**Bi 410L/510L**

**Introduction to Programming for Biologists**

This class is a hands-on introduction to practical programming skills for biologists. Using data sets from a variety of areas in the life sciences, students will write Python programs that read data files, define functions that carry out mathematical operations (e.g. compute the frequency of a wave recorded by an electrode or calculate state transitions in an epidemiogy model), and visualize the results. Projects will also use popular data science libraries like scipy and pandas for more advanced operations.

We will use Jupyter Notebooks to create “living documents.” Notebooks contain both Python code and documentation that explains what the code does and how it works. A person who reads a scientific paper written as a Jupyter notebook can not only read about the hypotheses and methods, they can run the code themselves to see results and visualizations, which are inserted automatically into the notebook.

**Course Information**

*Instructor:* John Conery (conery@uoregon.edu)

*GEs:* Mandie Driskill (mdriskil@uoregon.edu)  
Mahboubeh Khoddam (mkhoddam@uoregon.edu)

*Office Hours:*  
John 281 Onyx, Mon 1–2  
(Zoom link on Canvas) Wed 10–11, or by appointment  
Mandie TBD  
Mahboubeh TBD  

*Lectures:*  
175 LIL  
MW 4:00 – 5:20

*Labs:*  
130 HUE  
F 12:00 – 13:50  
F 14:00 – 15:50

*Textbook:* None

*Prerequisites:* None (no prior computer programming experience is necessary)
**Canvas**

We will use Canvas to distribute projects and milestone exams (described below).

Students will upload completed projects and exams through Canvas, and we will post scores and feedback on each submitted item.

Other things posted on Canvas include:

- instructions for installing software used in the course
- links to on-line references for Python and other software
- an FAQ document with answers to common questions

**Important:** We will also post announcements on Canvas, so make sure you check Canvas regularly (or better yet configure the system so you are automatically notified when there is a new announcement).

**Grading**

Grades will be based on the total number of points earned during the term. There are three ways to earn points:

- **programming projects** that introduce computing skills and give students a chance to practice those skills
- a series **milestone exams** to assess concepts and skills introduced in projects
- short **in-class exercises** and other projects, some of which may involve pre-class reading assignments

**Group Programming Projects**

The course topics have been divided into five separate units. Each unit will have a programming project based on the material in that unit. For these projects students are encouraged (but not required) to work in small groups of two or three people.

Each person in the group will receive the same number of points. Groups can revise and resubmit as often as they wish, correcting any issues identified by the graders.
Milestone Exams

There will be five milestone exams, one for each unit. The exams are designed to test how well a student has learned the concepts and skills from that unit, based on short programming exercises similar to the exercises in that unit’s programming project.

Exams are not group projects; each student must take their own exam.

Exams will be posted on Canvas as timed quizzes. Students can take an exam any time after they have completed the corresponding programming project. Students will have 90 minutes to complete the quiz and upload their solution to Canvas.

Note: A student can repeat an exam, up to a maximum of three attempts, and we will use the highest score.

Individual Programming Project

Individual programming projects will give students a chance to work with real data by developing a complete Jupyter notebook that analyzes and displays the data.

As with the group programming projects, students can revise and resubmit notebooks to correct issues and improve their scores.

In-Class Exercises

Throughout the term there will be other opportunities to earn points by participating in group programming exercises and discussion sessions that will take place during lecture periods.

Most of these exercises will involve some pre-class reading. The readings and the exercises will be published on Canvas a few days before class.

No Midterm or Final Exams

There will be no midterm exam or final exam.
**Deadlines**

This course is entirely self-paced. There are no deadlines for group projects or milestone exams. Groups and individuals can submit (or re-submit) programming projects at any time, and students can take milestone exams whenever they are ready.

Although there are no deadlines, students are encouraged to submit projects early so we can provide feedback. If programming projects are submitted by Wednesday at 5:00, we will grade the project and provide feedback so students can ask questions during that week’s lab session.

The final date for all items – group projects, milestone exams, and individual projects – is Monday, December 6, at 2:45 P.M. (the end of the final exam period scheduled for this class).

