CMPT 120
Basics of Python

Summer 2012

Instructor: Hassan Khosravi

Python

- A simple programming language to implement your ideas
  - Design philosophy emphasizes code readability
- Implementation of Python was started in 1989 by Guido van Rossum
- In this course we will be using the Python 2.7.3 version
  - http://www.python.org/download/releases/2.7.3/
- Python has an interactive interpreter. It will execute immediately.
- You can also type python code into a file and save it.

First program

- For some reason, when people are taught to program, the first program they see is one that prints the words “Hello world” on the screen.
- >>> print "Hello world"
  - Hello world

- The Interpreter vs. the Editor
  - Running “hello world” with both

- Any text in quotes, like "Hello world" in the example, is called a string.
- Characters are letters, numbers, spaces, and punctuation. Strings have to be placed in quotes to be distinguished from Python commands

Statement

- Statements are the basic building blocks of Python programs. Each statement expresses a part of the overall algorithm that you're implementing.

- The statements are executed in the order they appear in the file. So, the Python program
  - print "Hello world!"
  - print "I'm a Python program that prints stuff."
    - Hello world!
    - I'm a Python program that prints stuff.

Doing Calculations

- The Python operators +, -, *, and / perform addition, subtraction, multiplication, and division, as you might expect.
  - >>> print 10 - 2
    - 8
  - >>> print 15/3
    - 5
  - >>> print 25+19*5
    - 120
  - >>> print 10.2 / 2 / 2
    - 2.55
- The order is the same as mathematics operators
- You can () to change the order of operators
  - print (76+100)/2
  - print 76+100/2

Calculations on strings

- >>> print "An" + "Expression"
  - AnExpression
- >>> print "An " + ’Expression’
  - An Expression
- >>> print ’ABC’ * 4
  - ABCABCABCABC
- A number, or anything else in quotes, is treated like a string
  - >>> print 120 * 3
    - 360
  - >>> print "120" * 3
    - 120120120
  - >>> print "120 * 3"
    - 120 * 3
- single quotes (’) and double quotes (") can be used interchangeably.

Functions

- Python can also use functions as part of expressions.
  - You give the function some arguments, and something is done to calculate the result
- >>> print round(13.89)
  - 14.0
- >>> print round(-4.3)
  - -4.0
- >>> print round(1000.5)
  - 1001.0

Functions

- Functions can take any type of information as their argument and can return any type.

- print len("hello")
  - 5
- >>> print len("-<()>-")
  - 6
- >>> print len("")
  - 0

Storing Information

- Sometimes, you need to perform a calculation to be used later, without needing to display the results right away.

- Whenever we need the computer to temporarily remember some information in a program, we will use a variable.
- Average = (10 +20)/2
- >>> num = 7
- >>> word = "yes"
  - >>> print num - 3
    - 4
  - >>> print word + word
    - yesyes
- >>> num = 4
  - >>> print num – 3
    - 1

Types

- Python treats numbers (like 2, -10, and 3.14) differently than strings
  - print 10/2
    - 5
  - >>> print "abc" / 2
    - TypeError: unsupported operand type(s) for /: ’str’ and ’int’
- TypeError indicates that you’ve used values whose types can’t be used with the given operation.
- + operator does different things on numbers (addition) and strings (joining)
- >>> print 10/2
  - 5
- >>> print 10/3
  - 3
- >>> print 10.0/3
  - 3.33333333333

Type Conversion

- There is a built-in function called type
  - type(10/3)
  - type(10.0/3)

- There are Python functions that can be used to change a value from one type to another.
  - int() converts to an integer
  - float() converts to a floating point value
  - str() converts to a string.

Example

- float(10)
  - 10.0
- >>> str(10)
  - ’10’
- >>> int(’10’)
  - 10
- >>> int(83.7)
  - 83
- >>> str(123.321)
  - ’123.321’
- >>> int("uhoh")
  - ValueError: invalid literal for int(): uhoh

- >>> total = 46
- >>> num = 10
- >>> print total/num
  - 4
- >>> print float(total)/num
  - 4.6
- >>> print float(total/num)
  - 4.0

- You can print out multiple values with the comma, but they are separated by spaces:
  - >>> print "The sum was", total, "."
  - The sum was 46 .

- Note that there's a space between the 46 and the period. You can remove this by combining strings to get the result we want:
  - >>> print "The sum was " + total + "."
  - TypeError: cannot concatenate 'str' and 'int' objects

  - >>> print "The sum was " + str(total) + "."
  - The sum was 46.

User Input

- To do this in Python, use the raw_input function. This function will give the user whatever message you tell it to, wait for them to type a response and press enter, and return their response to your expression.
  - name = raw_input("What is your name? ")
  - print "Hello, " + name + ". "
- If you want to treat the user's input as an integer or floating point number, you have to use one of the type conversion
  - m = float(raw_input("Enter your height (in metres): "))
  - inches = 39.37 * m
  - print "You are " + str(inches) + " inches tall."

Example problem solving feet and inchs

- write “Enter your height (in metres):”
- read metres
- set totalinches to 39.37 × metres
- set feet to [totalinches/12]
- set inches to totalinches − feet × 12
- round inches to the nearest integer
- write “You are” feet inches ″ tall.”

  - metres = float(raw_input("Enter your height (in metres): "))
  - total_inches = 39.37 * metres
  - feet = int(total_inches/12)
  - inches = total_inches - feet*12
    - inches = int(round(total_inches - feet*12))
  - print "You are " + str(feet) + " feet and " + str(inches) + " inches tall."

Printing single and double quotes

- How can we print out single quote or double quote
  - Print “Hi”
  - Print ‘Hi’
  - What if you want “Hi” to be the output?

  - Put a backslash before the quote.
    - print " \"hi\" "
  - Use a single quote to wrap up the string.
    - print ' "Hi" '
    - print " 'Hi' “
  - Use s triple-quoted string
    - print """ "Hi" """
    - print """ 'Hi' """

Reading

- Read Chapter 2 from Introduction to Computing Science and Programming I
- Read Chapter 2 in How to Think Like a Computer Scientist