Course Information

Overview

The CS 61 series is an introduction to computer science, with particular emphasis on software and on machines from a programmer's point of view.

1. CS 61A concentrates on the idea of abstraction, allowing the programmer to think in terms appropriate to the problem rather than in low-level operations dictated by the computer hardware.
2. CS 61B deals with the more advanced engineering aspects of software, such as constructing and analyzing large programs.
3. CS 61C focuses on machines and how they execute the programs you write.

In CS 61A, we are interested in teaching you about programming, not about how to use one particular programming language. We consider a series of techniques for controlling program complexity, such as functional programming, data abstraction, and object-oriented programming.

CS 61A primarily uses the Python 3 programming language. Python is a popular language in both industry and academia. It is also particularly well-suited to the task of exploring the topics taught in this course. It is an open-source language developed by a large volunteer community that prides itself on the diversity of its contributors. We will also examine the Scheme programming language, as well as the Structured Query Language (SQL) in the latter half of the course.

Mastery of a particular programming language is a very useful side effect of CS 61A. However, our goal is not to choose what language you use in your future studies and career. Instead, our hope is that once you have learned the essence of programming, you will find that picking up a new programming language is but a few days' work.

Prerequisites

Math 1A is a corequisite for CS 61A. (That is, it may be taken concurrently.) Math 10A or Math 16A are also fine.

There is no formal programming-related prerequisites for admission to CS 61A, but that doesn't mean that it's the right first course for all students. Many CS 61A students have had significant prior programming experience, including prior coursework. Some students take the course without any prior programming experience, but they typically must work substantially harder to master the material, perhaps simply because they have less practice working with programs. If you have no prior experience, and you find it challenging to complete all of the required

coursework in the first three weeks, you should seriously consider taking another course first. You'll likely have a better experience taking 61A later, and you won't fall behind in any meaningful way by taking one of the alternatives below prior to taking 61A.

Alternatives

If you want to build programming experience before taking CS 61A, we recommend that you take one of these courses first. You can always take CS 61A in a future semester (including the summer).

CS 10

CS 10: The Beauty and Joy of Computing (http://cs10.org) provides a bird's-eye view of the field of computer science. The course teaches students how to program using Snap (based on Scratch), one of the friendliest programming languages ever invented, as well as Python, the same language used in 61A. But the course is far more than just learning to program! You'll also learn about some "Big Ideas" of computing, such as abstraction, design, recursion, concurrency, simulations, and the limits of computation. You'll also see some beautiful applications of computing that have changed the world, as well as talk about the history of computing and where it will go in the future.

Data 8

Data 8: The Foundations of Data Science (http://data8.org/) is an introduction to data science designed to be accessible and useful for all Berkeley students, and built for students without prior programming experience. The course teaches students to program in Python 3, but covers a much smaller subset of the language than CS 61A. Most of the course focuses on data processing and statistical techniques that are central to using computers to answer questions about the world. The overlap between Data 8 and CS 61A is small (perhaps 25%), but the programming skill you will acquire in Data 8 will prepare you for the faster pace of CS 61A.

Course Format

The course includes many events and opportunities for learning: lecture, lab, discussion, office hours, group mentoring, and one-on-one tutoring. Weekly lab and discussion sections are typically the most valuable events to attend. For the rest, your attendance is optional. These events exist to help you learn, and so you are advised to attend as many as required to master the material.

Lecture

There are three 50-minute lectures per week. Videos and slides will be posted the night before each lecture. Most students prefer the videos to live lecture, but you should try both and decide for yourself. **If you choose to watch the videos, watch them the day of live lecture. Please do not fall behind; many students who fall behind have a rough time trying to catch up.**

Section

There are both lab and discussion sections each week. They are paired by number: if you are enrolled in lab 120C, then you should attend discussion 120B. Lab exercises and discussion quizzes are both graded, so plan on attending every week. These sections are run by an amazing group of Teaching Assistants who have been carefully selected for their ability, enthusiasm, and dedication to learning. Getting to know your TA is an excellent way to succeed in this course.

