CS 151
Intro to Programming with Python

MWF 12:00pm, Ford 102
Spring 2026

Jed Rembold, Ph.D.
jjrembold@willamette.edu
http://willamette.edu/~jjrembold/classes/cs151
Ford 214
Office Hours: MW 4:30-5:30, TTh 2:00-4:00
Open Door: You are always welcome to contact me anytime online or when my office door is open!
Office Phone: (503) 370-6860

This syllabus is subject to change or adaptation as the semester progresses.

Course Description: The skills necessary to communicate with computers are becoming more and more imperative in the modern world. This course focuses on introducing concepts of computer science and programming to students through the use of the programming language Python. Fundamental concepts such as understanding variables, logic and loops will be covered, as well as object-oriented programming and working with graphical environments. Students should leave the course having a comfort and familiarity in writing Python scripts requiring several hundred lines of code to accomplish a variety of tasks.

Prerequisite(s): None
Note: A minimum grade of C- is required for this course to count toward gen-ed credit.
Credits: 4.0

Text: *Programming in Python*
Author: Eric S. Roberts
Availability: This book is available for free via PDF on the course website. While I will not assign problems directly out of the book, I'd highly recommend following along with the reading in the text, as it is tailored exactly to this course as we are teaching it. And it is free!

Course Objectives:
Over the semester, students will gain working knowledge in:

1. The fundamentals of coding: variables, logic and loops
2. Implementing fundamentals in Python to create basic programs
3. Applying decomposition and stepwise refinement to tailor a problem-solving strategy
4. Utilizing data structures to hold and model information
5. Debugging and testing programs to ensure they are working as intended

The field of computer science and programming is vast and the entirety of what is possible in Python could never be contained in a single course. This course seeks to provide solid fundamentals and confidence to students so that they might continue their learning on their own or in whatever direction their creativity might take them.

Grade Weighting

Perusal	5%
Attendance	5%
Problem Sets	15%
Projects	30%
Midterms	20%
Final Exam	25%

Letter Grade Distribution:

>= 92.00	A	72.00 - 77.99	C
90.00 - 91.99	A-	70.00 - 71.99	C-
88.00 - 89.99	B+	68.00 - 69.99	D+
82.00 - 87.99	B	62.00 - 67.99	D
80.00 - 81.99	B-	60.00 - 61.99	D-
78.00 - 79.99	C+	<= 59.99	F

Student Learning Objectives (SLO):
Upon completion of the course, students should be able to:

- Describe, design, implement and test structured programs to solve a problem. Problem-solving is at the heart of programming, and students must be able to a take a given problem, break it up into solvable steps, and implement each individual piece to achieve their solution.
- Explain what an algorithm is and how it relates to computer programming. Many types of problems share similar solutions. Knowing when they can be used, how to use them, and the benefits of one method over another is important.
- Recognize and construct common programming concepts, including variables, loops, functions, I/O, and logic. The bread-and-butter of basic programming. These are the tools in a student's toolbox from which students can pull to construct all their programs. Knowing them well leads to both increases in efficiency and creativity in how they can be used.
- Recognize the difference between various data structures like lists, dictionaries and tuples and decide the proper times to use each. Python has a variety of ways in which to store data. Choosing the correct one for a problem can eliminate many issues further down the line.

Course Assessment:

- **Homework**
  - There will be problem sets which will be due at 11:59pm on Monday of many weeks. Problem sets will comprise of 2-3 questions which require writing a program to accomplish a particular task. Starting template files are provided, and so students have simply to add the necessary code to the existing file to solve the objective. Problem set templates and submissions will be handled through GitHub Classroom, which we will discuss and demonstrate in class. Assignments will be posted on the class webpage each week and the provided link should be followed to download that week's assignment materials and templates. If you work with anyone on the homework, please provide their name at the top of the template in the designated line.
  - General Info: Programming is very hands-on, and the odds are high that you will not do well in the course if you do not practice! As far as I know, the best method to gain proficiency is solving problems and writing code. There are no real "shortcuts". The number and length of problems assigned is my best estimate for having you adequately practice and learn the material without being an excessive burden on your time. Working in groups and helping others is very encouraged, though students must turn in their own work. I highly recommend helping and instructing other classmates if you feel proficient on a topic, both to help them and because there is no better way to identify gaps in your own knowledge than when you attempt to teach something. *But clearly indicate who you worked with at the top of each problem!*
- **Projects**
  - There are 5 larger projects scattered throughout the semester. Unlike the problem sets, which tend to be more narrowly focused, the projects are a chance to pull from all the areas of computer science and programming that we have been discussing to form a cohesive and functional program. Because of their broader scope, all projects will be scored on a more holistic scale:

Score	Description
✚✚	An absolutely fantastic submission that goes far above and beyond the set requirements. Comes along maybe only a few times a semester, if at all.
✚	A submission that exceeds expectations. The program must reflect additional work beyond the requirements or get the job done in a particularly elegant way.
✔✚	A submission that satisfies all the requirements for the assignment. A job well done!
✔	A submission that meets the requirements of the assignment, but which is either stylistically weak or has a few minor problems.
✔▬	A submission that falls significantly short of the requirements of the assignment.
▬	A submission that shows even more serious problems but nonetheless shows some effort and understanding.
▬▬	A submission showing little effort and not representing passing work.

  A ✔✚ is the equivalent of a 95%, with other scores scaling above and below accordingly.

- **Small Sections**
  - To better facilitate students getting assistance, guidance, and role-models, we utilize a small section model in this course. Students will be split into small groups of 7-10 and assigned to a section leader. All section leaders are previous students who excelled in the course and are excited to help you do the same! Sections will meet for 1 hour with their section leader each week, wherein they will work through a few short problems and demonstrations. Section leaders will also serve as experts and peers that students can contact to ask questions and get extra help and guidance on problem sets and projects. Section attendance is mandatory and is graded as part of your course participation. Sections are designed to help you, and your attendance helps your peers. To further stress the importance of attending, we have a policy wherein you are allowed 3 unexcused absences over the course of the semester. Each unexcused absence after the 3rd will incur a 5% deduction on your final overall grade. Please show up! Should you not be able to attend your section some week, reach out to myself and your section leader, and we will endeavor to find you another section that you could attend that week. Sections will meet on Wednesdays and Thursdays, and a poll concerning your availability on those days will go out on the first day of classes.
- **Tests**
  - There will be 3 tests this semester: two midterms and one final. Exams will be open note and open book, but will not allow the use of the internet or any program to actually run code. This policy is a frequent cause of confusion for students, but it has been shown to actually improve student test scores, as students can focus on the larger picture of their code (which is what they are really being scored on) rather than getting hung up on one small error that they can't fix. There will be study guides and examples of old exams available before each exam, so students can feel prepared for the types of questions that may appear.
- **Perusal**
  - The flipped nature of this class can only be effective if you keep up with watching the content videos before each lecture section. Your viewing and interaction with these videos will be scored through Perusal, and is accessible through our Canvas page. Perusal has a lot of different methods for you to achieve full points, but they generally boil down to: watching the videos in their entirety and leaving or responding to comments that I or others have left on the videos. These comments are also a chance to highlight questions or points of confusion that you would like me to revisit or highlight in class! The goal is to not only ensure that you are absorbing the content before we work with it each day in class, but to foster camaraderie and community around the act of learning to code. It is far better that trying to do it alone!
- **Attendance**
  - Lecture hours will be split between group activities and discussions and live coding examples. They are meant to supplement and reinforce the content videos, and get you off to as strong a start as possible on each week's homework or project. Student's will check-in when they enter class to be placed in a group for the day, and will answer questions as part of that group. Just checking in and participating gets full points for the

day. In addition, there will frequently be a bonus question offered to groups, wherein correct answers count towards a bit of extra credit to those group members. I have restructured lecture sections significantly this semester to make them more dynamic and engaging, but that is only effective if you show up!

- **Contests**
  - In addition to the projects, there will be one or two programming contests scheduled at different points during the term. The point of these contests is to give you a chance to show some more creativity and personal initiative beyond what you might be able to show on projects or assignments. Rules for each contest will be released to the class when announced.
  - To encourage participation, an extra incentive is offered. Every reasonably serious entry gets you one virtual ticket in a random drawing for a grand prize at the end of the semester. Enter more contests, get more tickets! Getting runner-up or honorable mention in a contest will also grant you extra chances.

