Syllabus & Course Policies

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 programs.

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 use two other languages in the latter half of the course: the Scheme programming language and the Structured Query Language (SQL).

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

A complete list of lecture topics, readings, and assignments appears in the daily schedule (/).

Prerequisites

Math 1A is listed as a corequisite for CS 61A. (That is, it may be taken concurrently.) Math 10A or Math 16A are also fine. It is possible to take CS 61A without knowing or learning calculus; all of the old calculus-based examples have been removed over the years. However, taking calculus is a great way to brush up on the arithmetic and algebra that appear regularly in CS 61A.

There are no formal programming-related prerequisites for CS 61A, but it's not 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 limited 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.

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 see 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 and CS 88

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. This course was built for students without prior programming experience. It 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 help you maintain the faster pace of CS 61A.

CS 88: Computational Structures in Data Science (https://cs88-website.github.io/) is an introduction to programming and computing that has more than 50% concept overlap with CS 61A. It is designed for students interested in data science who want to expand their knowledge of programming and program structures beyond what is covered in Data 8. Students who complete CS 88 can either proceed directly to CS 61B or subsequently take CS 61A, a path that offers a substantial amount of review because of the high topic overlap between the courses.

Course Format

The course includes many events and opportunities for learning: lecture, lab, discussion, office hours, guerrilla sections, and group mentoring. We understand that everyone learns differently, so not all of these events are required. However, it is recommended that you try everything out to figure out what combination of these events works best for you.

Lecture

There are three 50-minute lectures per week. Slides and videos will be posted before each lecture. A screencast of the live lecture will be posted soon after each lecture occurs. This course moves fast, and lecture is tightly coordinated with section. Please attend or watch each lecture the day it is given and before you attend section.

Section

There are two sections each week: one lab and one discussion. 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. Participation in lab and discussion determines your participation score for the course.

Sign up for a section using this form (http://links.cs61a.org/section-signup).

Office Hours

In office hours, you can ask questions about the material, receive guidance on assignments, and work with peers and course staff in a small group setting. See the office hour schedule (/office-hours.html) and come by; no appointments are needed.

Assignments

Each week, there will be problems assigned for you to work on, most of which will involve writing and analyzing 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. Many students find that attending lab is more useful than working on lab assignments independently.

Lab exercises are scored on correct completion. To receive credit, you must complete all of the problems that are not marked as optional and pass all tests.

Homework

Homeworks are weekly assignments meant to help you apply the concepts learned in lecture and section on more challenging problems. They will usually be released on Friday night and be due the following Thursday night.

Collaboration

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, you can expect to receive substantial assistance from the course staff. You're welcome to help others once you solve a problem.

If you are stuck on a problem, come get help instead of copying the answer from someone else or the Internet; you'll still get credit and won't be flagged for cheating.

Partial Credit

There is partial credit, with every incorrect answer losing you one point on the homework (up till 0). Usually, homeworks are out of 2. The lowest homework will be dropped from the calculation of your grade in the course at the end of the semester

Homework Recovery Policy

You can recover one incorrect question per homework by going through the homework recovery process:

1. Fill out a form declaring which question you want to recover by Friday 11:59PM, the day after the homework is due
2. Go to lab and talk to your TA about that question in a group with other similarly situated students. Your TA will record your recovery point and your grade will be updated

Projects

Projects are larger assignments intended to combine ideas from the course in interesting ways. Some projects can be completed in pairs. When working in pairs, you should work together to ensure that both of you understand the complete results. We recommend finding a project partner in your section. Your TA will help if you ask. 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

Midterm 1 will likely be held 8pm-10pm Monday, February 10. **This date is not yet confirmed and is subject to change.** You are permitted to bring one double-sided, letter-sized, handwritten sheets of notes.

Midterm 2 will be held 7pm-9pm Thursday, March 19. You are permitted to bring two double-sided, letter-sized, handwritten sheets of notes. One of them can be your notes from Midterm 1.

> Students who have a course conflict with a midterm should notify the staff *before* the exam in question, and special arrangements will be made. Details of how to report an exam conflict will be announced shortly.

