Daily Log

This is where I'll record hopefully before class what we will be doing. Please check here and read before class, so you'll have an idea of what we will cover that day.

---

day 3

Today we will finish our discussion of data types, by talking about strings and talking about lists or arrays. We have already used strings to print statements in words to the terminal. They are an important class of data, but less used in physics simulations. However the most important data structure to us is the list, and the important upgrade to the array. We will learn to create and import a list, how to print it and will begin to see how to graph it.

A string is any kind of data stored between single quotes, such as `'Hello World'`. A string can contain numbers, but they are not stored or used as numbers but rather essentially as letters. If a number happens to be stored as a string and you would like to use it as a number, then you can use the `float` or `int` functions for this. There are a few other properties of strings that we can talk about once we get through the next section on *lists*.

A list is a collection of data structures, and while technically anything can go in the container, we will be dealing with lists mostly as a collection of float numbers.

---

day 2

Now that we have `conda` installed and working, we need to make sure it has all of the libraries that we will use. We will use matplotlib, numpy, pandas, and scipy for the most part although there may be others that would be nice. Let's try to use them first and then we will install them if necessary. To import these, we use an import statement like `import numpy as np`. This will bring all of the functions from the `numpy` package into our code, and then we can use them if we first use the `np` identifier. So for example we could now use `np.pi` to use numpy's value for pi. Keep in mind that this is slightly different from the way your book uses import statements. Our way might technically be slower overall, but we won't notice and this will allow us to explore things more easily.

We used a variable in Day 2, but a variable is simply a name or letter that stores a reference to another object in python. I have to *declare* a variable in order to use it, and in python that means I have to give it a value (even if it won't keep that value). `x` vs `x=5` vs `x=y`.

Also, python is very limited in the math that it can do for you. `x = 5*10` works just fine but `x=y/2` does not nor can python solve for another variable like `y`.

But something like `x = x + 1` works just fine (as long as x has been defined already), even though it makes no sense to us. In fact, we will do operations like this all the time!

Let's do a few examples (and we'll cover comments along the way):

1. a ball dropped from a tower, given a height and time, tell where the ball is.
2. conversion from polar coordinates to cartesian

day 1

Quickly review what we talked about last time.

1. Powershell/Terminal, cd ~, ls, pwd, mkdir
2. Opening a jupyter lab instance in a directory where you want to store things

What we want to talk about today are the various ways that you can run a script with python. So we are going to look at

1. The interpretor
2. A python script
3. A jupyter lab/notebook session

We will mostly be using jupyter lab sessions as they are interactive and offer a lot of opportunity for experimentation.

We will also show along the way how variables work and how basic math operations work. Everything is mostly what you expect with the exception of exponents, where 2^5 would be written as 2**5 using python. We will also cover code comments along the way.

day 0

This is where I have to tell you what you should install. Technically, I hope that everyone could install the following software on their computer. These are listed in order of priority

1. Anaconda distribution of Python
2. Mathematica
3. Dropbox (create account, but also download and install)
4. Keepassxc (not essential but a good habit)
5. git (extra special sauce)

If you could begin an account with the following:

1. Dropbox (from above)
2. Overleaf (free, student edition)
3. Github (extra if you get git working)

Talk about AI. I want students to learn to use it as much as they learn to use the computer itself. But start off without it. If you do use it, but big, bright lines around what you got it to teach you.

I talked through the python versions of 1. PowerShell/Terminal 2. Script 3. Jupyter Notebook/Lab. We did a very simple 'hello world' as well as a for loop just to show the differences in each, and went over some typo level gotcha's that come up. End on Jupyter Lab and showed them how that is a nice mid point between the interpreter and a script.

For the interpreter, you should just use terminal on Mac/Linux, and use Powershell for Anaconda on Windows.

For the script, I talk through using a shebang like #!/usr/bin/env python. We looked at mistakes and I showed them how to use the Traceback to kind of find where the problem is, although this is not perfect.

For jupyter, we will but using lab not notebook but they are very similar and students should know how to use both.

Here is chapter 2 of the book: chapter 2

```
In [ ]:
```