Programming

Introduction

"The good news about computers is that they do what you tell them to do. The bad news is that they do what you tell them to do. " -- Ted Nelson.

The quote above highlights the most wonderful and annoying aspects of programming: namely, that you need to be very precise when you tell a computer what you want it to do, or else it won't work. In this lab, you will spend some time writing "pseudocode" (i.e. sentence fragments in the structure of a program without worrying about the syntax of any specific language) in your lab notebook, and then later translate that to real code on your computer. By the end of this lab, you will have the skills to fit a line to some data, a very common task in astronomy and science in general (but we'll wait until next week to actually do it).

Definitions

Computers think in discrete "yes/no" (or **`True/False`**) steps. They can only do one thing at a time, so any program you write must split up the desired task into easily digestible chunks that can be done step by step. Think about simple tasks you do everyday. How would you turn them into a program?

Vocabulary

`if, else`
- these words can be used to describe conditional statements
- "**`if`** this condition is met, do the following thing, otherwise (**`else`**) do this other thing"

`for, in`
- these words are useful if you want to do the same thing a specific number of times
- "**`for`** each value **`in`** this list of elements, do the following thing"

`while`
- this word is useful if you want to do the same thing an unknown amount of times
- "**`while`** this condition is met, do the following thing"

`print` (also **`return`**)
- put this word at the end of a program to output the answer
- "after this thing has been accomplished, **`print`** the answer"

Operations

Mathematical
- add (+), subtract (-), multiply (*), divide (/)

- these work the same way they always have: you can add, subtract, multiply, or divide numbers or variables

Comparison

- is greater than (>), is less than (<), is greater than or equal to (>=), is less than or equal to (<=), is equal to (==), is not equal to (!=)
- Compares the "truth" value of two things
- `1 < 2` evaluates to **`True`**, `1 > 2` evaluates to **`False`**
- `1 == 2` evaluates to **`False`**, `1 != 2` evaluates to **`True`**

Assignment

- equals sign (=), assigns a value to an object
- **`x = 5`** means "the variable **`x`** is assigned the value of **`5`**"
- note: different from how the equals sign is usually used!

Arrays

Arrays (or lists, or tables) are data structures that store a bunch of elements in a single object so you can access them more easily. You can "index" an array to get an element from that array: **`array[n-1]`** gives you the **`n`**'th value in the array. Note that this means **`array[0]`** gives the first element, **`array[1]`** gives the second element, and so on. This type of numbering is called "zero-indexing," and is used across virtually all programming languages.

Pseudocode [15 pts]

With this basic vocabulary, let's try writing some pseudocode to accomplish some quick tasks. For an example of what I'd like your "pseudocode" to look like, see the example below.

```
Task: How many elements are in an array?

1    Set the initial count of elements to 0
2    For every element in the array:
3        add 1 to the count of elements
4    Print the count of elements
```

Note three things: (i) please place each separate piece of the program on a different line, (ii) please number your lines, and (iii) after a **`for`** or **`if`** statement, place a colon and indent any following lines that depend on that statement.

In your notebook, write some pseudocode to accomplish the following tasks on an array called **`your_array`**:

1. [1pt] Is the value **`6`** in **`your_array`**?
2. [2pts] What is the largest value in **`your_array`**? What is the smallest?
3. [3pts] What is the average value of **`your_array`**?
4. [3pts] What is the standard deviation of the values in **`your_array`**?

5. [3pts] What is the median value in **`your_array`**? Assume that the array is sorted from smallest to largest.
6. [up to 5pts] Challenge: partial credit will be awarded! Do #5 again, but don't assume that **`your_array`** is sorted.

Python [5 pts]

Now that you have some pseudocode written in your lab notebook, let's turn it into real code on your laptop! The reason we are using Python is twofold. First, it's very common in astronomy, and second, its syntax is much more straightforward and almost reads like English.

If you have Anaconda installed already (let me know if you're having trouble with this), download "Lab_4.ipynb" from the Google drive folder to your computer. Open up a terminal (on a Mac, click "Launchpad," then "Other," then "Terminal"). Type "**`jupyter notebook`**" (no quotes). A page should open in your internet browser. Navigate to wherever you saved the file you just downloaded (probably "Downloads") and click it to open it.

You should see some space to type answers to the problems: do so! Don't worry, I will go over some syntax, but I purposely had you write pseudocode that was very similar to Python's syntax to begin with, so there shouldn't be a whole lot you need to adjust.

Now let's run some programs! To do that, click in the "cell" you want to run and then hit "Shift+Enter". Does it work? If not, what do you need to change? Try to take the time necessary to understand what the program is doing and why it works. We will be using these kinds of principles and skills in subsequent labs.

Once you are satisfied with what you have typed up, change the name of the notebook to "Lab_4_FIRSTNAME" (with your first name replacing FIRSTNAME) and email it to me.

Before you leave…

What did you like or dislike about this lab? Any suggestions? Leave comments on this document and/or in your lab notebook.