Lecture 8:
Conditionals & Control Flow
(Sections 5.1-5.7)
CS 1110
Introduction to Computing Using Python

Review: Objects are referenced
- Must call constructor function to create object
- Object variable stores ID of object
- Multiple variables can reference same object

Methods: Functions Tied to Classes
• Method: function tied to object
  * Method call looks like a function call preceded by a variable name: (variable).(method)(arguments)

Example:
import shapes
u = shapes.Point3(4,2,3)
u.greet()

“Hi! I am a 3-dimensional point located at (4,2,3)”
Where else have you seen this??

Swap (Question)
What is in p.x at the end of this code?
A: 1
B: 2
C: 3
D: I don’t know

Global p (Question)
What is in global p after calling swap?
A: id1
B: id2
C: I don’t know

Announcements
• Optional 1-on-1 with a staff member to help just you with course material. Sign up for a slot on CMS under “SPECIAL: one-on-ones”.
• A1 first submission due Feb 19 Wedn at 11:59pm
### Example: String Methods

- `s1.upper()`: Returns an upper case version of `s1`.
- `s1.index(s2)`: Returns position of the first instance of `s2` in `s1`.
- `s1.count(s2)`: Returns number of times `s2` appears inside of `s1`.
- `s.strip()`: Returns a copy of `s` with white-space removed at ends.

### Built-in Types vs. Classes

#### Built-in types
- Built-into Python
- Refer to instances as **values**
- Instantiate with **literals**
- Can ignore the folders

#### Classes
- Provided by modules
- Refer to instances as **objects**
- Instantiate with **constructors**
- Must represent with folders

So far only about understanding **objects**; later will create your own **classes**.

### Big Picture

Statements either affect **data** or **control**

- **DATA**: change the value of a variable, create a variable, etc.
  - Examples:
    - `x = x + 1`
    - `name = "Alex"`

- **CONTROL**: tell python what line to execute next
  - Examples:
    - `greet(name)`
    - `if name == "Alex": earlier Lecture`

### Conditionals: If-Statements

#### Format

```
if <boolean-expression>:
  <statement>
  ...
  <statement>
```

#### Example

```python
# is there a new high score?
if curr_score > high_score:
  high_score = curr_score
  print("New high score!")
```

**Execution**

*if (boolean-expression) is true, then execute all of the statements indented directly underneath (until first non-indented statement)*

### What are Boolean expressions?

Expressions that evaluate to a Boolean value.

- **Boolean operations**:
  ```python
  if is_student and is_senior:
    print("Hi senior student!")
  ```

- **Comparison operations**:
  ```python
  if num_credits > 24:
    print("Are you serious?")
  ```

### What gets printed, Round 1

<table>
<thead>
<tr>
<th>a = 0</th>
<th>a = 0</th>
<th>a = 0</th>
<th>a = 0</th>
<th>a = 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>print(a)</td>
<td>a = a + 1</td>
<td>if a = 0: if a = 1: if a = 0:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>print(a)</td>
<td>a = a + 1</td>
<td>a = a + 1</td>
<td>a = a + 1</td>
<td></td>
</tr>
<tr>
<td>print(a)</td>
<td>a = a + 1</td>
<td>a = a + 1</td>
<td>a = a + 1</td>
<td></td>
</tr>
<tr>
<td>print(a)</td>
<td>a = a + 1</td>
<td>a = a + 1</td>
<td>a = a + 1</td>
<td></td>
</tr>
</tbody>
</table>
What gets printed? (Question)

```python
a = 0
if a == 0:
    a = a + 1
if a == 0:
    a = a + 2
a = a + 1
print(a)
```

A: 0  
B: 1  
C: 2  
D: 3  
E: I do not know

---

Conditionals: If-Else-Statements

**Format**

```
if <boolean-expression>:
    <statement>
else:
    <statement>
```

**Example**

```
if curr_score > high_score:
    print("New record!")
else:
    print("Try again next time")
```

**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   # statement
else:
    s1
    s2
```

Flow: Program only takes one path during an execution (something will not be executed!)

---

Program Flow (car locked, 0)

```
def get_in_car(car_locked):
    if car_locked:
        print("Unlock the door.")
car_locked = True
get_in_car(car_locked)
```

---

Program Flow (car not locked, 0)

```
def get_in_car(car_locked):
    if car_locked:
        print("Unlock car!")
    print("Open the door.")
car_locked = False
get_in_car(car_locked)
```
What does the call frame look like next? (Q)

```python
def max(x, y):
    if x > y:
        return x
    return y

max(0, 3)
```

Program Flow and Variables

Variables created inside `if` continue to exist past `if`:
```python
a = 0
if a == 0:
    b = a + 1
print(b)
```

...but are only created if the program actually executes that line of code

Control Flow and Variables (Q1)

```python
def max(x, y):
    """Returns: max of x, y""
    if x > y:
        bigger = x
    return bigger

maximum = max(3, 0)
```

Value of `maximum`?

- A: 3
- B: 0
- C: Error!
- D: I do not know

Control Flow and Variables (Q2)

```python
def max(x, y):
    """Returns: max of x, y""
    # check if x is larger
    if x > y:
        bigger = x
    return bigger

maximum = max(0, 3)
```

Value of `maximum`?

- A: 3
- B: 0
- C: Error!
- D: I do not know

Program Flow and Variables

```python
def zero_or_one(a):
    if a == 1:
        b = 1
    else:
        b = 0
    print(b)
```

Conditionals: If-Elif-Else-Statements

**Format**

```python
if <Boolean expression>:
    <statement>
elif <Boolean expression>:
    <statement>
else:
    <statement>
```

**Example**

```python
if score1 > score2:
    winner = "Player 1"
eelif score2 > score1:
    winner = "Player 2"
eelse:
    winner = "Players 1 and 2"
```
**Conditionals: If-Elif-Else Statements**

**Format**

```
if <Boolean expression>:
    <statement>
...
elif <Boolean expression>:
    <statement>
...
else:
    <statement>
```

**Notes on Use**

- No limit on number of **elif**
- Must be between **if**, **else**
- **else** is optional
- **elif** by itself is fine
- Booleans checked in order
  - Once Python finds a true `<Boolean-expression>`, skips over all the others
  - **else** means all `<Boolean-expression>` are false

**What gets printed, Round 3**

```python
a = 2
if a == 2:
a = 3
elif a == 3:
a = 4
print(a)
```

**If-Elif-Else (Question)**

```python
a = 2
if a == 2:
a = 3
elif a == 3:
a = 4
print(a)
```

**What gets printed?**

- A: 2
- B: 3
- C: 4
- D: I do not know

**Nested Conditionals**

```python
def what_to_wear(raining, freezing):
    if raining:
        if freezing:
            print("Wear a waterproof coat.")
        else:
            print("Bring an umbrella.")
    else:
        if freezing:
            print("Wear a warm coat")
        else:
            print("A sweater will suffice.")
```

**Program Flow and Testing**

Can use `print` statements to examine program flow

```python
# Put max of x, y in z
print('before if')
if x > y:
    print('inside if x>y')
    z = x
else:
    print('inside else (x<=y)')
    z = y
print('after if')
```

"traces" or "breadcrumbs"

x must have been greater than y