputs: a number representing degrees fahrenheit  
# Return value: a number representing degrees celsius
```

Function Engineering

- How do you design a function?
- Step 3: Implement the function

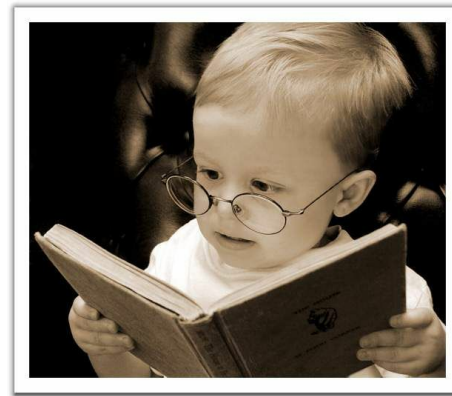
```
def f_to_c(num):  
    result = (f - 32) / 9.0 * 5  
    return result
```

File Input and Output

- As a programmer, when would one use a file?
- As a programmer, what does one do with a file?

Important operations

- open a file
- close a file
- read data
- write data



Read a file in python

```
myfile = open("datafile.dat", "r")  
  
for line_of_text in myfile:  
    print line_of_text
```



historical
convention

Assumption: file is a sequence of lines

File Gotcha: where does Python expect to find this file?

“Current Working Directory”

Documents

Name	Date Modified	Size	Kind
▶ EarthCube	Jan 18, 2012, 5:09 PM	--	Folder
▶ SIGMO2012_demopc	Jan 15, 2012, 4:12 PM	--	Folder
▼ Courses	Jan 13, 2012, 1:14 PM	--	Folder
▼ CSSE	Today, 2:27 PM	--	Folder
▼ wiki	Today, 2:44 PM	--	Folder
▶ lectures	Today, 2:45 PM	--	Folder
Schedule.wiki	Jun 18, 2012, 12:19 PM	4 KB	Document
▼ assignments	Jun 18, 2012, 2:52 AM	--	Folder
▶ twitter	Jun 18, 2012, 11:45 AM	--	Folder
▶ benfords-law	Jun 18, 2012, 3:03 AM	--	Folder
▶ assignment1	Jun 18, 2012, 3:03 AM	--	Folder
▶ twitter_scratch	Jun 18, 2012, 2:53 AM	--	Folder
▶ hw1	Jun 18, 2012, 2:49 AM	--	Folder
▶ social-network	Jun 17, 2012, 9:32 PM	--	Folder
Ideas.wiki	Jun 17, 2012, 9:32 PM	1...B	Document
▶ treatmentefficacy	Jun 17, 2012, 8:58 PM	--	Folder
▶ prochronisms	Jun 15, 2012, 11:34 AM	--	Folder
Makefile	Jun 13, 2012, 11:17 PM	4 KB	Plain text
▶ microarray	May 17, 2012, 6:56 PM	--	Folder
▶ assignment3	May 9, 2012, 11:17 PM	--	Folder
▶ assignment2	Apr 8, 2012, 1:39 PM	--	Folder
notes.txt	Jun 17, 2012, 9:32 PM	4 KB	Smult...ument
▶ handouts	Jun 17, 2012, 9:32 PM	--	Folder

346 items, 21.31 GB available

“Current Working Directory”

The screenshot shows a Windows Explorer window with the address bar displaying the path: Libraries > Documents > CloudLecture > CloudCertificate. The search bar contains the text "Search CloudCertificate". The left sidebar shows the "Documents" library selected. The main pane displays the "Documents library" for "CloudCertificate" with the following table:

Name	Date modified	Type	Size
big_data_cloud_syllabus.doc	4/2/2012 10:14 AM	Microsoft Office ...	51 KB
big_data_cloud_syllabus_old.doc	4/2/2012 10:11 AM	Microsoft Office ...	51 KB
Course topics.xls	12/4/2010 1:14 PM	Microsoft Office E...	19 KB

The status bar at the bottom indicates "3 items".

“Current Working Directory” in Python

```
>>> import os    # “os” stands for “operating system”  
>>> os.getcwd()  
'/Users/billhowe/Documents'
```

This point is minor, but can be the source of confusion and bugs.

A reasonable practice is just to work from the default location.

On my systems:

Windows: 'C:\\Python27'

Mac: '/Users/billhowe/Documents'

Read a file in python

```
myfile = open("datafile.dat", "r")
```

```
all_data_as_a_big_string = myfile.read()
```

Another way to read data

Write to a file in python

```
myfile = open("output.dat", "w")
```

```
myfile.write("a bunch of data")
```



historical
convention

Write to a file in python

```
myfile = open("output.dat", "w")
```

```
myfile.write(4)
```



historical
convention

```
Traceback (most recent call last):
```

```
  File "writefile.py", line 3, in <module>
```

```
    myfile.write(4)
```

```
TypeError: expected a character buffer object
```

```
myfile.write(str(4))
```

Write to a file in python

```
myfile = open("output.dat", "w")
```

```
myfile.write("a line of text\n")
```

Use "\n" to indicate the end of a line