If Statements and Boolean Expressions

CSE 1310 – Introduction to Computers and Programming
Vassilis Athitsos
University of Texas at Arlington

Credits: a significant part of this material has been created by Dr. Darin Brezeale and Dr. Gian Luca Mariottini

The Boolean Type

- Expressions of type boolean can only have two values: True, or False.
  - True and False are reserved keywords in Python.

```
>>> 3 > 2
True
>>> type(True)
<type 'bool'>
```

```
>>> a = 3 == 2
>>> a
False
>>> a = (3 == 2)
>>> a
False
```

Preferred style (parenthesize)

```
>>> b = 4;
>>> a = (b != 5)
>>> a
True
```

Comparisons Generating Booleans

- The following operators compare numerical values, or strings, and generate boolean results:

==	equality
!=	not equal
<	less than
>	greater than
<=	less than or equal to
>=	greater than or equal to

Doing Numerical Comparisons

```
>>> a = 3
>>> b = 5
>>> c = 3

>>> a == b
False
>>> a == c
True
>>> a >= c
True
>>> a != c
False
```

Doing Comparisons on Strings

```
>>> a = "hello"
>>> b = "goodbye"
>>> c = "hello"

>>> a == b
False
>>> a == c
True
>>> a < b
False
>>> a >= "world"
False
```

```
>>> a != "hello"
False
>>> a == "hello"
True
>>> a != b
True
>>> a < "sky"
True
>>> "apple" < "car"
True
>>> "apple" < "Car"
False
```

Doing Comparisons on Strings

- Note how "greater than" and "less than" behave on strings:
  - Not exactly based on alphabetical order.
  - Capital letters come before (are "less than") lower case letters.

```
>>> "apple" < "car"
True
>>> "apple" < "Car"
False
```

Logical Operators

- The following logical operators can be used to produce boolean results:

  not

  and

  or

Using Logical Operators

```
>>> a = 3
>>> b = 4

>>> (a == b) or (a+b == 7)
True

>>> (a == b) and (a+b == 7)
False

>>> not(a == b)
True
```

The **in** operator

- The **in** operator checks if a value is included in a set of values.
- For now, we will use **in** to check if a letter appears in a string.
  - We will see more uses when we do lists.

```
>>> var1 = 'a'
>>> vowels = 'aeiouAEIOU'
>>> var1 in vowels
True
>>> 'c' in vowels
False
```

Combining operators

- What does this line do?

```
>>> (3 == 5) and (2 < 3) or (3 >= 0)
```

Combining operators

- What does this line do?

```
>>> (3 == 5) and (2 < 3) or (3 >= 0)
```

- I don't know, and I don't want to know.
  - Use parentheses to make the meaning of these statements clear.

```
>>> ((3 == 5) and (2 < 3)) or (3 >= 0)     → True
>>> (3 == 5) and ((2 < 3) or (3 >= 0))     → False
```

Conditionals - **`if`** statements

- An **`if`** statement is defined as follows:

```
if (expression):
   line 1
   line 2
   …
   line n
```

- Line 1, line 2, …, line n are called the **body** of the **`if`** statement.

An example of an **`if`** statement

```
number_text = input("enter a number: ")
number = float(number_text)

if (number > 2):
  print(number, "is greater than 2")

print("good bye")
```

Another example of an **`if`** statement

```
number_text = input("enter a number: ")
number = float(number_text)

if (number > 2) and (number < 34):
  print(number, "is greater than 2")
  print(number, "is less than 34")

print("good bye")
```

Conditionals - **`if-else`**

- An **`if-else`** statement is defined as follows:

```
if (expression):
  one or more lines
else:
  one or more lines
```

An example of an **`if-else`** statement

```
number_text = input("enter a number: ")
number = float(number_text)

if (number > 2):
  print(number, "is greater than 2")
else:
  print(number, "is less than or equal to 2")

print("good bye")
```

Another example of an **`if-else`**

```
number_text = input("enter a number: ")
number = float(number_text)

if (number > 2) and (number < 34):
  print(number, "is greater than 2")
  print(number, "is less than 34")
else:
  print(number, "is less than or equal to 2")
  print("or greater than or equal to 34")

print("good bye")
```

Conditionals - **`elif`**

- An **`if-else`** can include additional conditions, using **`elif`**:

```
if (expression1):
  one or more lines
elif (expression2):
  one or more lines.
elif (expression2):
  one or more lines
…
```

Conditionals - **`elif`**

- You can use **`elif`** zero, one, or more times.

```
if (expression1):
  one or more lines
elif (expression2):
  one or more lines.
elif (expression2):
  one or more lines
…
```

An example of using **`elif`**

```
number_text = input("enter a number: ")
number = float(number_text)

if (number > 2) and (number < 34):
  print(number, "is greater than 2")
  print(number, "is less than 34")
elif (number <= 2):
  print(number, "is less than or equal to 2")
elif (number >= 34):
  print(number, "is greater than or equal to 34")

print("good bye")
```

Another example of using **`elif`**

```
number_text = input("enter a number: ")
number = float(number_text)

if (number > 2) and (number < 34):
  print(number, "is greater than 2")
  print(number, "is less than 34")
elif (number <= 2):
  print(number, "is less than or equal to 2")
else:
  print(number, "is greater than or equal to 34")

print("good bye")
```

Warning

- Be careful of the difference between = and ==

The Importance of Indentation

```
number_text = input("enter a number: ")
number = float(number_text)

if (number > 2) and (number < 34):
  print(number, "is greater than 2")
  print(number, "is less than 34")

print("good bye")
```

These two programs behave differently.

How, and why?

```
number_text = input("enter a number: ")
number = float(number_text)

if (number > 2) and (number < 34):
  print(number, "is greater than 2")
print(number, "is less than 34")

print("good bye")
```

Successive ifs, vs. if-elif

```
number_text = input("enter a number: ")
number = float(number_text)

if (number > 2)
  print(number, "is greater than 2")
if (number > 10)
 print(number, "is greater than 10")
if (number > 50)
 print(number, "is greater than 50")
```

These two programs behave differently.

How, and why?

```
number_text = input("enter a number: ")
number = float(number_text)

if (number > 2)
  print(number, "is greater than 2")
elif (number > 10)
 print(number, "is greater than 10")
elif (number > 50)
 print(number, "is greater than 50")
```

Conditionals with Strings: Example 1

```
m = input("Enter the name of a month: ")

if (m == "January") or (m == "March") or (m == "May") or (m == "July"):
    print(m, "has 31 days.")
elif (m == "August") or (m == "October") or (m == "December"):
    print(m, "has 31 days.")
elif (m == "April") or (m == "June") or (m == "September") or (m == "November"):
    print(m, "has 30 days.")
elif (m == "February"):
    print(m, "has 28 or 29 days.")
else:
    print(m, "is not a valid month")
```

Conditionals with Strings: Example 1

```
text = input("Enter a word: ")

if ('a' in text) or ('A' in text):
    print(text, "contains at least one 'a'")
else:
    print(text, "contains no 'a'")

if ('b' in text) or ('B' in text):
    print(text, "contains at least one 'b'")
else:
    print(text, "contains no 'b'")
```