**Bi 410/510**

**Introduction to Programming for Biologists**

This class is a hands-on introduction to practical programming skills for working with biological data. We use five fully-worked projects from ecology, evolutionary biology, molecular biology, and neuroscience as running examples throughout the term. Motivated by these examples, students will learn how to define functions that carry out mathematical operations (e.g. compute the frequency of a wave recorded by an electrode, the points on a psychometric curve, or state transitions in an epidemiology model), read data from files, and use popular data science libraries like *scipy* and *pandas* to analyze and visualize the data.

We will use Jupyter Notebooks to create “living documents.” Notebooks contains both Python code and documentation that explains what the code does and how it works. When a person using the notebook runs the code in the notebook results and visualizations are inserted automatically into the notebook.

**Course Information**

All lectures, lab sessions, and office hours will be online meetings managed with Zoom.

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

*GE:* Stacy Levichev (aleviche@uoregon.edu)

*Office Hours:* John: Mon 1–2, Wed 10–11, or by appointment

Stacy: TBD

*Lectures:* MW 4:15 – 5:45

*Labs:* F 10:15 – 11:45 and 12:15 — 1:45

*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:

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

**Programming Projects (500 Points)**

There will be five programming projects, each worth 100 points. The first project will be posted on Canvas the first week of class, the others will be released approximately every other week.

For these projects students are encouraged 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.

Projects are prerequisites for milestone exams and must be completed before taking the corresponding exam.
**Milestone Exams (750 Points)**

There will be five milestone exams, worth 150 points each, that test how well a student has learned the concepts and skills from a 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.

For more information about milestone exams see the FAQ section on Canvas.

**In-Class and Pre-Class Exercises (250 points)**

Throughout the term there will be other opportunities to earn points by participating in group projects. Some exercises will be based on reading assignments, others will be in-class group programming exercises.

There will be ten pre-class reading assignments as part of “flipped lectures” throughout the term. Each of these will be worth 10 points. To earn points, students will be expected to have done the reading before class, and then answer a question presented in lecture (answers will be uploaded to Canvas during the lecture).

The remaining 150 points can be earned by 25-point exercices spaced throughout the term.

**No Midterm or Final Exams**

There will be no midterm exam or final exam.

**Deadlines**

This course is entirely self-paced. There are no due dates during the term. Groups can submit (or re-submit) projects at any time, and students can take exams whenever they are ready.

If a group submits a project by Wednesday at 5:00, we will grade the project and provide feedback so the group members can ask questions during that week's lab session.

The final date for submitting projects is Monday, December 7, at 4:45 P.M. (the end of the final exam period scheduled for this class).
**Letter Grades**

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>500</td>
</tr>
<tr>
<td>C−</td>
<td>800</td>
</tr>
<tr>
<td>C</td>
<td>900</td>
</tr>
<tr>
<td>C+</td>
<td>1000</td>
</tr>
<tr>
<td>B−</td>
<td>1050</td>
</tr>
<tr>
<td>B</td>
<td>1100</td>
</tr>
<tr>
<td>B+</td>
<td>1200</td>
</tr>
<tr>
<td>A−</td>
<td>1300</td>
</tr>
<tr>
<td>A</td>
<td>1350</td>
</tr>
</tbody>
</table>

**Learning Outcomes**

I = introduce  
D = develop  
M = master  

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]