Lecture 4: Defining Functions (Ch. 3.4 - 3.11)

CS 1110
Introduction to Computing Using Python

[E. Andersen, A. Bracy, D. Gries, L. Lee, S. Marschner, C. Van Loan, W. White]

Things to Do Before Next Class

Readings:
• Sections 8.1, 8.2, 8.4, 8.5, first paragraph of 8.9

Labs:
• Go to Lab! (Lab 2 is this week)
• Get Credit for Lab 1:
  • can be checked off during Tuesday’s consulting hours 4:30-9:30 in the ACCEL lab
  • cannot be checked off after 3:45pm Wednesday
  • check online if you received credit:

Anatomy of a Function Definition

<table>
<thead>
<tr>
<th>name</th>
<th>parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>def increment(n):</td>
<td>Function Header</td>
</tr>
<tr>
<td>Returns: the value of $n+1$</td>
<td>Docstring</td>
</tr>
<tr>
<td>return $n+1$</td>
<td>Statements to execute when called also called Function Body</td>
</tr>
<tr>
<td>The vertical line indicates indentation</td>
<td>Use vertical lines when you write Python on exams so we can see indentation</td>
</tr>
</tbody>
</table>

The return Statement

• Passes a value from the function to the caller
• Format: `return <expression>`
• Any statements after `return` are ignored
• Optional (if absent, no value will be sent back)

Understanding How Functions Work

• We will draw pictures to show what is in memory
• Function Frame: Representation of function call
  - Draw parameters as variables (named boxes)
  - Number of statement in the function body to execute next
  - Starts with 1

Example: `get_feet(68)`

PHASE 1: Set up call frame
1. Draw a frame for the call
2. Assign the argument value to the parameter (in frame)
3. Indicate next line to execute

```
def get_feet(height_in_inches):
    return height_in_inches // INCHES_PER_FOOT
```
Example: get_feet(68)

PHASE 2:
Execute function body

The return terminates;
no next line to execute

def get_feet(height_in_inches):
    return height_in_inches // INCHES_PER_FOOT

PHASE 3: Erase call frame

But don't actually erase on an exam

def get_feet(height_in_inches):
    return height_in_inches // INCHES_PER_FOOT

Local Variables (4)

• Call frames can make “local” variables
  >>> import room_numbers
  >>> room_numbers.lab_rooms()

    def lab_rooms():
        red_room = 235
        orange_room = 236

        Variables are gone! This function is useless.

Function Access to Global Space

• All function definitions are in some module
• Call can access global space for that module
  • math.cos: global for math
  • height.get_feet uses global for height
• But cannot change values
  • Assignment to a global makes a new local variable!

Call Frames and Global Variables

Global Variables

Call Frame

def swap(a,b):
    """Swap global a & b""
    tmp = a
    a = b
    b = tmp

>>> a = 1
>>> b = 2
>>> swap(a,b)

# scope_example.py
"""Show how global work""
# global space
def get_a():
    return a # returns global

Function Access to Global Space

• All function definitions are in some module
• Call can access global space for that module
  • math.cos: global for math
  • height.get_feet uses global for height
• But cannot change values
  • Assignment to a global makes a new local variable!