CS177 Python Programming

Recitation 2 - Computing with Numbers
Announcement

• GTAs have posted their office hours. Check http://courses.cs.purdue.edu/cs17700:fall14:start
Outline

• Data types.
• Basic numeric data types in Python.
• Python math library.
• write programs that read user input and process numerical data.
What is data?

• The information that is stored and manipulated by computer programs is referred to as data.

• Data can be number, string (character, word, sentence, etc.), or even object (will be discussed in the later chapter)
Numeric Data Type

• Number is a most common data type
• Numbers can be classified as natural numbers, integers, rational numbers, real numbers, complex numbers, computable numbers ...
• Integers and decimals will be discussed and used in this course
Numbers in Python

• In Python, whole numbers are referred to as *integers*; decimals are referred to as *floating point*.
  – A numeric literal **WITHOUT** a decimal point produces an int value.
  – A literal that **WITH** a decimal point is represented by a float (even if the fractional part is 0).
Exercises

type(9.5)
type(2)
type(2.0)
type(0.0)
type(0)
What about large integers?

• What to do if our program involves very large numbers?
• In C programming language:
  - int (which is signed) can go: -32767 to 32767
    - unsigned int: 0 to 65535
    - signed long: -2147483647 to 2147483647
    - unsigned long: 0 to 4294967295
    - signed long long: -9223372036854775807 to 9223372036854775807
    - unsigned long long: 0 to 18446744073709551615
What about large integers?

• The good news is, Python has arbitrary precision integers so there is no true fixed maximum. You're only limited by available memory!
What about decimals?

- Floating-point numbers are represented in computer hardware as base 2 (binary) decimals, i.e. $0.125 = (0.001)_2$
- Unfortunately, the decimal floating-point numbers are only approximated by the binary floating-point numbers actually stored in the machine (finite number of digits).
  - Example $0.1 \approx (0.00011001100110011001100110011001100110011001100110011001100110011010)_2$
Arithmetic Operators

- Arithmetic operators inherit their definitions on numerical data types (int/floating point).
- Operations on floats produce floats.
- Operations on int produce int (except for `/`).
- What if one operand is int and the other is floating point?
  - `2+3.0`
Exercises

type(2)
type(0.0)
type(2+3)
type(2.0+3)
type(2*3)
type(2/3)
type(2//3)
type(2%3)
Math Library

• Python has math library that can perform powerful computations
  – time: provides various time-related functions
  – math: provides access to the mathematical functions defined by the C standard
  – random: generates pseudo-random numbers with various common distributions.
Math Library

• To use a library, we need to make sure to import the library in our program:

• For the Math library:
  – import math

• Then we can do...

  math.pow(x, y) - Return x raised to the power y.
  math.sqrt(x) - Return the square root of x.
  math.factorial(x) - Return x factorial.
  math.ceil(x) - Return the ceiling of x.
  math.floor(x) - Return the floor of x.
>>> import math
>>> a = math.factorial(6)
>>> print(a)
720

>>> b = math.sqrt(123)
>>> print(b)
11.0905365064
Math Library Examples

>>> c = math.floor(5.9)
>>> print(c)
5

>>> x = math.factorial(4) * math.pow(2, 3)
>>> print(x)
192.0
Math Library Examples

```python
>>> y = 5.5
>>> z = math.floor(y) * math.ceil(y)
>>> print(z)

>>> y = -5.5
>>> z = math.floor(y) * math.ceil(y)
>>> print(z)
```

The results for both code snippets are 30, but does `math.floor(5.5)` equal to `abs(math.floor(-5.5))`?
Range

• Python allows us to specify a range of values
  - Example
    range(n)  0, 1, ..., n-1

list(range(10))
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
Range

• What if we don’t want to start from 0?
• range(n) is short hand for range(0, n), and we can set the start number explicitly
• range(start, end)
  - Example
    list(range(-4, 4))
    [-4, -3, -2, -1, 0, 1, 2, 3]
Range

- What if we don’t want to count up by 1?
- Python allows us to “step” by a given integer
- `range(start, end, step)`
  
  - Example
    
    `list(range(0, 10, 2))`

    `[0, 2, 4, 6, 8]`
Range

• What if we want to count down?
• Since Python allows us to “step” by a given integer, let’s try `list(range(0, 10, -1)) : []`
• What about `list(range(10, 0, -1))`?
  
  `[10, 9, 8, 7, 6, 5, 4, 3, 2, 1]`
String in Python

- String is another data type in Python
- It is known as a sequence of characters
- String can be literal constants or variables
- Two useful functions with String object are `input()` and `eval()`
String in Python

• *input()* is used to obtain data from user
  - Example

  applicant = input("Enter the applicant's name: ")
  interviewer = input("Enter the interviewer's name: ")
  time = input("Enter the appointment time: ")
  print(interviewer, "will interview", applicant, "at", time)
String in Python

• `eval()` is used as `eval(expression)`
• The `expression` argument is parsed and evaluated as a Python expression
  - Example
  ```
x = 1
print(eval('x'))
  ```
  Result: 1
String in Python

• We can also do some calculation within `eval()`
  - Example
  
x = 1
  
  print(eval('x+1'))

  Result: 2

• Notice that the `expression` argument should be a string type