It will be difficult to switch sections this semester, due to the large enrollment of the course. You may attempt to attend a different section than the one in which you are enrolled, but seating priority will be given to people who are officially enrolled in that section. If you do end up changing sections, you do not need to officially update your section on CalCentral. It's best to attend one section regularly, rather than attempting to move around week to week.

Office Hours

Attending office hours is another excellent way to succeed in this course. You can ask questions about the material, receive guidance on assignments, work with peers and course staff in a small group setting, find project partners, and learn about computer science at Berkeley.

Assignments

Each week, there will be problems assigned for you to work on, most of which will involve writing, debugging, and discussing programs. These assignments come in three categories: lab exercises, homework assignments, and projects.

Labs

Lab exercises are designed to introduce a new topic. You can complete and submit these during the scheduled lab sections, or on your own time before the scheduled due date. Most students find that attending lab is **much** more useful than working on lab assignments independently.

Lab exercises are graded on completion and understanding. To receive credit, you must make good progress and submit your work. It is not required that you complete all problems to receive credit.

Check-offs will be completed during lab. Academic interns or other staff will check your understanding of lab assignments by asking you a few short questions about the current lab or the previous lab (your choice). If you answer their questions in a way that demonstrates good progress, they will give you *check-off* credit. Check-offs are a great opportunity to receive one-on-one feedback and gain a deeper understanding of the material.

Quizzes

Discussion quizzes are designed to detect whether students are keeping up with the course material. Although the quizzes will be graded for correctness, submitting any work during discussion is sufficient to earn full credit for the purpose of points. If you cannot make your

discussion, email the TA of another section and see if you can take the quiz in their section. If you can't make any section, email your TA with your conflict and they will send you a make up quiz.

Homework

Homework assignments are meant to illustrate and explore new topics. You are encouraged to discuss the homework with other students, as long as you write your own code and submit your own work. Finding a study group is a great idea. The purpose of homework is for you to learn the course material, not to prove that you already know it. Therefore, homework is not graded on the correctness of your solutions, but on effort. You will get full credit for reasonable effort so long as you make good progress on **all** the problems. Note that there is no partial credit for homework assignments. **If you are stuck on a problem, simply describe your progress rather than copying the answer from someone else or the Internet; you'll still get credit and won't be flagged for cheating.**

Projects

Projects are larger assignments intended to combine ideas from the course in interesting ways. The first two projects will be done individually; the last two in groups of two students. When working in pairs, you should work together to ensure that both group members understand the complete results. We recommend finding a project partner in your section. Your TA will help. You may also work alone on all projects, although partners are recommended for the paired projects. Projects are graded on both correctness and composition (composition.html).

Exams

The first midterm exam will be held 8-10 PM Thursday, September 14. You are permitted to bring one double-sided, letter-sized, handwritten cheat sheet.

The second midterm exam will be held 8-10 PM Thursday, October 19. You are permitted to bring two double-sided, letter-sized, handwritten cheat sheets.

The final exam will be held 3-6 PM Wednesday, December 13. You are permitted to bring three double-sided, letter-sized, handwritten cheat sheets.

Resources

Textbook

The online textbook for the course is Composing Programs (http://composingprograms.com/), which was created specifically for this course. Readings for each lecture appear in the course schedule. We recommend that you complete the readings before attending lecture.

We may occasionally differ from the material found in Composing Programs. As a result, we recommend the lecture notes, labs, and discussion handouts as your primary source of information.

Computing Resources

If you are enrolled in the class, you may request a CS 61A instructional account. This will allow you to use any EECS instructional lab computer in Soda or Cory Hall. You may use any lab you wish, as long as there is no class using the space.

Be respectful of the lab space. Please don't steal the chairs, and definitely do not eat or drink in the lab. Don't unplug anything; unplugged computers make our hard-working instructional computing team very sad. If you see someone disrupting the space, ask them to stop.

Labs are normally available for use at all times, but you need a card key for evening access. If you are a Berkeley student, your student ID will automatically grant you access to the Soda second floor labs. Otherwise, you can fill out an application from 387 Soda (the front desk).

DSP Accommodations

If you are enrolled in DSP and would like an accommodation, please email cs61a@berkeley.edu (mailto:cs61a@berkeley.edu). We will post detailed information on Piazza as the semester progresses for critical matters such as scheduling an alternate exam time.

