

## Objectives

- Wrap up string operations
- Built-in functions
- Import statements
- Broader Issue

Jan 18, 2008

Sprenkle - CS111

1

## Review from Last Time

- Type conversion
  - Use type's constructor
- Shorthands, such as `x+=1`
- String operators
  - `+` : concatenate strings together
  - `*` : concatenate the string *n* number of times
  - `%`: format operator
    - template % (replacement\_values)
    - Format specifiers: %[flags][width][.precision]code

Jan 18, 2008

Sprenkle - CS111

2

## Formatting Practice

- `x=10`
- `y = 3.5`
- `z = "apple"`
- `"%6.2d" % x`
- `"%6.2f" % x`
- `"%06.2f" % y`
- `"%+6.2f" % y`
- `"%-10s" % z`
- `"%5d %-7.3f" % (x,y)`

Jan 18, 2008

Sprenkle - CS111

3

## Example: Printing Out Tables

- A table of temperature conversions

Temp F	Temp C	Temp K
-459.7	-273.1	0.0
0.0	-17.8	255.4
32.0	0.0	273.1

- If we want to print data in rows, what is the template for what a row looks like?
  - How do we make the column labels line up?

Jan 18, 2008

Sprenkle - CS111

4

## Using Built-in Functions

- Functions perform some task
  - May take **arguments/parameters**
  - May **return** a value that can be used in assignment
- Syntax
  - `func_name(arg0, arg1, ..., argn)`
    - Argument/parameter list
- Depending on the function, the arguments may or may not be required
  - `[]` indicate an optional argument
- Semantics: depend on the function

Jan 18, 2008

Sprenkle - CS111

5

## Example Functions

Known as function's "signature"

Template for how to "call" function

Optional argument

- `raw_input([prompt])`
  - If prompt is given as an argument, prints the prompt without a newline/carriage return
  - If no prompt, just waits for user's input
  - Returns user's input (up to "enter") as a **string**
- `input([prompt])`
  - Similar to `raw_input` but returns a **number**

Jan 18, 2008

Sprenkle - CS111

6

## More Examples of Built-in Functions

- **round(x, n)**
  - Round the float **x** to **n** digits after the decimal point
  - If no **n**, round to nearest **int**
- **abs(x)**
  - Returns the absolute value of **x**
- **type(x)**
  - Return the type of **x**
- **pow(x, y)**
  - Returns  $x^y$

Jan 18, 2008

Sprengle - CS111

7

## Using Functions

- Example use: Alternative to Exponentiation
  - Goal: compute  $-3^2$
  - Python alternatives:
    - `pow(-3, 2)`
    - `(-3)**2`
- Typically, we use functions in assignment statements
  - Function does something
  - We save the result of function in a variable

Jan 18, 2008

Sprengle - CS111 `function_example.py` 8

## Python Libraries

- Beyond built-in functions, Python has a rich **library** of functions and definitions available
  - The library is broken into **modules**
  - A **module** is a file containing Python definitions and statements
- Example modules
  - `math` -- useful math functions
  - `os` -- useful OS functions
  - `network` -- useful networking functions

Jan 18, 2008

Sprengle - CS111

9

## Example Library: Math Module

- Has constants (variables) for  $\pi$  (i.e., pi) and  $e$ 
  - These values never change, i.e., are constants
  - Typically, we'll name constants with all caps
- Has functions such as
  - **ceil(x)**
    - Return the ceiling of **x** as a float
  - **exp(x)**
    - Return  $e$  raised to the power of **x**
  - **sqrt(x)**
    - Return the square root of **x**

Jan 18, 2008

Sprengle - CS111

10

## Using Python Libraries

- To use the definitions in a module, you must first **import** the module
  - Example: to use the `math` module's definitions, use the the import statement: `import math`
  - Typically import statements are at *top* of program
- To find out what's available in module, use the **help** function
  - Example:

```
import math
help(math)
```

Jan 18, 2008

Sprengle - CS111

11

## Using Definitions from Modules

- Prepend constant or function with "**module**name."
  - Examples for constants:
    - `math.pi`
    - `math.e`
  - Examples for functions:
    - `math.sqrt`
- Practice
  - How would we write the expression  $e^{\ln} + 1$  in Python?

Jan 18, 2008

Sprengle - CS111 `module_example.py` 12

## Alternative Import Statements

- **from <module> import <defn\_name>**
- Examples:
  - `from math import *`
    - Means "import everything from the math module"
  - `from math import pi`
    - Means "import pi from the math module"
- With this **import** statement, don't need to prepend module name before using
  - Example: `e**(1j*pi) + 1`

Jan 18, 2008

Sprengle - CS111

13

## Python Libraries

- Python has a rich library of functions and definitions available for your use
  - The library is broken into **modules**
  - A **module** is a file containing Python definitions and statements
- Benefits of functions/definitions in modules
  - Don't need to rewrite someone else's code
  - If it's in a module, it is a very efficient (in terms of computation speed and memory usage)

Jan 18, 2008

Sprengle - CS111

14

## Finding Modules To Use

- How do I know if some code that I want already exists?
  - Python Library Reference:
    - <http://docs.python.org/lib/lib.html>
- For example, **string** module has functions/constants for manipulating strings
- For the most part, to practice, in the beginning you will write most of your code from scratch

Jan 18, 2008

Sprengle - CS111

15

## Broader Issues

- Great questions, summaries
- If alternatives, make explicit which article you read in your blog entry
- Groups:

#1:	#2:	#3:	#4:
Greg	Alex	Clay	Stuart
Dave	Nay	Arturo	Vasil
Joe	Julie	Joa	Colin
Andrew	Ty	Lucy	

Jan 18, 2008

Sprengle - CS111

16

## Broader Issues: Brief Summary

- WikiScanner
  - Tracks IP address/domain of who edits articles
  - Corporations change article for their benefit
- Knol
  - Similar (but different) goals to Wikipedia
    - Maintain author's name
  - Google's side project where author is accountable

Jan 18, 2008

Sprengle - CS111

17

## Broader Issues

- Pros and Cons of Wikipedia
- Would you use the Google version?
  - What would be necessary for you to use the new version?
- How are these articles related?
- Do you have a preferred search engine?
  - If so, why do you use this particular search engine?
- Is having an unbiased search engine important to you?

Jan 18, 2008

Sprengle - CS111

18

## Broader Issues: Class Relation

- Feature options
  - Which to implement and how does that affect your product?
- Better Algorithms == Better Business?
- Networking