Strings, if/else, return, user input

Special thanks to Scott Shawcroft, Ryan Tucker, and Paul Beck for their work on these slides.
Except where otherwise noted, this work is licensed under:
http://creativecommons.org/licenses/by-nc-sa/3.0
Strings

<table>
<thead>
<tr>
<th>index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>or</td>
<td>-8</td>
<td>-7</td>
<td>-6</td>
<td>-5</td>
<td>-4</td>
<td>-3</td>
<td>-2</td>
<td>-1</td>
</tr>
<tr>
<td>character</td>
<td>P</td>
<td>.</td>
<td>D</td>
<td>i</td>
<td>d</td>
<td>d</td>
<td>y</td>
<td></td>
</tr>
</tbody>
</table>

- Accessing character(s):
  - `variable [ index ]`
  - `variable [ index1 : index2 ]`

  - `index2` exclusive
  - `index1` or `index2` can be omitted (end of string)

```python
test_name = "P. Diddy"
>>> test_name[0]
'P'
>>> test_name[7]
'y'
>>> test_name[-1]
'y'
>>> test_name[3:6]
'Did'
>>> test_name[3:]
'Diddy'
>>> test_name[::2]
'P. Did'
```
## String Methods

<table>
<thead>
<tr>
<th>Java</th>
<th>Python</th>
</tr>
</thead>
<tbody>
<tr>
<td>length</td>
<td>len(str)</td>
</tr>
<tr>
<td>startsWith, endsWith</td>
<td>startswith, endswith</td>
</tr>
<tr>
<td>toLowerCase, toUpperCase</td>
<td>upper, lower, isupper, islower, capitalize, swapcase</td>
</tr>
<tr>
<td>indexOf</td>
<td>find</td>
</tr>
<tr>
<td>trim</td>
<td>strip</td>
</tr>
</tbody>
</table>

```python
>>> name = "Martin Douglas Stepp"
>>> name.upper()
'MARTIN DOUGLAS STEPP'
>>> name.lower().startswith("martin")
True
>>> len(name)
20
```
for Loops and Strings

- A for loop can examine each character in a string in order.

\[
\text{for name in string: statements}
\]

```python
>>> for c in "booyah":
...     print c
...
...     print c
...

b
o
o
y
a
h
```
raw_input

raw_input : Reads a string from the user's keyboard.
– reads and returns an entire line of input

```python
>>> name = raw_input("Howdy. What's yer name?")
Howdy. What's yer name? Paris Hilton

>>> name
'Paris Hilton'
```
• **raw_input** can be used to read a number from the user's keyboard by casting the result as an int
  - Only numbers can be cast as ints!
  - Example:

    ```python
    age = int(raw_input("How old are you? "))
    print "Your age is", age
    print "You have", 65 - age, "years until retirement"
    ```

    Output:
    
    How old are you? **53**
    Your age is 53
    You have 12 years until retirement
Exercise

• Write a program that reads two employees' hours and displays each's total and the overall total.
  – Cap each day at 8 hours.

Employee 1: How many days? 3
Hours? 6
Hours? 12
Hours? 5
Employee 1's total hours = 19 (6.33 / day)

Employee 2: How many days? 2
Hours? 11
Hours? 6
Employee 2's total hours = 14 (7.00 / day)

Total hours for both = 33
"format string" % (parameter, parameter, ...)

- **Placeholders** insert **formatted values** into a string:
  - `%d` an integer
  - `%f` a real number
  - `%s` a string
  - `%8d` an integer, 8 characters wide, right-aligned
  - `%08d` an integer, 8 characters wide, padding with 0s
  - `%8d` an integer, 8 characters wide, left-aligned
  - `%12f` a real number, 12 characters wide
  - `%0.4f` a real number, 4 characters after decimal
  - `%6.2f` a real number, 6 total characters wide, 2 after decimal

```python
>>> x = 3; y = 3.14159; z = "hello"
>>> text = "%-8s, %04d is close to %.3f" % (z, x, y)
>>> print text
hello   , 0003 is close to 3.142
```
if

if condition:
    statements

- Example:
  gpa = int(raw_input("What is your GPA? "))
  if gpa > 2.0:
      print "Your application is accepted."
if/else

if condition:
    statements
elif condition:
    statements
else:
    statements

- Example:
  gpa = int(raw_input("What is your GPA? "))
  if gpa > 3.5:
      print "You have qualified for the honor roll."
  elif gpa > 2.0:
      print "Welcome to Mars University!"
  else:
      print "Your application is denied."
## Logical Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Meaning</th>
<th>Example</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>==</td>
<td>equals</td>
<td>1 + 1 == 2</td>
<td>True</td>
</tr>
<tr>
<td>!=</td>
<td>does not equal</td>
<td>3.2 != 2.5</td>
<td>True</td>
</tr>
<tr>
<td>&lt;</td>
<td>less than</td>
<td>10 &lt; 5</td>
<td>False</td>
</tr>
<tr>
<td>&gt;</td>
<td>greater than</td>
<td>10 &gt; 5</td>
<td>True</td>
</tr>
<tr>
<td>&lt;=</td>
<td>less than or equal to</td>
<td>126 &lt;= 100</td>
<td>False</td>
</tr>
<tr>
<td>&gt;=</td>
<td>greater than or equal to</td>
<td>5.0 &gt;= 5.0</td>
<td>True</td>
</tr>
</tbody>
</table>

### Logical Operators Examples

<table>
<thead>
<tr>
<th>Operator</th>
<th>Example</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>and</td>
<td>(2 == 3) and (-1 &lt; 5)</td>
<td>False</td>
</tr>
<tr>
<td>or</td>
<td>(2 == 3) or (-1 &lt; 5)</td>
<td>True</td>
</tr>
<tr>
<td>not</td>
<td>not (2 == 3)</td>
<td>True</td>
</tr>
</tbody>
</table>
Exercise

• Write a program that judges a couplet by giving it one point if it
  – is composed of two verses with lengths within 4 chars of each other,
  – "rhymes" (the two verses end with the same last two letters),
  – alliterates (the two verses begin with the same letter).
• A couplet which gets 2 or more points is "good"

Example logs of execution:
(run #1)
First verse: I joined the CS party
Second verse: Like "LN" and Marty
2 points: Keep it up, lyrical genius!

(run #2)
First verse: And it's still about the Benjamins
Second verse: Big faced hundreds and whatever other synonyms
0 points: Aw, come on. You can do better...