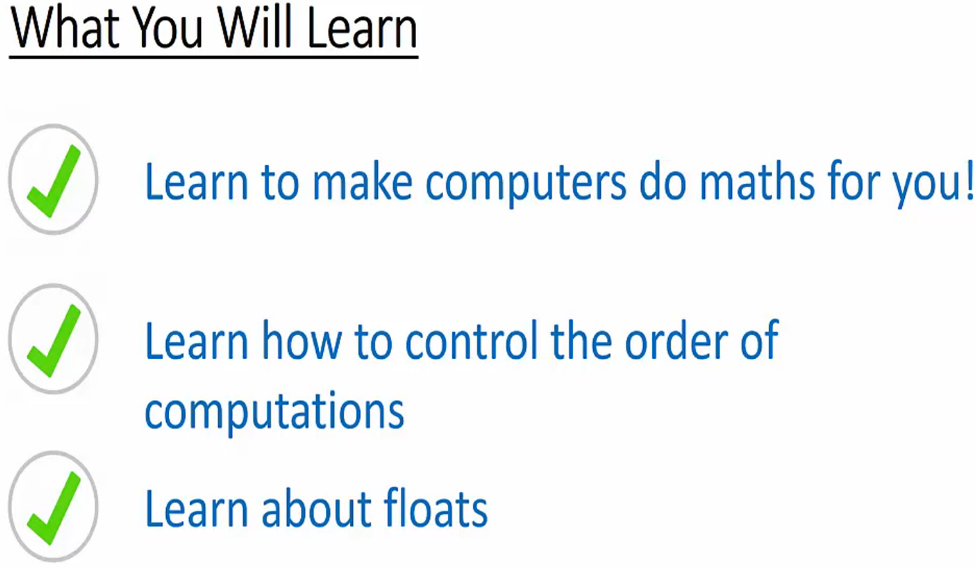
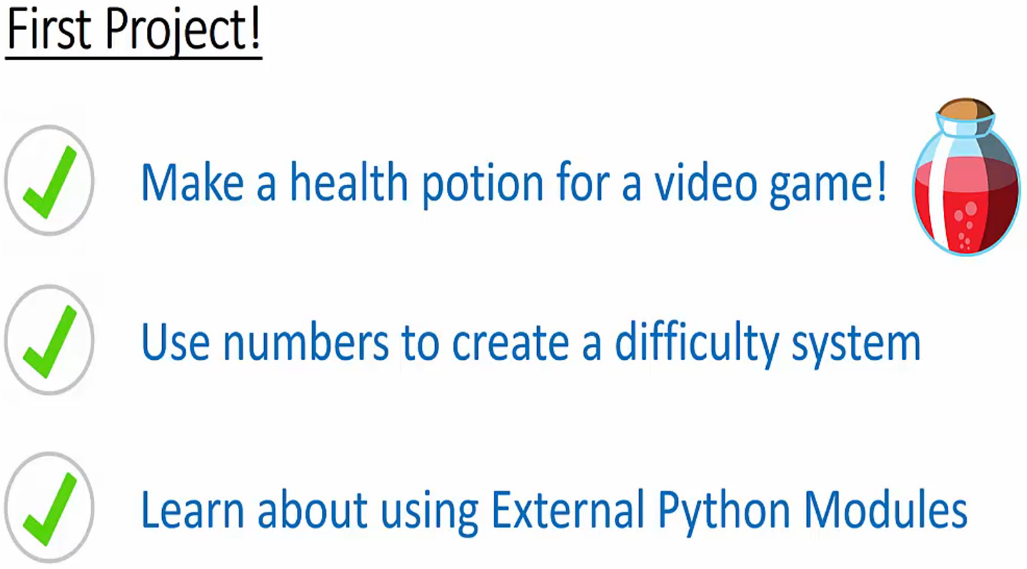
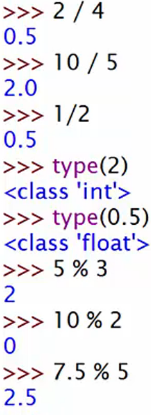
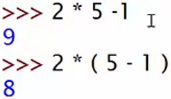
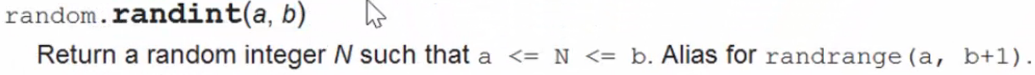
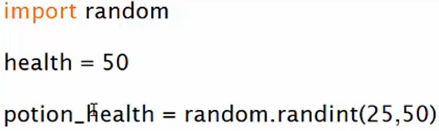
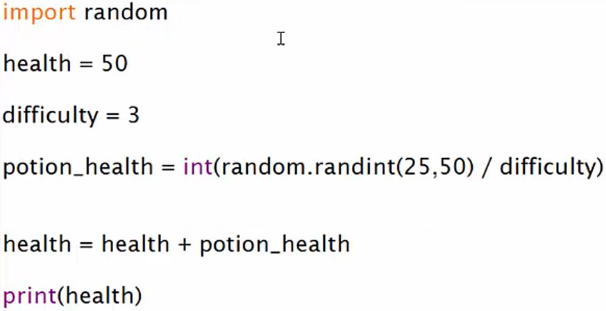
**Section Overview**  
 

