Syllabus
Computer Science 600.439/639
Computational Genomics
Fall, 2015
(3 credits, E)

Description
Your genome is the blueprint for the molecules in your body. It’s also a string of letters (A, C, G and T) about 3 billion letters long. How does this string give rise to you? Your heart, your brain, your health? This, broadly speaking, is what genomics research is about. This course will familiarize you with a breadth of topics from the field of computational genomics. The emphasis is on current research problems, real-world genomics data, and efficient software implementations for analyzing data. Topics will include: string matching, sequence alignment and indexing, assembly, and sequence models. The course will involve significant programming projects.

Prerequisites
Intermediate Programming (EN.600.120 or equivalent)
Data Structures (EN.600.226 or equivalent)

Instructor
Professor Ben Langmead, langmea@cs.jhu.edu, www.langmead-lab.org
Office: Malone 329, 410-516-2033
Office hours: Thursdays 11:00am – noon, and by appointment

Teaching Assistant
Ankit Garg, agarg8@jhu.edu
Office hours: Tuesdays 4:30pm – 5:30pm, Malone 239

Meetings
Tuesday, Thursday, 12:00–1:15 pm, Shaffer 101

Textbooks

Attendance
There are 23 lecture-based class sessions, 1 session for the in-class midterm exam, and 2 sessions for final project updates. Students auditing the course are required to attend all lecture-based sessions, barring illness or religious holidays.
Course Objectives

(1) Apply data structures, algorithms, analysis of algorithms, and algorithm design principles to problems in computational genomics
(2) Write programs to solve key computational problems in current-day genomics research
(3) Work effectively in a group, including group software development
(4) Communicate scientific results effectively
(5) Prepare to lead new research projects in computational genomics

Course Topics

These are listed in approximately the order they are discussed in the course.

• Background
  ○ What is Genomics?
  ○ What is Computational Genomics?
  ○ Genotype and phenotype, evolution, DNA, genome, RNA and protein
  ○ DNA sequencing and sequencing-by-synthesis

• Matching
  ○ Naïve exact matching
  ○ Boyer-Moore
  ○ Preprocessing and online versus offline
  ○ Approximate matching
  ○ Hamming and edit distance

• Indexing
  ○ Inverted indexing
  ○ Index-assisted matching
  ○ Tries and trees
  ○ Suffix trie
  ○ Suffix tree
  ○ Suffix array
  ○ Burrows-Wheeler Transform
  ○ FM Index

• Sequence alignment
  ○ Dynamic programming algorithm for edit distance
  ○ Global alignment
  ○ Local alignment
  ○ Time and space optimizations
  ○ String neighborhoods
  ○ Co-traversal

• Assembly
  ○ Coverage
  ○ Overlaps
  ○ Shortest Common Superstring
  ○ Overlap Layout Consensus
  ○ De Bruijn graphs
  ○ Eulerian walks
  ○ Error correction
  ○ Scaffolding

• Sequence classification
  ○ CpG Islands
  ○ Probability review
Course Expectations & Grading
There will be five or six homeworks, an in-class midterm, and a final group project. 50% of your final grade is from homework, 20% from midterm, and 30% from final project. There is no final exam, though the final project will be due during the final-exam period.

A grade of 97% or higher earns an A+. 93-97% or higher earns an A, 90-93% earns an A-, 87-90% earns a B+, etc, down to 70%. 60-65% earns a D and 65-70% earns a D+. Lower than 60% is a failing grade. Grades will likely not be curved. If a curve is applied, it will only be used to improve grades, not make them lower.

439 versus 639
The 600.639 section is somewhat more challenging than the 600.439 section. 600.639 has additional homework questions, additional midterm questions, and a higher grading standard for the final project. Only 600.639 can be counted for graduate CS credit. This is true even if you are currently an undergraduate and, for example, plan to enroll in the concurrent Bachelor’s/Master’s program.

Online Resources
See the Piazza site: piazza.com/jhu/fall2015/en600439639/home. This is the home for discussions, questions, announcements, homeworks, solutions, lecture notes, etc.

We will use the rosalind.info website for some of the homework questions. See below under “Rosalind.info Programming Assignments” for information about use of the site.

Homework Assignments
Homeworks include both programming problems and written problems. All solutions should be submitted via email to cs439@cs.jhu.edu, including solutions to rosalind.info problems. Written solutions must be in Word, PDF or plain text formats. Include the following in the email subject line when submitting: “EN 600.439/639 Homework (hw #) submission.” When submitting code in a compiled language like C/C++, include both source and an already-compiled binary. Inform the instructor/TA what command was used to compile the source code. You may re-submit freely. Your grade is based on your latest submission prior to the deadline.

Grading of Programming Assignments
When grading your software, the instructor/TA will run it on the undergraduate Linux cluster administered by the Computer Science department. If you cannot currently log into this cluster, ask the professor for the form to submit for temporary access. Please do this in the first week of class, as you will need access to complete the first homework.

You may use any programming language that runs on the CS undergraduate cluster. C, C++, Java, Perl, Python and Ruby are all acceptable. If you plan to use a language not on this list, please write to
Note that the language version on the cluster nodes may not be the same as the one you used while developing your software. For instance Java 1.6 and Java 1.7 are not the same. Test early and often.