Grading

The grading policy of the course has these goals: it should encourage you to do the coursework and reward reasonable effort with reasonable grades; it should minimize competitiveness and grade pressure, so that you can focus instead on the intellectual content of the course; and it should minimize the time we spend arguing with students about their grades. To meet these goals, your course grade is computed using a point system with a total of 300 points.

- Midterm 1, worth 40 points.
- Midterm 2, worth 50 points.
- The final exam, worth 75 points.
- Four projects, worth 100 points in total.
- Homework, worth 25 points in total.
- Lab exercises, check-offs, and discussion quizzes, worth 10 points in total. See the Lab and Discussion grading policy below for more information.

Each letter grade for the course corresponds to a range of scores:

```
A+  ≥ 296    A  ≥ 280    A-  ≥ 265
B+  ≥ 245    B  ≥ 225    B-  ≥ 205
C+  ≥ 195    C  ≥ 185    C-  ≥ 175
D+  ≥ 170    D  ≥ 165    D-  ≥ 160
```

Notice that this scale is nonlinear; the steps are wider in the B range. Your final score will be rounded to the nearest integer before being converted to a letter grade.

There is no curve; your grade will depend only on how well you do, and not on how well everyone else does. It is based on how students performed in previous semesters. It is possible that the instructors will adjust the thresholds in your favor, for example if exam scores are abnormally low, but that scenario is unlikely. More likely, these are the exact thresholds that will be used at the end of the course to assign grades (contrary to popular rumor).

Incomplete grades will be granted only for dire medical or personal emergencies that cause you to miss the final, and only if your work up to that point has been satisfactory. You must complete all coursework before the drop deadline to be considered for an incomplete grade.

Lab and Discussion

Lab exercises, check-offs, and discussion quizzes, worth 10 points in total.

Each lab assignment, check-off, and discussion quiz is worth 1 point. As long as you demonstrate good progress on the lab assignment and check-off, you will receive 1 point for that lab assignment or lab check-off. Any discussion quiz submission will receive 1 point of credit.

Any combination of lab exercises, check-offs, and discussion quizzes can be completed for credit in this category up to the maximum of 10 points. All lab assignments, check-offs, and discussion quizzes completed in excess of the 10 points become exam recovery points.

Exam Recovery Policy

We understand that exams may not be entirely indicative of your efforts in the class. For this reason, you may earn back a certain number of points for each exam based on your participation in lab and discussion sections. Participation is not required, but it is tracked and potentially beneficial.

There are 15 lab exercises, 9 lab check-off opportunities, and 11 discussion quizzes for a total of 35 possible participation points. Each point from lab exercises, check-offs, and discussion quizzes in excess of the 10 required participation points qualifies as 1 **recovery point**. Recovery points help recover points lost on *all three exams*.

The maximum number of recovery points is capped at 20 so you don't need to worry about completing every single participation opportunity.

We calculate your exam recovery using the following logic:

```
def exam_recovery(your_exam_score, participation_points, max_exam_score, recovery_cap=20):
    if participation_points < 10:
        return 0
    else:
        half_score = max_exam_score / 2
        max_recovery = max(0, (half_score - your_exam_score) / 2)
        recovery_points = participation_points - 10
        recovery_factor = min(recovery_points, recovery_cap) / recovery_cap
        return max_recovery * recovery_factor
```

This means if you receive more than half the points on every exam, your score remains the same. If you score below half the points, you will recover a few points. If you score far below half the points, you will recover many points. The more recovery points you earn, the more exam points will be recovered.

Exam

Midterm 1

Your Exam Score

0

Participation Points

0

Calculate

Points Recovered

Adjusted Exam Score

Late Policy

If you cannot turn in an assignment on time, contact your TA and partner as early as possible. Depending on the circumstance, we may grant extensions.

- **Labs**: We will not accept any late lab submissions. Lab exercises are graded on completion.
- **Homeworks**: We will not accept any late homework submissions. Remember that homework is not graded on the correctness of your solutions, but on effort. You will get full credit for reasonable effort, as long as you make good progress on all the problems.
- **Projects**: Submissions within 24 hours after the deadline will receive 75% of the earned score. Submissions that are 24 hours or more after the deadline will receive 0 points.

