Programming for Biology, BIOB 488, 3 credits, CRN 34808
Tuesdays and Thursdays, 2:00pm-3:20pm, Health Sciences 114

Instructor: John McCutcheon, john.mccutcheon@umontana.edu

Office Hours: By appointment, email John.

Textbooks: Required: *Practical Computing for Biologists*, Haddock & Dunn, 2011.
Optional: *Processing*, 2nd Ed., Reas & Fry, 2014.

Overview: Cheap, easily accessible DNA sequencing has transformed biology. For a couple of thousand dollars, individual researchers can generate more data in a week than the entire Human Genome Project generated in ten years. However, while the generation of these large data sets is routine, the analyses of these data are not. This course is aimed at teaching students the skills required to manage, analyze, and display large biological data sets, including (but not limited to) genomic data.

Approximately 2/3 of the course will be taught in the UNIX environment using the programming language python. Students will write on average one new program a week, using genomic data sets as examples. We will cover the use of regular expressions (pattern matching), data types and structures, program control using logic and loops, reading and writing files, and python modules for biology. In the last ~1/3 of the course, we will explore large data visualization using Processing, a programming language originally designed for visual artists. For graduate students, this last part of the class will involve using real datasets (from your own work, ideally) as raw material to generate publication-quality figures using some combination of python and Processing.

This course is intended for advanced undergraduates and graduate students. The prerequisites are either completion of BIOB 486, Genomics, or consent of instructor. Students already working with large data sets—whether genomic in nature or not—are encouraged to bring these data to the course. Because this course will be taken by students with varying levels of experience, needs, and expectations, I reserve the right to deviate substantially from this syllabus if I feel that certain topics need more or less time. If this makes you uncomfortable, I encourage you to drop the course sooner rather than later.

Learning Outcomes: I want you to learn the fundamentals of computer programming. Programming is a field all on its own, with rich traditions and deep theoretical underpinnings, and so we cannot hope to cover even a tiny fraction of it in a semester-long course for non-specialists. The goals of the course are for you to understand the basics of the python programming language, regular expressions, basic data types, program logic and control, reading and writing files, python modules, and rudimentary visualization of large genomic datasets using Processing. In general, you will learn this content by **doing**—that is, actually working out problems and writing your own code. I will lecture as little as possible, and as such this course will be more like a lab course than a didactic course.

Grading: 60% Approximately 8 programming exercises
20% Final programming project
10% Two midterm exams (5% each)
10% A few homework assignments, mostly at the start of the course

Final grades will be based on your total points as a percentage of the total points possible. Pluses (+) and minuses (–) will be used (A, A–, B+, B, B–, C+, C, C–, D+, D, and D–) in the assignment of letter grades will be determined by the distribution of total scores, following these guidelines:

>90% of points (540): A- or better
>80% of points (480): B- or better
>70% of points (420): C- or better
>60% of points (360): D- or better

These cutoffs may be adjusted downward (in favor of the student).

Graduate Increment: The assignments and grading for undergraduate and graduate students will be the same on the homework assignments, exams, and programming assignments. Graduate students will be required to incorporate and integrate data from their own research with at least one other publicly available database (genomic, physiological, environmental, etc.) into the final programming project. This project must be accompanied by a 4 page referenced paper that contains a general introduction (1 page), scientific motivation for the study (1 page), rational for the programming methods used (1 page), and discussion of findings (1 page). For graduate students, the final programming project must include a visualization component that results in a publication-quality image or video.

Accommodations: to ensure accessibility of students with disabilities will be gladly made, but to qualify you must be registered with Disability Services for Students (DSS). Arrangements for accommodations on exams must be made through DSS.

Late work policy: This class will cover a lot of ground, and will require you to keep up with the assigned reading and assignments. If you have a problem understanding the material, or with turning an assignment in on time, I strongly encourage you to speak with us as early as possible. In general I won't accept late work, but I am sympathetic and reasonable if you deal with me in an upfront and honest manner and do not wait until the last minute to explain your situation.

Academic misconduct will be reported and handled as described in the University of Montana Student Conduct Code. All students must practice academic honesty. Academic misconduct is subject to an academic penalty by the course instructor and/or a disciplinary sanction by the University. All students need to be familiar with the Student Conduct Code: http://life.umt.edu/vpsa/student_conduct.php

The work you turn in should be your own. You free to discuss any aspect of the course with us or your classmates, including questions on the homework and programming assignments. But at the point when you begin formulating your answer on the computer, the work must become completely your own. Every individual develops a programming style, and it is surprisingly easy for me to see cheating when looking at code. As you will see from the very first assignment, there are usually several ways to get a solution when programming, and if I see solutions that are too similar I will ask the involved students about the incident; if no obvious explanation exists I will treat the matter extremely harshly. This may include receiving a failing grade for the entire course and filing a report with the Provost & Vice President for Academic Affairs. I don't expect this to be an issue with this course, but I do want you to know that I take plagiarism very seriously. If you are unsure about any of this, I urge you to ask us before turning something in.

Dropping course or changing grading status will strictly follow the University policies and procedures, which are described in the catalog. Please note that dropping the course or changing the grading status (to CR/NCR) is not automatically approved after the 30th day of the semester. These may be requested by petition, but the petition must be accompanied by documentation of extenuating circumstances. Requests to drop the course or change the grading status simply to benefit a student's grade point average will not be approved.