BE500 A3 - Programming Fundamentals for Biomedical Engineering Data Analysis with Python (Spring 2023)

Time: T/Th 11:00AM – 12:45PM **Location: LSE B03**

Instructor: James Galagan (jgalag@bu.edu) **TA: Aiden Riley**

Course Goals. The course provides an overview of the fundamentals of computation and programming for BME students. The course is designed for graduate students and senior undergraduates with minimal or no programming experience. The course will cover a range of topics - not typically found in a single programming course - that will prepare students for an array of common BME data analysis tasks, spanning:

1. Simple scripts for data analysis to standalone software for performing complex tasks
2. Simple mathematical calculations to more sophisticated computational algorithms
3. Parsing individual data files to processing large scale data sets
4. Outputs from scientific plots and formatted reports to serialized objects and reusable code

Experience with these tasks will prepare students for downstream classes in machine learning applied to BME problems.

The course will go beyond teaching the Python language to include a broader set of concepts common to software engineering in any language including an emphasis on algorithms, abstract data structures, and object-oriented programming as well as an introduction to data modeling, SQL, and data analysis pipelines.

Course Format: The course will be taught in a combination of (1) lectures/live demonstrations and interactive periods in class, and (2) homework assignments and a final programming project to put material learned into practice. Examples and assignments will be selected to illustrate common problems in BME.

Homework assignments will be used to practice real-life programming skills. Homeworks will be submitted through a course software repository and graded by checking out and running unit and functional tests. Assignments will build on previous assignments where possible to emphasis code reuse.

The final project will require students to develop a software package to analyze real BME datasets. The package will parse and load data, encapsulate data into appropriate data structures, and perform data analysis and plotting. The final project will build on concepts introduced in the class and reuse code developed throughout the class.

Textbooks: **(1) Learning Python** (5th Edition for Python 3), Mark Lutz, O'Reilly. This is the core textbook for Python proper. **(2) Introduction to Python for Science and Engineering**, David Pine, CRC Press. A textbook for engineering data analysis using Python. We will also refer students to other important references as we move into other topics.

Course Outcomes:

As an outcome of completing this course, students will:

1. Understand fundamental programming constructs and their implementations in python, and be able to more quickly learn other programming languages.
2. Gain experience setting up and using common tools for program development, debugging, management, and deployment.

3. Be able to parse common data files into appropriate computational data structures for effective data analysis.
4. Identify, formulate, and solve BME analysis problems by applying principles of mathematical, algorithmic, and software design.
5. Be able to efficiently develop accurate, reproducible, documented, and reusable software for BME analysis problems.
6. Be introduced to fundamental tools for data modeling, data management, and data analysis pipelines as applied to common BME research areas.
7. Be prepared for more advanced courses in machine learning and data analytics.

Course Topics Overview (subject to change):

1. Overview

Overview of how computers and software work

Using software to solve biomedical engineering problems

The Linux shell, file system, and shell commands

Different ways of running a python program

Outline of a python program

Layers of abstraction

The essential programming toolkit

Setting up your programming environment

2. Data Types, Variables, and Operations

The central role of data structures

Overview of variables and values

The Variable, Object, Reference Model

Expressions and statements

Numbers and their operations

Strings and their operations

Sequence operations

Lists and their operations

Tuples

Sequence assignment and unpacking

Dictionaries and their operations

Type casting - implicit and explicit

3. Functions

Compound statements

Overview of functions

Writing functions

Function arguments

Function best practices

Basic debugging

Unit testing

Common useful functions

4. Iteration and Conditional Flow

While loops

For loops

Iterator type

Loop coding techniques

Conditional execution

Boolean logic and truth tables

List comprehension

First introduction to recursion

More debugging techniques

5. Introduction to Algorithms

Introduction to algorithms

Pseudocode

Search and sort algorithms

Optimization algorithms

Techniques for debugging algorithms

6. Introduction to Classes, Objects, and OOP

Encapsulating data and functions together

Programming to behavior

Class coding fundamentals

Everything in Python is an object

Developing some core data structure classes

7. Modules, Packages, and Documentation

Overview of Python program architecture

How modules and packages work

Module and package coding basics

Python documentation model and tools

8. Plotting: The MatplotLib Module

Plotting with MatplotLib

Scientific plotting

Custom plotting functions

9. File I/O: Basics, Pickle and Pandas Modules

File IO

Parsing data files

Serializing data with Pickle

Pandas and dataframes

Working with series and tabular data

10. Graphs, Trees, and Tree Algorithms

Introduction to Linked Lists, Graphs, and Tree data structures

Tree construction

Tree editing

Tree traversal and search algorithms

Tree analysis algorithms

BME Examples: neuronal networks, molecular networks

11. Advanced Objects and OOP

Polymorphism and Inheritance

Principles of OOP design

Introduction to data modeling

Developing derived classes from earlier classes

Biomedical data modeling examples

12. Numerical Calculations: The SciPy and Numpy Module

Arrays, multi-dimensional arrays, and tensors

Regression/curve fitting

Linear algebra, eigenvalues/vectors, Jacobian

Computational graphs

Forward and back traversal on computational graphs

13. Advanced Topic 1- An Introduction to SQL databases

What is a database and why you should use one for your data

MySQL

Basic SQL syntax – making hard data analysis easy

How to query SQL from Python

14. Advanced Topic 2 – Data Analysis Pipelines

The role of data analysis pipelines

Robust pipeline principles

Introduction to Bpipe

Examples from biomedical engineering and bioinformatics