Learning Cooperatively

With the obvious exception of exams, we encourage you to discuss course activities with your friends and classmates as you are working on them. You will definitely learn more in this class if you work with others than if you do not. Ask questions, answer questions, and share ideas liberally.

Working cooperatively in groups is a change from the traditional approach in schools, in which students work either in isolation or in competition. But cooperative learning has become increasingly popular as educational research has demonstrated its effectiveness. One advantage of cooperative learning is that it allows us to give intense assignments, from which you'll learn a great deal, while limiting the workload for each individual student. Another advantage, of course, is that it helps you to understand new ideas when you discuss them with other people. In the past many students have commented that they didn't really understand the course until they worked as academic interns and explained the ideas to later students.

Since you're working collaboratively, keep your project partner and TA informed. If some medical or personal emergency takes you away from the course for an extended period, or if you decide to drop the course for any reason, please don't just disappear silently! You should inform your project partner, so that nobody is depending on you to do something you can't finish.

Online Forum

If you have any questions, please post them to Piazza (http://www.piazza.com/class), the course discussion forum. Piazza allows you to learn from questions your fellow students have asked. We encourage you to answer each others' questions!

Piazza is the best and most reliable way to contact the course staff. You are also welcome to email the instructors or your TA directly.

Academic Honesty

Cooperation has a limit, and in CS 61A that limit is sharing code. You are free to discuss the problems with others beforehand, but you must write your own solutions. The only student with whom you can share code is your project partner.

Since this may be your first computer science class, exactly what constitutes as cheating might be unclear. If you are unsure if what you are doing is cheating, please clarify with the instructors or TAs. The following is a list of things you should NOT do. This list is not exhaustive, but covers most of the big offenses:

- Do not copy code from any student who is not your partner.
- Do not allow any student other than your partner to copy code from you.
- Do not copy solutions from online or post your solutions publicly. This includes websites such as Stack Overflow, Pastebin, and public repositories on GitHub. (You are welcome to use private repositories.)

If you find a solution online, please submit a link to that solution (http://goo.gl/GEHk49). When we find an online solution, we ask the author to remove it. We also record the solution and use it to check for copying. By reporting online solutions, you help keep the course fair for everyone.

In summary, we expect you to hand in your own work, take your own tests, and complete your own projects. The assignments and evaluations are structured to help you learn, which is why you are here. The course staff works hard to put together this course, and we ask in return that you respect the integrity of the course by not misrepresenting your work.

If you are found to be cheating in this course, we will always ask you to come in and discuss the situation and give you a chance to explain. The first offense results in a negative score for the assignment. If you are found to be cheating a second time, we will not hesitate to fail you, report you to the center for student conduct, and then also call your parents. Detecting copied assignments is very easy for our computers, so please don't try!

Rather than copying someone else's work, ask for help. You are not alone in this course! The entire staff is here to help you succeed. If you invest the time to learn the material and complete the projects, you won't need to copy any answers.

A Parting Thought

Grades and penalties aren't the purpose of this course. We really just want you to learn. The entire staff is very excited to be teaching CS 61A this semester and we're looking forward to meeting such a large and enthusiastic group of students. We want all of you to be successful here. Welcome to CS 61A!

CS 61A (/~cs61a/fa17/)

Weekly Schedule (/~cs61a/fa17/weekly.html)

Office Hours (/~cs61a/fa17/office-hours.html)

Staff (/~cs61a/fa17/staff.html)

Resources (/~cs61a/fa17/resources.html)

Studying Guide (/~cs61a/fa17/articles/studying.html)

Debugging Guide (/~cs61a/fa17/articles/debugging.html)

Composition Guide (/~cs61a/fa17/articles/composition.html)

Policies (/~cs61a/fa17/articles/about.html)

Assignments (/~cs61a/fa17/articles/about.html#assignments)

Exams (/~cs61a/fa17/articles/about.html#exams)

Grading (/~cs61a/fa17/articles/about.html#grading)