CS1110
Lecture 7: More on function calls; if-then-else

Announcements
No office hours this week:
They've been replaced by the scheduled one-on-ones.

Readings:
Today: 3.9-3.10 (note that our notation differs slightly) and 5.1-5.7
Next time: 10.0-10.2 and 10.4-10.6

Q: Why is it important to understand the notation for and mechanics of variables, objects, and frames?
A: You get a clear model of what’s going on with Python, and, often, what’s gone wrong with your code when you have a bug.

1. How evaluate a function call expression, reformatted slightly:
   Use: Create a frame for the call
   Dos: Assign arguments to parameters
   (a) For each parameter ("the names in parentheses in the function header"), put a variable with that name in the frame
   (b) Evaluate the arguments ("the values of the stuff in parentheses in the function call")
   (c) Put the argument values in the corresponding parameter variables in the frame.
   Tris: Execute function body, updating the frame’s program counter (line number) as you go
   Quatro: Erase Cross out the frame
   The value of the function call expression is the returned value (if there is one)

2. Conditionals: If-Statements
   Format
   if <boolean-expression>:
     <statement>
   elif <boolean-expression>:
     <statement>
   else:
     <statement>

   Example
   # Put x’s value in z if it is positive
   if x > 0:
     z = x
   else:
     z = y

   Execution
   If <boolean-expression> is true, then execute all of the statements indented directly underneath (until first non-indented statement)

3. Conditionals: If-Else-Statements
   Format
   if <boolean-expression>:
     <statement>
   else:
     <statement>

   Example
   # Put max of x, y in z
   if x > y:
     z = x
   else:
     z = y

   Execution
   If <boolean-expression> is true, then execute statements indented under if, otherwise execute the statements indented under else
**Conditionals: “Control Flow” Statements**

```
if b:
    s1 # statement
    s3
```

- **Branch Point:** Evaluate & Choose
- **Statement:** Execute

**Flow** Program only takes one path each execution

```
if b:
    s1
else:
    s2
s3
```

**Conditional Expressions**

<table>
<thead>
<tr>
<th>Format</th>
<th>Example</th>
</tr>
</thead>
</table>
| e1 if bexp else e2 | # Put max of x, y in z  
| e1 and e2 are any expression |
| bexp is a boolean expression |

**Example**

```
# Put max of x, y in z  
\[ z = x \text{ if } x > y \text{ else } y \]
```

**Program Flow and Local Variables**

```
def max(x,y):
    **"Returns max of x, y"**
    # swap x, y  
    # put the larger in y  
    if x > y:
        temp = x  
        x = y  
        y = temp  
    return temp
```

- **Value of max(3,0)?**
  - A: 3  
  - B: 0  
  - C: Error!  
  - D: I do not know  

- **What about if we said max(0,3) instead?**