Python + Math = <3

If you haven't set up your Python environment with a Python install and an IDE, please go back to the main menu and go to setup.

Using Python as a calculator

You can use Python as a calculator. You can use it to calculate the area of a circle, the number of minutes in a year, the amount of distance traveled at a speed, volume of a swimming pool and many more things.

However you need to be careful of how to calculate things because Python acts a bit different than a standard calculator. The reason for this is data types.

Types of Data

In computers all values are stored in binary, which is a base-2 system. Certain types of data have certain formats in the memory saved to the computer. In C++ and C, you must tell the program what type of data is being saved. Since Python is based on C++, we have the same types that C++ has in programming.

int

Int is for integer. They are whole number values such as the following: {... -2, -1, 0, 1, 2, ...}.

float

Float is for floating point. They are real numbers like 1.5, pi and e.

These are all we need to know about for now.

Operations

In Python we have multiplying (*), dividing (\), exponents (**), adding (+), subtracting (-), and modulo (%).

Modulo

Modulo probably stands out as something you might not know, but you actually do know. It is the operator to find the reminder after division. If I did `7%3`, I'd get `1` as the answer. It also works with negative numbers. To explain what the answer to `-1 % 5`, think of the action of `A%B` as subtracting `B` from `A` until A is positive and less than B (you can divide as well). Since we have a negative

value, that means we went too far and we have to add `B`, So we add `5` to `-1` and get `4` as the answer.

Problems:

Try to do these by math and then put them into Python to see if you understand how it works. * `13 % 2` * `8 % 2` * `-2 % 7` * `30 % 13`

Division with integers

The best way I can explain this quirk is that think of division as a math equation. If you have a equation that takes in two integers, you'd expect it to output an integer. That is what happens when adding, subtracting, and multiplying. The same thing happens for dividing two integers. So when you type `3/2` into the Python shell, you will get `1` as the answer.

Problems:

- `13 / 2`
- `8 / 2`
- `-2 / 7`
- `30 / 13`

Division with integers and floats

Now what if you divide an int by a float or a float by an int? Python decided that you get a float. so `3/2.0` is `1.5`, which is the answer to `3.0/2` as well.

A note, when dealing with very small numbers, sometimes the numbers become incorrect in computers. The reason for this is binary. When storing numbers as decimals, they are stored as 1/2 + 1/4 + 1/16 or what ever. 1/3 in binary is still a repeating decimal and the computer will try to store a value as close as it can to this. Some values will appear different than what you expect.

Problems

Play with Python as a calculator

- `3 ** 2 ** 1`
- `2 ** 3 ** 4`
- `2 / (2.0 ** 3)`
- `(7 ** 2) % 3`