Course Policies:

Late Work Policy

I understand that sometimes things come up and you are unable to get an assignment in on time, and I strive to be incredibly flexible and accepting of late work. However, there also comes a point when you get too far behind to realistically keep up with the class, and just need to turn in what you have. In an effort to compromise between the two, my late policy allots you 3 cumulative days of late work throughout the entire semester. So you can turn 3 assignments in one day late, 6 assignments in 12 hours late, etc. without penalty. Once you have used up your 3 days (72 hours), projects are assessed a late penalty of one category point per extra late day used (a ✔✚ becomes a ✔, etc.), and problem sets a 20% penalty per day late. Late days can be precious, and it can really behoove you to keep some late days around for later in the semester when life gets busier.

AI Policy

Large language models (LLMs) and modern AI have taken the world by storm, and computer science and programming is certainly no exception. Modern tools like ChatGPT and GitHub's Copilot can pull together impressive programs given only a basic description and set of requirements. But they are not perfect, and thus humans are still left the task of ensuring the code that is produced actually does what was intended. As such, in this class, you are allowed, even encouraged, to use AI to assist your coding, *so long as you give it proper credit and realize that you are still responsible for the final output.* Know though that you will still be expected to produce code without AI assistance. You will still be expected to find errors in code without AI assistance. Modern LLMs can be terrific tools, but they are no substitute for programming language competency (at least yet!). Especially when learning, they can be more hindrance than help if used incorrectly, so proceed thoughtfully and with caution.

Incomplete Policy

An incomplete grade will only be granted in the case of prolonged illness or family emergencies that remove the student from the campus for an extended time period during the semester. Under no situations will an incomplete be granted due to a student falling behind through lack of motivation, understanding, or time management skills. If you are concerned about your

progress and how you are doing in the class, please come visit me! We can sort out where you are struggling and work out a plan to get you back on track.

Trans Inclusive Space

I will gladly honor your request to address you by your affirmed name or gender pronoun. Please advise me of this at any point in the semester so that I may make appropriate changes to my records. If I accidentally use an incorrect gender pronoun when addressing you, please feel free to let me know, in whatever manner makes you comfortable, that I messed up and explain or remind me of what pronouns you use so that I can make every effort to correct that error.

Willamette Policies:

Academic Honesty

Cheating is defined as any form of intellectual dishonesty or misrepresentation of one's knowledge. Plagiarism, a form of cheating, consists of intentionally or unintentionally representing someone else's work as one's own. Integrity is of prime importance in a college setting, and thus cheating, plagiarism, theft, or assisting another to perform any of the previously listed acts is strictly prohibited. I may impose penalties for plagiarism or cheating ranging from a grade reduction on an assignment or exam to failing the course. An instructor can also involve the Office of the Dean for further action. For further information, visit: http://willamette.edu/cla/catalog/policies/plagiarism-cheating.php.

This can be particularly problematic in programming courses, so know that I will be keeping an eye out for it. Do your own work, and always indicate if you have worked with someone else.

Classroom Conduct

As an educational institution, Willamette is committed to supporting the ideals and standards that help create a constructive and healthy learning community. That requires, among other things, encouraging positive classroom behaviors, discouraging disruptive classroom behaviors, and setting clear standards for both of those things.

To that end, constructive classroom behaviors are those that support learners and teachers in an environment that promotes trust, respect, and collaborative learning.

Disruptive classroom behaviors are those that undermine or interfere with individual's abilities to learn and teach. Clear examples of disruptive behaviors include, but are not limited to:

- Interrupting others or persistently speaking out of turn
- Distracting the class from the subject-matter or discussion at hand
- Any physical threat, physical, psychological, or sexual harassment, ridicule, or abusive act towards a student, staff member, or instructor in a classroom or related setting.

Time Commitments

Willamette's Credit Hour Policy holds that for every hour of class time there is an expectation of 2-3 hours work outside of class. Thus, for a class meeting three days a week you should anticipate spending 6-9 hours outside of class engaged in course-related activities. Examples include study time, reading and homework, assignments, research projects, and group work. This is a course that can often demand the full extent of that time, so plan accordingly!

