

## General Information

### Instructor

Matt Bishop; *email:* [bishop@cs.ucdavis.edu](mailto:bishop@cs.ucdavis.edu); *web:* <http://seclab.cs.ucdavis.edu/~bishop>  
*Office:* 2209 Watershed Science; *phone:* (530) 752-8060  
*Office hours:* Mon 9:00–10:00AM, Wed 3:00–4:00PM, Fri 12:00–1:00PM

### TAs

Jishang Wei; *email:* [jswai@ucdavis.edu](mailto:jswai@ucdavis.edu)  
Justin Cummins; *email:* [jpcummins@ucdavis.edu](mailto:jpcummins@ucdavis.edu)

### Lectures

MWF 11:00AM–11:50AM in 194 Young

### Discussion Sections

You may go to any discussion section you like as long as there is room for those registered for that section.

Section 10-A01: Thu 8:00–8:50AM in 125 Wellman; TA Jishang Wei  
Section 10-A02: Thu 9:00–9:50AM in 290 Hickey; TA Jishang Wei  
Section 10-A03: Wed 4:10–5:00PM in 129 Wellman; TA Justin Cummins  
Section 10-A04: Thu 12:10–1:00PM in 103 Wellman; TA Justin Cummins

### Laboratories (and TA Office Hours)

You may go to any laboratory you like as long as there is room for those registered for that section. All labs are in 1131 Meyer.

Lab 10-A01: Mon 8:00–10:50AM; TA Jishang Wei  
Lab 10-A02: Wed 8:00–10:50AM; TA Jishang Wei  
Lab 10-A03: Tue 2:10–5:00PM; TA Justin Cummins  
Lab 10-A04: Fri 8:00–10:50AM; TA Justin Cummins

### Course Goals

The overall goal is to learn computers by studying programming and how to use them to solve problems. More specifically, we hope you will:

- Learn some basics about computers: a bit about their organization, software, and how they represent information
- Learn how to write small and moderately-sized programs, and how to use an integrated development environment, debuggers, compilers, and interpreters
- Learn the basics of the Python programming language, and through it the basic control and data structures, operations and data types in programming languages
- Learn how to design and write an algorithm

## Prerequisite

Two years of high school algebra

## Text

John Zelle, *Python Programming: An Introduction to Computer Science*; Franklin, Beedle & Associates, Inc., Wilsonville, OR 97070; © 2004. [ISBN 1-887902-99-6](http://www.fbeedle.com/).

## Class Web Site

The class web site is on SmartSite. To access it, go to <http://smartsite.ucdavis.edu> and log in using your campus login and password. Then go to ECS 10 in your schedule. Announcements, assignments, handouts, and grades will be posted there, and you must submit assignments there. The alternate web site, <http://nob.cs.ucdavis.edu/classes/ecs010-2009-02>, has everything except grades, and you cannot submit work there.

## Extra Credit

Extra credit is tallied separately from regular scores. If you end up on a borderline between two grades at the end of the course, extra credit will count in your favor. But, failure to do extra credit will never be counted against you, because grades are assigned on the basis of regular scores. You should do extra credit if you find it interesting and think that it might teach you something. Remember, though, it is not wise to skimp on the regular assignment in order to do extra credit!

You will also get extra credit if you go to discussion section. Be sure you sign in so we know to record the points.

## Grading

Written assignments	10%	Midterms	30%
Programming assignments	30%	Final	30%

## Academic Integrity

The UC Davis Code of Academic Conduct, available at <http://sja.ucdavis.edu/cac.html>, applies to this class. In particular, for this course, all work submitted for credit must be your own. You may discuss your assignments with classmates, with the instructor, or with the teaching assistant in the course to get ideas or a critique of your ideas, but the ideas and words you submit must be your own. Unless explicitly stated otherwise, collaboration is considered cheating and will be dealt with accordingly.

The single exception to this rule is debugging. Once you have written your program, if you need help debugging it, you are free to ask a classmate for help *providing that classmate has also written the program*. Sometimes having someone else look over a program that is not quite working right will lead you to the best way to fix it, and you both will gain valuable experience in looking at programs and figuring out what is going on. But you must not collaborate on writing the program.