McGill initiative in Computational Medicine

Workshop outline

Intermediate Python Skills

Instructor: Benjamin Rudski (benjamin.rudski@mail.mcgill.ca)

Registration link: https://forms.gle/XfDqMJDXM9v2RCQYA

Hours of instruction: Tuesday, October 31, 2023 – 13:00 to 17:00 (4 hours)

Prerequisites:

- Introductory Python programming experience required (or programming experience in another language and comfort with Python syntax).
- To be able to fully participate, attendees must have a local Python installation with the ability to run Jupyter Notebooks, as well as another Python IDE. I will be using Microsoft Visual Studio Code with the Python extension. I highly recommend that participants use this software, as well. Alternatively, PyCharm Professional can be used. Participants without a local installation of Python can still follow along most of the workshop using Google Colab but they will be unable to perform certain tasks which require a local terminal.
- Basic knowledge of the Unix command line is an asset to be able to understand the command-line section of the workshop. Most of the workshop can be completed without this knowledge and I will review the key concepts, but having this knowledge in advance is an asset.

Summary:
In this workshop, you will learn how to go beyond Python basics. Through a small project, we will learn more advanced programming topics, such as functions, classes and using packages. By the end of this workshop, you will be able to write command line Python scripts to accomplish complicated tasks.

Contents:

I. Module 0: Introduction and Problem Scenario **(10 min)**
   a. Introducing DNA sequence processing
      i. This “project” will be a running example that we will slowly build over the course of the workshop. The idea is that we will take multiple FASTA files containing DNA sequences, translate them to proteins and perform some simple analysis on them.
   b. Why do we need scripts?
II. Module 1: Functions **(45 minutes)**
   a. Writing custom functions
      i. Function parameters

   ii. Variable scope
   iii. Function return values
   iv. Function documentation and type hints
b. Hands-on activity: Write protein expression functions.
c. *By the end of this module, we will have working functions to transcribe DNA to RNA and then translate RNA to a protein sequence.*

III. Module 2: Classes and Object-Oriented Programming **(1 hour)**
   a. Introduction to classes and objects
      i. What are classes?
      ii. What are objects?
      iii. What's the difference????
   b. Writing classes – Gaining a sense of `self`
      i. Attributes and initialization
      ii. Methods
      iii. Documentation
      iv. Hands-on activity: Writing classes to represent biological sequences.
   c. Data classes
      i. What are data classes?
      ii. Using data classes
      iii. Hands-on activity: Updating out biological sequence classes.
   d. *By the end of this module, we will have classes for DNA, RNA and protein sequences, as well as simple methods for each (e.g., for DNA, we will have a .transcribe() method, for RNA we will have a .translate() method and for proteins, we will have a method that gets the amino acid properties).*

IV. Module 3: Packages **(45 minutes)**
   a. Installing packages
      i. Finding packages online (GitHub, PyPI)
      ii. Installing packages using `pip`
      iii. Installing packages using `conda`
   b. Using packages
      i. Importing packages
      ii. Using NumPy (includes reading documentation)
   c. Hands-on activity: Use NumPy to analyse the properties of multiple similar, but slightly different sequences. *By the end of this module, we will have functions that use NumPy to analyse the different sequences (e.g., looking at the relative frequency of different nucleotides or amino acids at each position in a sequence).*

V. Module 4: Working with the Operating System and the User **(1 hour)**
   a. Interacting with the operating system
      i. Working with text files
      ii. Hands-on activity: Write code to read and write FASTA files.

iii. Files and paths: `os` and `shutil`
iv. Hands-on activity: Write code to copy all the FASTA files from a folder to another folder, read and translate them, then write the amino acid sequence to a new FASTA file with the same header in a new folder called “proteins”.

b. Basic scripts
i. Running Python files from the command line
ii. Hands-on exercise: Move the code we’ve been working on in the Jupyter Notebook to a Python script.
iii. Adding command line arguments using `argparse`
iv. Hands-on exercise: Update the script to accept the input sequence directory and output directory as command line arguments. Add optional arguments, like maximum number of sequences to process or maximum sequence length.

VI. Module 5: Where to go from here **(10 min)**
a. Next steps (topics to mention):
i. Enumerated types
ii. Class inheritance
iii. Working with data frames using Pandas
iv. Creating Python packages and distributing using PyPI, developing GUIs using PyQt5/6.

b. Important resources
i. Python documentation
ii. Online tutorials
iii. Stack Overflow