Diversity and Disability

Willamette University values diversity and inclusion; we are committed to a climate of mutual respect and full participation. Our goal is to create learning environments that are usable, equitable, inclusive and welcoming. If there are aspects of the instruction or design of this course that result in barriers to your inclusion or accurate assessment or achievement, please notify me as soon as possible. Students with disabilities are also encouraged to contact the Accessible Education Services office in Smullin 155 at 503-370-6737 or accessible-info@willamette.edu to discuss a range of options to removing barriers in the course, including accommodations.

Tentative Course Outline:

The weekly coverage might change as it depends on the progress of the class. However, I highly recommend you follow along with the reading, as it makes a large difference!

Note that "assignments" starting with `S` followed by a number are just listing out what you'll be working on in your small section that week. Nothing actually needs to be turned in there.

Week	Date	Chapter	Description	Due
1	Mon, Jan 12 Wed, Jan 14 Fri, Jan 16		Karel the Robot Karel the Robot Karel the Robot	
2	Mon, Jan 19 Wed, Jan 21 Fri, Jan 23	 Ch 1	*MLK Day* Karel the Robot Introducing Python	Problem Set 0: Setup S1: Karel the Robot
3	Mon, Jan 26 Wed, Jan 28 Fri, Jan 30	Ch 1 Ch 2 Ch 3	Introducing Python Control Statements Algorithms/Debugging	Problem Set 1: Karel S2: Simple Python and Control Statements
4	Mon, Feb 02 Wed, Feb 04 Fri, Feb 06	Ch 7 Ch 7 Ch 7	Strings Strings Strings	Problem Set 2: Algorithms S3: Strings
5	Mon, Feb 09 Wed, Feb 11 Fri, Feb 13	Ch 7 Ch 4 Ch 4	Strings Simple Graphics Simple Graphics	Problem Set 3: Strings S4: Wordle
6	Mon, Feb 16 Wed, Feb 18 Fri, Feb 20	Ch 5 Ch 5 	Functions Functions **Midterm 1 (Ch 1 - 3, 7)**	Project 1: Wordle S5: Midterm 1 Prep
7	Mon, Feb 23 Wed, Feb 25 Fri, Feb 27	Ch 6 Ch 6 Ch 6	Writing Interactive Programs Writing Interactive Programs Writing Interactive Programs	 S6: Graphics
8	Mon, Mar 02 Wed, Mar 04 Fri, Mar 06	Ch 8 Ch 8 Ch 8	Lists Lists Lists	Problem Set 4: Graphics S7: Breaking out Pacman
9	Mon, Mar 09 Wed, Mar 11 Fri, Mar 13	Ch 8 Ch 10 	Lists Classes and Objects Imageshop Overview	Project 2: Breakout S8: Lists
10	Mon, Mar 16 Wed, Mar 18 Fri, Mar 20	Ch 10 Ch 10 Ch 11	Classes and Objects Classes and Objects Dictionaries and Sets	Problem Set 5: Arrays S8: Images
11	Mon, Mar 23 Wed, Mar 25 Fri, Mar 27	 	*Spring Break* *Spring Break* *Spring Break*	
12	Mon, Mar 30 Wed, Apr 01 Fri, Apr 03	Ch 11 	Dictionaries and Sets Catch-up or Review Day **Midterm 2 (Ch 4-6, 8)**	Project 3: ImageShop S9: Midterm 2 Prep
13	Mon, Apr 06 Wed, Apr 08 Fri, Apr 10	 Ch 12 Ch 12	Enigma Overview Data Structures Data Structures	 S11: Enigma
14	Mon, Apr 13 Wed, Apr 15	Ch 9 	Algorithmic Analysis *SSRD*	

Week	Date	Chapter	Description	Due
	Fri, Apr 17	Ch 9	Algorithmic Analysis	
15	Mon, Apr 20 Wed, Apr 22 Fri, Apr 24	Ch 9	Adventure Overview Algorithmic Analysis Fun Libraries	Project 4: Enigma S12: Adventure
16	Mon, Apr 27 Wed, Apr 29		Fun Algorithms	Project 5: Infinite Adventure S13: Final Prep (Optional)
17	Fri, May 08		**Final**	