Be aware of how much memory and disk space your software uses. Computer science cluster nodes are shared with other students and faculty, so you must be good citizens. Use the “top” command to monitor your program’s CPU and memory usage. Use the “du” command to measure the size of any final or intermediate files generated. For more information on these commands, type “man top” or “man du”. Also, note that the undergraduate cluster enforces a disk space quota.

Rosalind.info Programming Assignments
Some homework assignments for class will involve solving problems on the website http://rosalind.info. In the first week of class, please sign up for a Rosalind.info account then register for this “class” by going to http://rosalind.info/classes/enroll/d8381eea4c/.

http://rosalind.info has a mechanism for deciding whether you solved a problem. This helps you check your solution, but you must also submit your solution directly to us for a grade. Your program should accept input on the standard-in file handle, and write output on the standard-out file handle. In other words, when we run your solution we will do this:

```
your-executable < input.txt > output.txt
```

And you will be graded based on the content of output.txt. See your language’s documentation to learn how to access the standard-in and standard-out filehandles (it’s usually very straightforward).

When you are on a http://rosalind.info problem page (e.g. http://rosalind.info/problems/dna/) background information is provided at the top of the page but it is hidden by default. You should always expand this description and read it before starting in on the problem. The background material is helpful, with informative definitions and links.

Late Policy
Homework submitted after the deadline is late. The number of days late is calculated by rounding up to the nearest day. Each student has a total 5 “free” late days for the semester. Once these are exhausted, each day late incurs a 20% penalty on the late homework. No more than 2 free days can be used per homework. In other words, if a homework is 2+N days late, 20% is deducted for each of the N days, and you “spend” 2 of your 5 free days.

You must keep careful track of how many late days you have used. You do not need to ask permission to use late days. Note that some assignments (e.g. the one due before the midterm) may have restrictions on late day use. This is to ensure solutions can be posted in a timely fashion.

There is a separate late day policy for the final project, as described in the final project document.

Key Dates
See the Piazza “Schedule” post for all key dates.

Assignments & Readings
See the Piazza “Schedule” post for all assignments.

Ethics
The strength of the university depends on academic and personal integrity. In this course, you must be honest and truthful, abiding by the Computer Science Academic Integrity Policy.
Cheating is wrong. Cheating hurts our community by undermining academic integrity, creating mistrust, and fostering unfair competition. The university will punish cheaters with failure on an assignment, failure in a course, permanent transcript notation, suspension, and/or expulsion. Offenses may be reported to medical, law or other professional or graduate schools when a cheater applies.

Violations can include cheating on exams, plagiarism, reuse of assignments without permission, improper use of the Internet and electronic devices, unauthorized collaboration, alteration of graded assignments, forgery and falsification, lying, facilitating academic dishonesty, and unfair competition. Ignorance of these rules is not an excuse.

Academic honesty is required in all work you submit to be graded. Except where the instructor specifies group work, you must solve all homework and programming assignments without the help of others. For example, you must not look at anyone else’s solutions (including program code) to your homework problems. However, you may discuss assignment specifications (not solutions) with others to be sure you understand what is required by the assignment.

If your instructor permits using fragments of source code from outside sources, such as your textbook or on-line resources, you must properly cite the source. Not citing it constitutes plagiarism. Similarly, your group projects must list everyone who participated.

Falsifying program output or results is prohibited.

Your instructor is free to override parts of this policy for particular assignments. To protect yourself: (1) Ask the instructor if you are not sure what is permissible. (2) Seek help from the instructor, TA or CAs, as you are always encouraged to do, rather than from other students. (3) Cite any questionable sources of help you may have received.

On every exam, you will sign the following pledge: "I agree to complete this exam without unauthorized assistance from any person, materials or device. [Signed and dated]". Your course instructors will let you know where to find copies of old exams, if they are available.

In addition, the specific ethics guidelines for this course are:

(1) You may work together on homeworks but your submission must be your work alone. In other words, you can work together to figure out how to approach a problem, but you must do the problem by yourself, you must not look at another student’s solution, and you must submit only your own work.

(2) You may use code snippets that are presented in the lecture notes, or that are in any of the iPython notebooks associated with this class.

(3) An exception to the no-collaboration policy is the final project. For the final project, you are encouraged (possibly required) to work on a team. Your team will be required to report the contributions of the individual team members.

Report any violations you witness to the instructor.

You can find more information about university misconduct policies on the web at these sites:

- For undergraduates: [http://e-catalog.jhu.edu/undergrad-students/student-life-policies/](http://e-catalog.jhu.edu/undergrad-students/student-life-policies/)
- For graduate students: [http://e-catalog.jhu.edu/grad-students/graduate-specific-policies/](http://e-catalog.jhu.edu/grad-students/graduate-specific-policies/)

**Students with Disabilities**
Any student with a disability who may need accommodations in this class must obtain an accommodation letter from Student Disability Services, 385 Garland, (410) 516-4720, studentdisabilityservices@jhu.edu.

ABET Outcomes

- An ability to function effectively on teams to accomplish a common goal (d)
- An ability to communicate effectively with a range of audiences (f)
- An ability to use current techniques, skills, and tools necessary for computing practice (i)
- An ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices (j)