DAT-119 – Python 1

Spring 2019

**In-Class Assignment Week 8**

This week we’re going to practice the skills we learned in the last four weeks:

1. Conditionals (if/elif/else)
2. Repetition (for, while)
3. Functions
4. Lists

And we’ll remind ourselves how to use GitHub, as well.

We’re going to work on a program in class: a todo list application. (If you prefer to call it “a to do list” instead of “a todo list,” that’s fine. Microsoft Word agrees with you, for what that’s worth.)

**Our class schedule for tonight is as follows:**

* I’ll say hi and describe the problem (also written out below), showing you what the output looks like in my implementation.
* You’ll get some solo time to think through how this program should be designed, with special attention to what functions you think need to be implemented.
* In small groups of 2-3 you’ll discuss your planned implementations and share ideas.
* You’ll take some time to plan a little more, if needed, and write code – this is largely individual work, but you are also welcome to help one another, as issues arise. And of course, I’ll be here to help, too.
* In the last 20 minutes of class your small groups will reconvene and share what you’ve learned while working on this project, so far.

**To turn in at the end of class (no code, this time):**

1) your planning doc,

2) a list of who you worked with, and

3) a couple of sentences about how the project is going for you so far.

You’ll submit the finished project as homework, via a link to GitHub, along with your reading for the week.

**Problem specification:**

We’re going to create an application to track a user’s todo list!

Our application will maintain at least two lists:

1. items that need to be done and
2. items that have already been completed.

When the user starts the program, they will be presented with a menu of options, including the option to view either list, to add a new item to the todo list, to mark an item as completed, or to exit the program.

I recommend numbering the choices in the menu to provide the user with an easy way to choose their preferred option, but if you can think of an equally user-friendly way for them to make a choice (letters, for instance), feel free to implement it that way.

When an item is marked completed, it is removed from the todo list and is added to the list of items that have been completed.

The program should not end until the user tells it to end. Imagine, if you will, that the user just keeps this program open in their terminal for days or weeks, until everything they have to do has been completed. (Imagine, *ever completing all of the things on your todo list*. That sounds nice.)

Whenever the user starts up the application, both lists will always be empty, and when they close the application, they’ll lose their lists … for now, but spoiler: we’ll fix this next week.

In addition to showing you how the program should run at the beginning of class and being available to answer clarifying questions, I’ve also captured sample output and added it to a separate Word doc, which you can examine at your leisure.

This is a fairly formal project – think of it as a dress rehearsal for the final project at the end of class – so I expect input validation, adherence to the Style Guide (available in Course Documents), and user-friendly prompts and behavior.