Computer Science 134: Introduction to Computer Science
Fall 2020
Prof. Daniel Aalberts (`daalbert@williams.edu`)
Prof. Duane Bailey (`bailey@williams.edu`)
Prof. Molly Feldman (`mqf1@williams.edu`)

Lecture Videos:	As posted on Glow
Discussion:	MWF 9:20-10:10am (Bailey, virtual) or MWF 12:00-12:50pm (Aalberts, in person)
Lab Sections:	Monday 1:30-2:45pm or 3:15-4:30pm (Bailey or Feldman, virtual) Tuesday 9:45-11:00am or 3:15-4:30pm (Aalberts, virtual)
Instructor Hours:	See course calendar
Assistants:	Alexandra Bonat, Amelia Chen, Kirun Cheung, Harun Curak, Caleb Dittmar, Scarlet Rusch, Benjamin Siu, Mira Sneirson, Nathan Thimothe, Jules Walzer-Goldfeld, Meredith Wolf, George Yacoub
TA Hours:	See course calendar
Python Resources:	`https://www.python.org/doc/` Allen Downey's *Think Python, 2ed*, an online textbook John V. Guttag, *Introduction to Computation and Programming Using Python,* 2nd edition, MIT Press.
Course web site:	`http://www.cs.williams.edu/~cs134/`
Technical Support:	Lida Doret (`lida@cs.williams.edu`) & Mary Bailey (`mary@cs.williams.edu`).

We are surrounded by information. This course introduces fundamental computational concepts for representing and manipulating data. Using the programming language Python, this course explores effective ways to organize and transform information in order to solve problems. Students will learn to design algorithms to search, sort, and manipulate data in application areas like text and image processing, scientific computing, and databases. Programming topics covered include procedural, object-oriented, and functional programming, control structures, structural self-reference, arrays, lists, streams, dictionaries, and data abstraction. This course is appropriate for all students who want to create software and learn computational techniques for manipulating and analyzing data.

Organization. Lecture material for this course will be delivered on-line, through Glow. During these lectures we investigate new concepts and problem solving strategies. Students are expected to have reviewed the lectures before attending scheduled synchronous discussion sections. There, we will discuss the material in greater detail, with a focus on applying learned techniques to sample problems. We expect a dynamic approach to class discussions that will allow us to steer our focus in directions of mutual interest.

Each week students will work on a larger, independently worked lab project. Students will check in, virtually, with the teaching staff during scheduled "laboratory" meetings to gauge progress and receive feedback. Teaching assistants will hold regular remote hours, focused on small group interaction.

Throughout the semester students will complete "weeklies": quick Glow check-ins with questions that will directly prepare students for quizzes. We will have three more formal Glow quizzes on October 7, November 11, and during finals.

Work. Students are responsible for watching pre-recorded lectures, and participating in discussion sections. Each week, students will complete either a Glow weekly check-in or a formal quiz. Most weeks will require students to complete a lab, a significant programming exercise.

Grading. Your final grade will be determined according to the following:

- Participation is expected.
- Glow "weekly" check-ins: 10%
- Glow quizzes: 30%
- Weekly lab assignments: 60%

Weekly labs will be graded on a 10 point scale. Excellent implementation of the bare minimum requirements of a lab will earn between 8.5 and 9. Higher grades will be awarded for demonstrating extended effort and dedication to thoughtful experimentation. We believe there is an aesthetic to presenting solutions to problems in Python. Students should be aware a portion of their lab grade is earned for writing beautiful programs that are easy to read and understand.

Tentative Schedule of Topics

Week of	Week's lab	Monday	Wednesday	Friday
Sept. 7		—	—	1. Hello, world!
Sept. 14	I. Python/Git workflow	2. Expressions	3. Conditions	4. Functions
Sept. 21	II. Phase of the Moon	5. Python Usage Modes	6. Iteration	7. Strings
Sept. 28	III. A Word Toolbox	8. Abstraction	9. Lists, Tuples	*Mountain Day?*
Oct. 5	IV. Debugging Strategies	10. Sets, Dicts	*Quiz*	11. Files & Comprehensions
Oct. 12	*Fall Break*	*Fall Break*	12. Mutability	13. Iterators
Oct. 19	V. Generating Text	14. Generators	15. Classes	16. Classes
Oct. 26	VI. Clustering Data	17. Classes	18. Special Methods	19. Recursion
Nov. 2	VII. Fractals	20. Recursion	21. Recursion	22. Inheritance
Nov. 9	VIII. Simple Ciphers	23. Inheritance	*Quiz*	24. Sorting.
Nov. 16	IX. Trivia!	25. Sorting	26. Sorting	27. Simulation
Nov. 23	*Break*	28. Simulation	*Break*	*Break*
Nov. 30	X. Simulation	29. Simulation	30. Simulation	31. Regular Expressions
Dec. 7	XI. Simulation II	32. Regular Expressions	33. Slack	34. Now what?

Quiz dates: October 7, November 11, and during final exam period.

Late Policy. You are expected to turn in all assignments in a timely manner receive full credit. Please contact us—*ahead of time*—if this is not possible.

Intellectual Property. As per College policy, no part of this course may be reproduced and/or distributed. In particular, no videos recorded as part of this class may be shared with anyone external to the CS134 course.

Community. We embrace diversity. We welcome all students and expect everyone to contribute and support a respectful and welcoming environment. If you have concerns, please share them with us or the college administration.

Students Who Need Accommodations. If formal accommodations need to be made to meet your specific learning or physical abilities, please contact one of us as soon as possible to discuss appropriate accommodations. Please also contact the Director of Accessible Education, Dr. G. L. Wallace (x4672) or the Dean's office (x4171). We will work together to ensure this class is as accessible and inclusive as possible.

Mental Health. Students experiencing mental or physical health challenges that are significantly affecting their academic work are encouraged to contact one of us or to speak with Dean's Office staff (x4171).

Honor Code. The Honor Code as it applies to non-programming assignments is outlined in the Student Handbook.

The honor code, as it applies to assignments in this class: You may discuss the labs and Python syntax with other students (e.g. Lab breakout buddies or in your specific Herd). However, each student should write up and hand in their own assignments separately. What you turn in should reflect your independent understanding after any collaboration.

Guideline: Discussion is allowed. Copying is forbidden. You should be able to articulate what every part of your program does. Prioritize understanding over finishing. Become good programmers.

If you do not understand how the Honor Code applies to a particular assignment, consult your instructor. Students should be aware of the Computer Ethics outlined in the Student Handbook. Violations (including uninvited access to private information and malicious tampering with or theft of computer equipment or software) are subject to disciplinary action.

The Department of Computer Science takes the Honor Code seriously.
Violations are easy to identify and will be dealt with promptly.