Model 1 Integers and Floats

Every value in Python has a ***data type*** which determines what can be done with the data. Enter the following code, one line at a time, into a Python Shell. Record the output for each line (if any) in the second column.

Python code	Shell output
`integer = 3`	
`type(integer)`	
`type("integer")`	
`pi = 3.1415`	
`type(pi)`	
`word = str(pi)`	
`word`	
`number = float(word)`	
`print(word * 2)`	
`print(number * 2)`	
`print(word + 2)`	
`print(number + 2)`	
`euler = 2.7182`	
`int(euler)`	
`round(euler)`	

Questions (15 min) Start time:

2. What is the data type (`int`, `float`, or `str`) of the following values? (Note: if you're unsure, use the `type` function in a Python Shell.)

a) pi

b) integer

c) word

d) number

3. List the function calls that convert a value to a new data type.

4. How does the behavior of the operators (+ and *) depend on the data type?

5. What is the difference between the `int` function and the `round` function?

6. What is the value of `3 + 3 + 3`? What is the value of `.3 + .3 + .3`? If you enter these expressions into a Python Shell, what do you notice about the results?

7. In order to store a number with 100% accuracy, what data type is required? How might you precisely represent a bank account balance of $123.45?

8. Try calculating a very large integer in a Python Shell, for example, 123^{456}. Is there a limit to the integers that Python can handle?

9. Try calculating a very large floating-point number in a Python Shell, for example, 123.0^{465}. Is there a limit to the floating-point numbers that Python can handle?

10. Summarize the difference between the numeric data types (`int` and `float`). What are their pros and cons?