CSC 110: Week 2, Lecture1

Dr. Bodine

Last week we covered…

- What are computers
- Basics of programming language
  - Compiled vs interpreted
- Python environment
  - IDLE, shell, importing, etc.
- Writing a basic program
  - Print statements
  - User input
  - Very basic data manipulation
- Expression, variables, operators, literals

This week…

- Number types in python and in other languages
  - Different types of arithmetic
  - import math
- Number operators
  - +, -, *, /, **
  - abs, %, //
- Loops
  - range(n)

Assignments and Quizzes

- Assignment 1 is due today!
  - Assignments will be due most Tuesdays, covering previous week's material
  - Submit via canvas; attach .py file. Remember to give it an appropriate name!
- Assignment 2 will be posted this week, due next Tuesday
- Quiz 1 is today in class
  - Will have quizzes on most Tuesdays, covering material from previous week
  - Administered in Canvas (requires passcode)
  - If you are going to miss a class, inquire ahead of time

Numbers in math

- Integers (and natural numbers)
  - Whole numbers: -3,-2,-1,01,2,3,

- Real numbers
  - Numbers with fractional parts (Ex: 1.3)
  - Whole numbers where you put a 0 behind the decimal (Ex: 1.0)

- Signed vs unsigned
  - Positive or negative
  - Arithmetic defined in a universal way

Arithmetic in math

- Integer addition/subtraction/multiplication produces an integer
- Real number addition/subtraction/multiplication produces a real number
- What about division?
  - Foreshadowing…

Zero-based numbering

- Do numbers start at 1 or 0?
  - Well that depends on who you ask.
  - Natural numbers, counting numbers, …
- Examples of zero-based numbering?
- In general in CS, start index labeling at 0
  - 0^{th} element, 1^{st} element, 2^{nd} element, … $(n-1)^{th}$ element
  - Read: *Why numbering should start at zero* by E. Dijkstra

Numbers in programming

- Integers
  - Whole numbers
  - Discrete, not fractional
- Floats, doubles, etc.
  - Represent real numbers, can be whole or fractional
  - Approximations of the value
- Signed vs unsigned
  - Some languages have different versions for signed and unsigned
  - Why might this be useful?

- Arithmetic (in general)
  - Operations on integers produce integers*
    - *minor exception: division in python
  - Operations on floats produce floats

Numbers in python

- Integers
  - Whole numbers, can be positive or negative
- Floats
  - Numbers with fractional parts
  - Size of numbers allowed

- Arithmetic
  - Operations on ints produce ints (* except division)
  - Operations on floats produce floats
  - Mixed operations produce floats
  - Integer division with **//** operator

But last week we didn't declare a type!

- True! Python interprets the type on the fly
  - "Duck-typing"
- To check the type use the type function
  - >>> type(<variable>)
- If we want to change the type we can on the fly

```
>>>
>>> x = 3
>>> type(x)
<class 'int'>
>>> x = 1.0*x
>>> type(x)
<class 'float'>
>>> x
3.0
>>>
```

Dynamic typing

- Eases development as you don't have to declare types up front
- Succint and timely
  - No need to specify int, float, etc

- Can promote bad habits of not thinking through types up front
- Difficulty in bug detection
  - Only find when run and comes cross error
- Harder to trace issues when types are not declared
    - Have to work through program keeping track of types on your own

Integer division operator: //

- The // operator is used to find the integer component of division
  - x // y returns the number of times y goes into x (ignores remainder)

```
>>> x = 3
>>> y = 13
>>> x//y
0
>>> y//x
4
>>>
```

Remainder operator: %

- The % operator returns the remainder
  - x%y is the remainder when x is divided by y

```
>>>
>>> x = 3
>>> y = 13
>>> x%y
3
>>> y%x
1
>>>
```

Absolute value: abs

- abs(<number>) returns the absolute of <number>

- Built-in function in python
  - Do not need any external library

```
>>> x = -3.5
>>> abs(x)
3.5
>>>
>>> y = -2
>>> abs(y)
2
>>>
```

Type conversion

- May want to change type between ints and floats

- Can convert ints to floats
  - Explicitly: float(x)
  - Implicitly: x = 1.0*x

- Can convert floats to ints
  - Explicitly: int(x)

Think pair share

- Partner up. Decide ahead of time who will be person A and who will be person B.

- Person A: Think about the difference between the following lines:
  - x = 11/3
  - y = 11.0/3.0

- Person B: Think about the difference between the following lines:
  - x = 11//3
  - y = 11.0/3.0

In-class activity

- Simplify the following expressions by hand and give the type:
  - 11**2/10.0+5
  - 4//3+8*2
  - (3**3)/3.0
- When finished confirm the simplifications via python command line

Challenge

- Using the built-in python functions, write a program that takes two numbers as inputs (prompting user for them) and finds the improper fractional representation and the mixed number representation of the division of the two numbers.