# Line Continuation, Output Formatting, and Decision Structures

CS303E: Elements of Computers and Programming
June 13, 2012

#### **Line Continuation**

- What do you do if your line of Python code is too long?
- Use the line continuation character!
  - -'\
  - Place at the very end of the line
  - Python interpreter will assume the next line is part of the same line

### **Output Formatting**

■ We've seen this:

print "The temperature is",temp,"degrees"
 -Now, we'll see another way

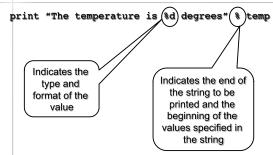
- Better able to control how print displays values
  - -88.3333333 -> 88.33
  - -4/15/2010 -> 4/15/2010

### Output Formatting: Format Operators

print "The temperature is %d degrees" % temp

- **■** String formatting operator: %
  - NOT modulus: modulus operates on numbers
  - Appears in strings
- Indicates how and where a value should be printed in a string
- Also indicates end of print string and beginning of the values to be printed

# Output Formatting: Format Operators



### Output Formatting: Format Specifiers

Specifier	Value Type
%d	integer
%f	float
%s	string
%e or %E	exponential

For each, you can also specify width and precision:

%<width>.<precision><type character>
print "The average is %6.2f" % avg

### Output Formatting: Embedded Operations

You can calculate values in your print statement:

print "
$$x/y = %.2f$$
" % ( $x/y$ )

### Output Formatting: Examples

- Explore use of width and precision using math.pi in the interpreter
- Modify average.py to print 2 decimal places of the average
- Practice printing strings from raw\_input()
- Print using multiple values
  - The values must be enclosed in parentheses

### iClicker Question: Output Formatting

What is the expected output?

A. 7 C. 7.9 B. 7.8 D. 8

### **Comparisons**

- Allows you to compare two values
- Result in a True or False value
   type Boolean (bool)
- You can compare numbers or strings, literals, variables, or expressions

# How do you specify a comparison?

Specify the comparison using a *relational* operator

Operator	Meaning
<	Less than
>	Greater than
<=	Less than or equal
>=	Greater than or equal
==	Equality
!=	Not equals

## Comparisons: Examples

1. test = 13 < 15

2. test = 101 >= 99

3. test = "a" < "b"

4. test = 4 == 2+2

5. test = 15 != 16

6. test = 12 == 3\*5

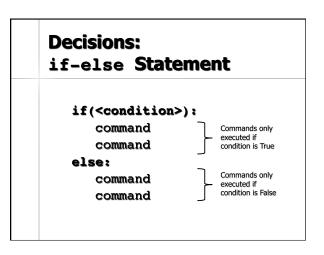
### **Lexicographic Order**

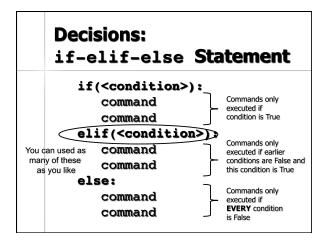
- Strings are rated according to lexicographic order
- Orders words A-Za-z
  - Capital letters first in alphabetical order
  - Lower-case letters second in alphabetical order
- NOT in dictionary order

#### **Decisions**

- Gives you the ability to specify different instructions based on a condition
- The condition is typically a comparison
  - if some comparison is true do something

#### **Decisions:** if Statement def main(): Commands not dependent on the command if(<condition>): command Commands only executed if condition is true command command Commands not dependent on the condition command main() Indentation matters! (Again)





### Decisions: Nested ifs

elif(<condition>):

You can put if statements inside the
body of the if (or elif or else)
statement:
if(<condition>):
 if(<some other condition>):
 command
else:
 command

### **Decisions: Gotchas**

- Exactly ONE of the bodies in if-elif-else will be executed
  - Only the first True condition
  - THINK about the construction of your if statements before coding
- Comparison of floats

if(.3 == .1+.2) is False

### iClicker Question: Decisions

What is the expected output?
if(125<140):
 print "first one"
elif(156>=140):
 print "second one"
else:
 print "third one"

A. first one C. third one

B. second one D. first one, second one

#### **Exercise**

Write a program that prompts the user for a number and then prints "You are lucky!" if the number is a multiple of 7, "Dozens and dozens!" if the number is a multiple of 12 (but not of 7), and prints the number otherwise.

#### **Reminders**

#### Assignment 2 due tomorrow

- RUN your program before you turn it in
   If it doesn't run without an error, you get a 0. (No tracebacks!)
- Type it in the EDITOR. Not the interpreter, or it will not RUN.
- Test it and include your test cases.
- Turn in your assignment using the RIGHT filename. (EasterSunday.py) Make sure your capitalization is right, too.