**Basic Arithmetic, Floats and Modulo**  
\* **It’s better to put spaces around the operator, making it less cramped**.  
\* **In Python 2, every / division would give back a whole number**.  
\* **In Python 3, every / division gives back a float number**.  
**int** => a whole number.  
=> **takes 14 Bytes of memory**.  
**float** => a decimal number, a floating point number, allowing for extra levels of precision.  
=> **takes 16 Bytes of memory**.  
\* **Modulo allows us to find the remainder of a division**.  


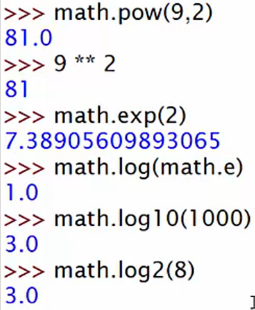
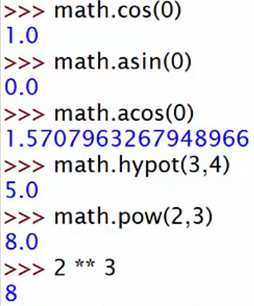
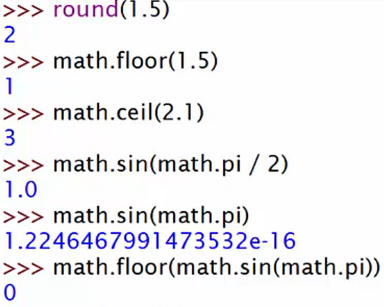
**Ordering Operations using (Brackets!)**  
**BODMAS**

**Brackets** => () parentheses  
**Order** => squaring, cubing, square roots, cube roots, raising any number to the power of something  
**Division** => /  
**Multiplicaton** => \*  
**Addition** => +  
**Subtraction** => -  
\* **Brackets can be used to force operations to occur in a specific order**.  


**PROJECT 1: Crafting a Health Potion - Part 1**  
\* **The way that we use Python Libraries is by importing them**.  
**import random**  
**=> Help => Docs => Modules (top right) => r => random**  
  
**random.randint()**  
\* **In the shell if you want to copy & paste a command, put your cursor on that line and hit enter**.  


**PROJECT 1: Crafting a Health Potion - Part 2**  
**Casting** => **converting a certain piece of data to another type**  
**int()**  
**float()**  


**Fun with the Python math Module (Optional)**  
**round()** => **rounds to the nearest 1.1 to 1 and 1.5 to 2  
=> it’s built-in to Python, no need to import  
math.floor() => rounds down  
math.ceil() => rounds up  
math.pi => 3.141592653589793  
math.e => 2.718281828459045  
Trigonometric functions take numbers in radiants and not degrees by default  
math.sin()  
math.cos()  
math.asin()  
math.acos() => half of pi - as you can see on the cosine graph  
We can also use the math library to find the hypotenuse of a right angle triangle using the Pythagorean Theorem  
math.hypot() => 3\*3 + 4\*4 = 25, sqrt of 25 = 5  
math.pow() => number, power, gives us a float  
\*\* => square, gives us a whole number if we use whole numbers  
math.exp() => find the exponential of e (2.718… \* 2.718… = 7.389…)  
math.log() => natural log  
math.log2() => log base 2  
math.log10() => log base 10**

 **\* You can check the numpy and scipy libraries for more advanced math/science that is best suited for data analysis.  
for data analysis:  
pandas  
NumPy  
SciPy**

**Section Review**

**Quiz 2: Python numbers quiz**

**Coding Exercise 2: Basic Algebra Coding Challenge**

**Coding Exercise 3: Overwriting Variables**