EECS 1516 Lab 1: Getting Started

Due: January 23, 9 p.m.

1 Introduction

Welcome to your first EECS 1516 lab! :)

Labs will be your opportunity to practice the course material with your friends and TAs, so you can get assistance when you have a question. Don't hesitate to ask questions of each other or of the TA. However, do not share code with your peers. You are submitting your code to PrarieLearn and we may be checking for plagiarism. If you do use AI to complete you labs note that you are doing yourself a disservice; reliance on AI will make the practical exams that much harder!

The purpose of this lab is to refresh your memory of Python. You will have two weeks to complete this lab, so that you have time to troubleshoot any installation or setup errors you may encounter. This lab description is a little long, as it includes some information about some of the data and tools we will use moving forward.

2 Getting Started

To program in Python, we will be using an integrated development environment (IDE) for the Python language. The IDE we will use is called **PyCharm**. You'll need to set up PyCharm so that you can program in the latest version of Python, which is **Python 3**.

For testing, we will be using a module called **doctest**. This module facilitates testing of Python code; you will be using this frequently.

You can download PyCharm at this URL:

https://www.jetbrains.com/pycharm/download

You can use the community edition, which is free. PyCharm will also be available for you to use in the labs.

You will also need to enroll on PrarieLearn, as this is where you will be submitting completed labs. You will submit this lab at the following URL:

https://us.prairielearn.com/pl/course_instance/200825/assessment/2625070

3 Programming Task

In this lab and a few others we will be looking at information from a city inventory of trees. The data is publicly available and part of Mississauga's 'Open Data' initiative, and they can be accessed here:

https://data.mississauga.ca/datasets/65e5b975dea04bba9b9fcfe8066ba7f3$_0$*/explore*

The city of Toronto maintains an analogous tree inventory here:

https://open.toronto.ca/dataset/street-tree-data/

Why do cities take these inventories? This is because the inventories inform decisions, including (but not limited to):

- Development decisions. The distribution and health of our trees help us decide where to plant new trees and where to implement green spaces.
- Environmental decisions. Our trees absorb carbon dioxide, provide oxygen, nurture a diverse ecosystem, contribute to our well-being, and mitigate the effects of pollution.
- Maintenance decisions. Inventories help us identify diseased or damaged trees, plan for effective stormwater management, and allocate resources for tree care.

In our lab we will be reading information about trees from a CSV (comma separated value) file and then representing these trees as lists of lists in the Python programming language. We will also write some simple routines to operate on lists of lists. In future labs, we will represent individual trees as software objects. These software objects will be used to encapsulate attributes and methods that are specific to a tree, like its species or size.

When we write code in our class, we will be encouraging a test-first programming paradigm. This means we encourage work flow that involves:

1. Studying the specification;
2. Reviewing the existing unit tests (e.g. doctests) and writing a few of your own;
3. Implementing the methods that are requested;
4. Revising your implementation until it passes the tests;
5. Submitting your work.

For this lab, we specifically ask that you write one test to supplement each that has been supplied.

You will find Lab 1 starter files on eClass and you can submit your completed code at the PrarieLearn URL listed above. The files in the starter package are as follows:

user_code.py and treelist.csv

We suggest that you work on your solution within PyCharm on your local machine, and then cut and paste your completed *user_code.py* into the appropriate box on PrarieLearn, where it will be marked. You do not need to submit *treelist.csv.*

Each method you must implement is described in brief below; more details and initial unit tests can be found in the starter code.

1. The method *read_data* will read data from a CSV file and return a list of lists. Each inner list represents a row of data in the CSV file.
2. The method *get_tree_names* will return a list of tree names that are extracted from a list of lists. The input is assumed to be a list of tree data, where each inner list represents data about an individual tree.
3. The method *to_lower_case* will accept a list of tree names, and return a list containing the same tree names converted to lower case.
4. The method *get_tree_coords* which will return a list of tree coordinates that are extracted from a list of lists. As before, the input is assumed to be a list of tree data, where each inner list represents data about an individual tree. The last two entries in each list of tree data are the longitude and latitude coordinates of the tree. These coordinates will be represented as strings.

4 Testing your Code

In order to receive full marks, we are also asking you to write at least one of your own tests for each of the methods listed above. When you write your own tests, consider edge and corner cases. What happens when the argument to *to_lower_case* is an empty list, or *read_data* is asked to open a non-existing file? Are you confident that your code will still work?

Make sure you can pass all of the tests we have provided as well as those you create before you submit.

5 How to Submit

To submit, you will cut and past the code you write in *user_code.py* into the Lab 1 instance of PrarieLearn. The score you get on PrarieLearn is the score you get for the lab!