CS 151
Intro Programming with Python

MWF, Ford 204
Fall 2021

Jed Rembold, Ph.D
jjrembold@willamette.edu
http://willamette.edu/~jjrembold/classes/cs151
Collins 311
Office Hours: MW 4:15-5:15, TTh 2:00-4:00, or catch me anytime online!
Phone: (503) 370-6860

This syllabus is subject to change or adaptation as the semester progresses. (Particularly the schedule...)

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 basic visualization. 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 university credit.
Credits: 1.0

Text: *Programming in Python*
Author: Eric S. Roberts
Availability: This book is available for free via pdf on the course website. You *will* want to be using the book throughout the semester to supplement what we talk about in class, so make sure you are using at least one of the versions!

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

1. The fundamentals of coding: variables, logic and loops
2. Implementing fundamentals in Python to create basic programs
3. Thinking algorithmically in approaching problems and problem-solving
4. Understanding the basics of object-oriented programming and when and why it is useful
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	5%
Problem Sets	15%
Projects	25%
Lab work	10%
Midterm	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 students 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 Sunday of most weeks. Problem sets will be a combination of a few theory type questions as well as some problems requiring programming. Solutions to theory questions should either by typed and saved as pdf or neatly (*and legibly*!) handwritten and photographed or scanned to pdf. Both pdfs and any code written will be submitted through Github Classroom before the deadline. We will discuss and demonstrate the process in lab. Assignments will be posted on the class webpage each week and the provided link should be followed to download that week's assignment materials. If you work with anyone on the homework, please provide their name at the top of the pdf or in a comment in your code.
  - 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.
- **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.

- **Lab Time**
  - Immediately following class on Wednesday is an hour of lab time. Generally most of this time will be devoted to an exercise that emphasizes and provides practice for what we talked about in class that day. To better accomodate the fact that these labs will be done remotely, finished exercises will be submitted through Github Classroom. Any remaining lab time is available for students to work freely on their homework assignment. If students are feeling comfortable, they are free to leave early during this second portion.
- **Small Sections**
  - To better get students assistance, guidance, and role-models, we are piloting a small section model this semester. Students will be split into small groups of 5-7 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 on a short problem. 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. There is one less lab per week this semester to account for the extra time you will be spending with your section, so you are expected to show up and participate.
- **Tests**
  - There will be 2 tests this semester: one midterm and one final. Exams will take place in class and will not allow the use of computers on the exam. 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.
- **Participation**
  - Participation in lectures will be graded. Questions will be asked in class and students will respond via polling technology. Simply responding to each question will earn all your participation points for the day, but answering correctly will earn you some extra credit. If you absolutely can not be in class, you can earn the base participation points by emailing me asking a question about a specific slide. Over the course of the semester, that extra credit can really add up, so I recommend showing up!
- **Contests**
  - In addition to the projects, there will be 1-2 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. 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). It can really behoove you to keep some late days around for later in the semester when the projects can get harder.

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.

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. An instructor may imposed 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.

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.

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 the professor 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	Chapter	Description	Due
	Mon, Aug 30		Karel the Robot	
1	Wed, Sep 01		Karel the Robot	
	Fri, Sep 03		Karel the Robot	
	Sun, Sep 05			Problem Set 0
	Mon, Sep 06		*Labor Day*	
2	Wed, Sep 08	Ch 1	Introducing Python	
	Fri, Sep 10	Ch 2	Control Statements	
	Sun, Sep 12			Problem Set 1
	Mon, Sep 13	Ch 3	Simple Graphics	
3	Wed, Sep 15	Ch 3	Simple Graphics	
	Fri, Sep 17	Ch 4	Functions	
	Sun, Sep 19			Problem Set 2
	Mon, Sep 20	Ch 4	Functions	
4	Wed, Sep 22	Ch 5	Writing Interactive Programs	
	Fri, Sep 24	Ch 5	Writing Interactive Programs	
	Sun, Sep 26			Problem Set 3
	Mon, Sep 27	Ch 5	Writing Interactive Programs	
5	Wed, Sep 29	Ch 5	Writing Interactive Programs	
	Fri, Oct 01	Ch 6	Strings	
	Sun, Oct 03			Project 1
	Mon, Oct 04	Ch 6	Strings	
6	Wed, Oct 06	Ch 6	Strings	
	Fri, Oct 08	Ch 6	Strings	
	Sun, Oct 10			Problem Set 4
	Mon, Oct 11	Ch 7	Lists	
7	Wed, Oct 13	Ch 7	Lists	
	Fri, Oct 15		*Mid-Semester Day*	
	Sun, Oct 17			Project 2
	Mon, Oct 18	Ch 7	Lists	
8	Wed, Oct 20	Ch 7	Lists	
	Fri, Oct 22		**Midterm**	
	Sun, Oct 24			Graphics Contest
	Mon, Oct 25	Ch 7	Lists	
9	Wed, Oct 27	Ch 9.1–9.2	Classes and Objects	
	Fri, Oct 29	Ch 9.1–9.2	Classes and Objects	
	Sun, Oct 31			Problem Set 5
	Mon, Nov 01	Ch 10.3–10.5	Inheritance	
10	Wed, Nov 03	Ch 10.3–10.5	Inheritance	

Week	Date	Chapter	Description	Due
	Fri, Nov 05		The Enigma Machine	
	Sun, Nov 07			Project 3
	Mon, Nov 08	Ch 11.1–11.3	Dictionaries and Hashing	
11	Wed, Nov 10	Ch 11.1–11.3	Dictionaries and Hashing	
	Fri, Nov 12	Ch 12	Designing Data Structures	
	Sun, Nov 14			Bonus Problem Set 6
	Mon, Nov 15	Ch 12	Designing Data Structures	
12	Wed, Nov 17	Ch 11.4	Sets	
	Fri, Nov 19	Ch 11.4	Sets	Project 4
	Mon, Nov 22		*Fall Break*	
13	Wed, Nov 24		*Fall Break*	
	Fri, Nov 26		*Fall Break*	
	Mon, Nov 29		The Adventure Project	
14	Wed, Dec 01	Ch 8	Algorithmic Analysis	
	Fri, Dec 03	Ch 8	Algorithmic Analysis	
	Mon, Dec 06	Ch 8	Algorithmic Analysis	
15	Wed, Dec 08		Fun Libraries	
	Fri, Dec 10		Cool Algorithms and Looking Ahead	Project 5
	Fri, Dec 17		**Final**	