def abs(x):
    if x < 0:
        return -x
    elif x > 0:
        return x
    else:
        return 0

>>> abs(-42)
42

>>> abs(0)
0
Print vs Return

def even(x):
    if x % 2 == 0:
        return True
    else:
        return False

def even(x):
    if x % 2 == 0:
        print True
    else:
        print False
def even(x):
    if x % 2 == 0:
        return True
    else:
        return False

if even(6):
    print "6 is even"
else:
    print "6 is odd"
Print vs Return

- Functions which `print` values are useful only to the user.
- Functions which `return` values can be used as building blocks in other functions.

When should you `print`?
- When you want to convey information
- Interacting with the user
- Useful for debugging code

When should you `return`?
- When you want to use your function in a larger context
- Most of the time
A Conditional Shortcut

def even(x):
    if x % 2 == 0:
        return True
    else:
        return False

→ Evaluates to True or False
def even(x):
    if x % 2 == 0:
        return True
    else:
        return False
Logical Connectives

- What can we do with booleans?
  - Combine them

- Logical Connectives
  - and
  - or
  - not
Logical Connectives - and

- When is a and b true?
  - When both a and b are true

>>> True and True

True

>>> True and False

False

>>> False and False

False

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>a and b</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
</tr>
</tbody>
</table>
Logical Connectives - or

- When is **a or b** true?
  - When a is true or b is true (or both)

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>a and b</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
</tr>
</tbody>
</table>
Logical Connectives - or

- When is not a true?
  - When a is false

<table>
<thead>
<tr>
<th>a</th>
<th>not a</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
</tr>
</tbody>
</table>
Logical Connectives Quiz

- $1 < 2 \text{ and } 2 < 3$
- $10 > 100 \text{ or } 'a' == 'a'$
- $(\text{not not True})$
- $(\text{True and False}) \text{ or } (\text{not 7} != 8)$
- $(5 <= 5) \text{ and } (\text{not 'red' == 'blue'}') \text{ and } (\text{'a' >= 0 or 'a' <= 0})$

- BONUS: What does this code do?
  - $x = (x == \text{False})$ (assume $x$ is a defined boolean var)
What's so great about booleans?

● What can we use as a condition?
  
● Boolean values
  ○ False
  ○ True

● Expressions that evaluate to booleans
  ○ 2 < 1
  ○ True or False

● Values that could be interpreted as booleans
  ○ 0
  ○ Any number other than 0
What's so great about booleans?

- "Empty" values are interpreted as False
  - 0
  - 0.0
  - "" (the empty string)

- Everything else is interpreted as True
  - -7
  - 0.1
  - " " (the space character)