CS 500
Bridge Course

Wed, Remote
Summer 2024

Jed Rembold, Ph.D.
jjrembold@willamette.edu
http://willamette.edu/~jjrembold/classes/cs500
Ford 214
Office Hours: By appointment virtually or just catch me online
Phone: (503) 370-6860

This syllabus is very 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 bridging the gap between where student are in their programming journey and where they need to be for the start of a Masters program: 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, working with graphical environments, and more advanced data structures. 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: This course is credit/no credit for all students.
Credits: 2.0

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, it can serve as an excellent resource should you be looking for different descriptions or examples than those shown in class or the videos. 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

Participation	50%
Weekly Projects	50%

Grade Distribution:

>= 70.00	Credit
<70.00	No Credit

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:

- **Projects**
  - There are weekly projects due each Thursday to allow you for some last minute questions in the following class. These projects are a chance to implement the various concepts learned and practiced in class to construct a more complicated and interesting application. All projects will come with a comprehensive guide and a breakdown of milestones to help you through the assignment. Projects are additionally a chance to explore on your own, and all projects come with potential extensions that you can pursue (or come up with your own!). 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 appropriately.

- **Participation**
  - Participation in class will be scored in two main ways. The first is that there will be regular "Understanding Check" or comprehension polling questions that students will have a chance to respond to. These can be freely discussed with your peers before submission, but will aim to check or confirm understanding of concepts laid out in the videos or lecture. The second half of class most evenings is going to be devoted to pair programming, wherein students will work alongside a peer to accomplish several different programming tasks. Completion of these tasks will earn students full points for that portion of the participation score.

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.).

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 latter portion of 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.

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 humans (at least yet!).

Classroom Conduct

As an educational institution, Willamette is committed to support 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 the 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
- Making unauthorized recordings or photos of a class meeting or discussion (except as permitted as part of an Accessible Education Services-mandated accommodation)

- 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.

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 of the College of Liberal Arts for further action. For further information, visit: http://www.willamette.edu/cla/catalog/resources/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.

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 hours 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.

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 Matthews 103 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!

Week	Date	Description	Due
1	Wed, Jul 10	Syllabus, Types and Operations	
2	Wed, Jul 17 Thu, Jul 18	Functions and Graphics	Roman Numeral Calculator
3	Wed, Jul 24 Thu, Jul 25	Interactive Graphics	Optical Illusions
4	Wed, Jul 31 Thu, Aug 01	Mutable Types	Breakout
5	Wed, Aug 07 Thu, Aug 08	Classes	Fingerprint Validation
6	Wed, Aug 14 Thu, Aug 15	Data Structures and Complexity	The Enigma Machine
7	Tue, Aug 20		Linked Labyrinths