

Python Basics!

scripting, logic, control

CS101 Lecture #6

Administrivia

- ▣ Homework #2 is due Wed Oct. 19.
- ▣ Homework #3 is due Wed Oct. 26.

Warmup Question (No Quiz this lecture!)

Question #1

```
s = "74.125.21.147"  
i = s.find( "." )  
x = s[i+1:i+3]  
x = x * 2
```

What is the value of x?

- A "125125"
- B 250
- C "1212"
- D 24

Composing Functions

Example: Defining functions

```
def pow(a, b):  
    y = a ** b  
    return y
```

Defining functions

- ❖ We define a function with the following:
 - ❑ the keyword `def`
 - ❑ the name of the function
 - ❑ a pair of parentheses
 - ❑ a **block** of code

Example: Defining functions

```
def greetings():  
    print("Bom dia!")  
    print("Bonjour!")  
    print("Ni hao!")  
    print("Hello!")  
    print("Shalom!")  
    print("Guten tag!")  
    print("Konichiwa!")  
    print("As-salamu alaykum!")
```

header

body

Block

- ❖ A section of code grouped together.
- ❖ Begins with a `:`.
- ❖ Contents of the block are indented:

```
def hello():  
    print('hello')
```

- ❖ Variables defined inside of a block are *independent* of variables outside of the block.
- ❖ Variables inside a block *do not exist* outside of the block.
- ❖ Blocks are isolated from the rest of the code!

Example: Defining functions

```
a = 5
def fun():
    a = 3
    b = 4
    a = a + b
fun()
print(a)
```

return

- ❖ Functions can return values with the keyword `return`.

```
def three():  
    return 3
```

- ❖

```
def zero():  
    return 0  
    print('0')
```

- ❖ Does the above code face an error?
Does the `print` statement take effect if invoking the function?

Parameters

- ❖ Functions can accept values as parameters (input, arguments).
- ❖ These variables are declared in the function header.
- ❖ Multiple parameters are separated by commas.

```
def print_message( msg ):  
    print( msg )
```

Example

```
def fun(a):  
    return a+2
```

```
x = fun(2) * fun(3)
```

What is the value of x?

A 6

B 8

C 24

D None of the above.

Example

```
def fun(m):  
    return m.title().swapcase()  
  
x = fun( "abb") + fun( "acab" )
```

What is the value of x?

- A 'AbbAcab'
- B 'aBBaCAB'
- C 'abbacab'
- D 'ABBACAB'

Example

```
def fun(a,b):  
    c = ((a + ' ') * len(b)).title()  
  
x = fun( "ab", "caa" )
```

What is the value of x?

- A 'ab ab ab '
- B 'Ab Ab Ab '
- C 'AB AB AB '
- D None of the above.

Example

```
def fun(a,b):  
    c = ((a + ' ') * len(b)).title()  
    return c
```

```
x = fun( "ab", "caa" )
```

What is the value of x?

- A 'ab ab ab '
- B 'Ab Ab Ab '
- C 'AB AB AB '

Boolean Logic

Boolean

- ❖ `bool` is a type with two possible values:
 - ❑ `True`
 - ❑ `False`
- ❖ We use these to make decisions.
- ❖ Their logic is based on Boolean algebra.
- ❖ Operators:
 - ❑ `and`
 - ❑ `or`
 - ❑ `not`

Example: Boolean logic

$$0 < x \leq 10$$

$$(x > 0) \text{ and } (x \leq 10)$$

Boolean operators

and	True	False
True	True	False
False	False	False

True when BOTH
inputs are true

or	True	False
True	True	True
False	True	False

True when EITHER
input is true

Boolean operators

	Result
not True	False
not False	True

Inverts truth-value

Example

```
def fun():  
    return True and False
```

```
x = fun() and not (True or False)
```

What is the value of x?

A True

B False

Comparison operators

- ❖ These produce Boolean output.
 - ❑ less than, <
 - ❑ greater than, >
 - ❑ less than or equal to, <=
 - ❑ greater than or equal to, >=
 - ❑ equal to, ==
 - ❑ not equal to, !=

Example

a = 5

b = 3

x = (a < 5) and ((b <= 5) or (a != b))

What is the value of x?

A True

B False

Example

```
a = 'URSA MAJOR'  
b = 'GEMINI'
```

```
x = a < b and a[1] != b[-2]
```

What is the value of x?

A True

B False

Example

```
def fun(a,b):  
    return a<b  
a = 3  
b = 4  
x = fun(b,a)
```

What is the value of x?

A True

B False

Conditional Execution

- ❖ *Control flow* represents actual sequence of lines executed by processor.
- ❖ *Conditional execution* lets you execute (or not) a block of code based on logical comparison.

Example: *if* statement

```
ans = input( "Enter a number:" )  
if float(ans) < 0:  
    print( "The number was negative." )
```

if statement

- ❖ We create an `if` statement as follows:
 - ❑ the keyword `if`
 - ❑ a logical comparison (results in `bool`)
 - ❑ a **block** of code

Alternative execution

- ❖ This lets us make decisions in the program!
- ❖ We can change program behavior as it executes.

Example: *if* statements

```
ans = input( "Enter a number:" )
if float(ans) < 0:
    print( "The number was negative." )
if float(ans) > 0:
    print( "The number was positive." )
if float(ans) == 0:
    print( "The number was zero." )
```

Reminders

Reminders

- ❖ Homework #2 is due Wed Oct. 19.
- ❖ Homework #3 is due Wed Oct. 26.