**Letter Grades**

The complete list of programming projects, exams, and exercises is shown in a table on the next page, along with the number of points that can be earned on each item. The total number of points available is 1400.

Letter grades will be assigned based on the number of points a student has earned throughout the term. This table shows the minimum number of points required for each letter grade.

<table>
<thead>
<tr>
<th>Grade</th>
<th>Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>700</td>
</tr>
<tr>
<td>C−</td>
<td>900</td>
</tr>
<tr>
<td>C</td>
<td>950</td>
</tr>
<tr>
<td>C+</td>
<td>1000</td>
</tr>
<tr>
<td>B−</td>
<td>1075</td>
</tr>
<tr>
<td>B</td>
<td>1100</td>
</tr>
<tr>
<td>B+</td>
<td>1125</td>
</tr>
<tr>
<td>A−</td>
<td>1225</td>
</tr>
<tr>
<td>A</td>
<td>1275</td>
</tr>
</tbody>
</table>
Graded Items

Items marked with a ☐ symbol are group projects.

In-Class Exercises (150 Points)

☐ 25 Marwick  
   How Computers Broke Science
☐ 25 GC content  
   group programming exercise
☐ 25 GPA  
   group programming exercise
☐ 25 Loop strategies  
   group programming exercise
☐ 25 Recombination rate  
   group programming exercise
☐ 25 Somers  
   The Future of the Scientific Paper

Programming Projects (725 Points)

25 Project 0  
   “warmup” project
☐ 100 Project 1  
   shell commands, iPython
☐ 100 Project 2  
   expressions and functions
☐ 100 Project 3  
   conditionals and lists
☐ 100 Project 4  
   loops and I/O
☐ 100 Project 5  
   pandas
200 Individual Projects  
   topics TBA

Milestone Exams (500 Points)

100 Milestone 1  
   shell commands, iPython
100 Milestone 2  
   expressions and functions
100 Milestone 3  
   conditionals and lists
100 Milestone 4  
   loops and I/O
100 Milestone 5  
   pandas

Reflections (25 Points)

25 Reflections  
   end of the term written assignment
**Academic Disruption**

In the event of a campus emergency that disrupts academic activities, course requirements, deadlines, and grading percentages are subject to change. Information about changes in this course will be communicated as soon as possible by email, and on Canvas. If we are not able to meet face-to-face, students should immediately log onto Canvas and read any announcements and/or access alternative assignments. Students are also expected to continue coursework as outlined in this syllabus or other instructions on Canvas.

In the event that the instructor of this course has to quarantine, this course may be taught online during that time.

**Academic Misconduct**

The following passage from the University of Oregon Student Conduct Coordinator defines expectations common to all courses at UO:

*The University Student Conduct Code (available at conduct.uoregon.edu) defines academic misconduct. Students are prohibited from committing or attempting to commit any act that constitutes academic misconduct. By way of example, students should not give or receive (or attempt to give or receive) unauthorized help on assignments or examinations without express permission from the instructor. Students should properly acknowledge and document all sources of information (e.g. quotations, paraphrases, ideas) and use only the sources and resources authorized by the instructor. If there is any question about whether an act constitutes academic misconduct, it is the students’ obligation to clarify the question with the instructor before committing or attempting to commit the act. Additional information about a common form of academic misconduct, plagiarism, is available at https://researchguides.uoregon.edu/citing-plagiarism.*

We want to emphasize the point about documenting and acknowledging sources of information. You will find everything you need to complete all the projects and exams in the lecture notes we post on Canvas. However, we readily acknowledge that students can benefit from finding additional information on the internet, including books, tutorials, and community sites like StackOverflow. You are welcome to use any information you find, but be very careful to explain what you found and how you are using it in your own notebooks.
Learning Outcomes

I = introduce  
D = develop  
M = master  

Be comfortable using a command line interface to organize and work with files [D]  
Know how to implement simple functions in Python [D]  
Understand basic data structures and control flow in Python, how to apply them [D]  
Learn general techniques for developing, testing, and debugging software [I/D]  
Be able to find, install, and use Python modules [D]  
Basic knowledge of Pandas (Python’s data analysis library) and Matplotlib (Python’s data visualization library) [I]