CSCI 141
Scott Wehrwein
String Methods
Goals

- Know how to use a few of the basic methods of string objects:
  - `upper`, `lower`, `find`, `replace`
- Be able to look up and make use of other string methods as needed.
Strings are **objects**.

We’ve seen other objects before: turtles!

Turtles had methods:

```
t = turtle.Turtle()
t.forward(100)
```

- `turtle.Turtle()` is a **module function** (turtle constructor).
- `t.forward(100)` is a **method of a turtle object**.
- `t` is a **variable that refers to a turtle object**.

```
# data and methods
```
Strings are objects.

Strings are objects too - they also have methods.

```
last_name = "Wehrwein"
```

Methods can be called directly on the literal string, too:

```
"Wehrwein".upper()
```
Strings have many methods

here are a few of them:

<table>
<thead>
<tr>
<th>Method</th>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>upper</td>
<td>none</td>
<td>Returns a string in all uppercase</td>
</tr>
<tr>
<td>lower</td>
<td>none</td>
<td>Returns a string in all lowercase</td>
</tr>
<tr>
<td>strip</td>
<td>none</td>
<td>Returns a string with the leading and trailing whitespace removed</td>
</tr>
<tr>
<td>count</td>
<td>item</td>
<td>Returns the number of occurrences of item</td>
</tr>
<tr>
<td>replace</td>
<td>old, new</td>
<td>Replaces all occurrences of old substring with new</td>
</tr>
<tr>
<td>find</td>
<td>item</td>
<td>Returns the leftmost index where the substring item is found, or -1 if not found</td>
</tr>
</tbody>
</table>
String methods: demo
upper, lower, count, replace, find, strip
String methods: demo

upper, lower, count, replace, find, strip

```python
word = "Banana"
word.upper()
word.lower()
word.count("a")
word.replace("a", "A")

line = " snails are out "
line.find("s")
line.find("snails")
line.find("banana")
line.strip()
line.strip().upper()
```

```python
word = "Bellingham"
word = word[0:9] + word[9].upper()
```
String Methods: More

The textbook (Section 9.5) has a more complete listing of string methods:
http://interactivepython.org/runestone/static/thinkcspy/Strings/StringMethods.html

The Python documentation has full details of the \texttt{str} type and all its methods:
https://docs.python.org/3/library/stdtypes.html#str

You should know how to use \texttt{upper}, \texttt{lower}, \texttt{replace}, and \texttt{find}. 
String Methods

Problem: write an expression to determine if a string user_input contains the word "yes", with any capitalization and with any amount of spaces.

```
user_input
=> " Y  eS "
```
String Methods

**Problem:** write an expression to determine if a string `user_input` contains the word "yes", with any capitalization and with any amount of spaces.

```python
user_input.replace(" ", ")
```

=> "YeS"
String Methods

**Problem**: write an expression to determine if a string `user_input` contains the word "yes", with any capitalization and with any amount of spaces.

```python
user_input.replace(" ", ").lower()
```

=> "yes"
String Methods

Problem: write an expression to determine if a string `user_input` contains the word "yes", with any capitalization and with any amount of spaces.

```python
user_input.replace(" ", ").lower() == "yes"
```

=> " Y eS" .replace(" ", ").lower() == "yes"
=> "YeS".lower() == "yes"
=> "yes" == "yes"
=> True

dot (method call) operators are evaluated left-to-right!
Effects vs Return Values, again.

Most turtle methods **change the state** of the turtle object they're called on:

```python
t.forward(100)  # actually moves t forward
```

Most string methods return a **new** string with the given modifications:

```python
s = "BOO"
s.lower()  # => "boo"
print(s)   # prints BOO
t = s.lower()  # if you want "boo", save it
```

Why is this? Because strings can't be modified. Try this:

```python
s = "Scott"
s[3] = "o"  # error
```