The final exam will be held 11:30pm-2:30pm Tuesday, May 12. You are permitted to bring three double-sided, letter-sized, handwritten sheets of notes. Two of them can be your notes from Midterm 2.

> If you have a direct conflict with another final exam, we will allow you to take an alternate final 3pm-6pm on Tuesday, May 12. We will not provide final alternates for any other reason.

Resources

Textbook

The online textbook for the course is Composing Programs (http://composingprograms.com/), which was created specifically for this course, based on the classic textbook Structure and Interpretation of Computer Programs (https://mitpress.mit.edu/sites/default/files/sicp/index.html). Readings for each lecture appear in the course schedule. We recommend that you complete the readings before attending lecture.

Computing Resources

If you are enrolled in the class, you may request a CS 61A instructional account. This account 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.

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).

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.

Grading

Your course grade is computed using a point system with a total of 300 points.

- Midterm 1, worth 40 points.
- Midterm 2, worth 25 points. (This was adjusted due to virus concerns)
- The final exam, worth 100 points. (This was adjusted due to virus concerns)
- Four projects, worth 100 points.
- Homework, worth 20 points.
- Lab, worth 10 points.
- Discussion, worth 5 points.

There are a handful extra credit points available throughout the semester, perhaps around 10, that are available to everyone.

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

```
A+  ≥ 300    A  ≥ 285    A-  ≥ 270
B+  ≥ 250    B  ≥ 225    B-  ≥ 205
C+  ≥ 195    C  ≥ 185    C-  ≥ 175
D+  ≥ 170    D  ≥ 165    D-  ≥ 160
```

Your final score will be rounded to the nearest integer before being converted to a letter grade. 0.5 rounds up to 1, but 0.49 rounds down to 0.

There is no curve; your grade will depend only on how well you do, and not on how well everyone else does. Score thresholds are 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 we try to avoid that scenario. If all goes according to plan, then these are the exact thresholds that will be used at the end of the course to assign grades (contrary to popular rumor). In a typical semester, about 60% of students taking the course for a letter grade will receive a B+ or higher.

Incomplete grades will be granted only for medical or personal emergencies that cause you to miss the final or last part of the course, only for students who have completed the majority of the coursework, and only if work up to the point of the emergency has been satisfactory.

Your two lowest homework scores will be dropped (policy updated 3/14).

Each lab that you complete is worth 1 point, and you can receive a maximum of 10 lab points. There are going to be around 13 lab assignments, so you can skip a few.

Each discussion you attend is worth 1 point, excluding discussion 0 (during week 1). You can receive a maximum of 5 discussion points. There are going to be around 12 discussions, so you can skip many, but most students continue to attend discussion throughout the semester because they find it useful.

Discussion Participation

Attending more than 5 discussions will contribute to recovery points on midterms. Attending at least 10 discussions will give you the maximum amount of midterm recovery.

We calculate your midterm recovery using the following logic, where `attendance` is the number of weeks that you attend discussion.

```python
def exam_recovery(your_exam_score, attendance, max_exam_score, cap=10):
    half_score = max_exam_score / 2
    max_recovery = max(0, (half_score - your_exam_score) / 2)
    recovery_ratio = min(attendance, cap) / cap
    return max_recovery * recovery_ratio
```

According to this formula, if you receive more than half the available points on each midterm, then you don't recover any points. If you score just 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 weeks you attend discussion, the more exam points will be recovered.

The purpose of this policy is to ensure that students who continue to invest time in the course througout the semester are able to pass.

There are no recovery points available on the final exam.

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 rarely accept late lab submissions. There is no partial credit.
- **Homework**: We rarely accept late homework submissions.
- **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.

Citizenship

For exceptionally rude or disrespectful behavior toward the course staff or other students, your final grade will be lowered by up to a full letter grade (e.g., from an A- to a B-) at the discretion of the course instructor. You don't need to be concerned about this policy if you treat other human beings with even a bare minimum of respect and consideration and do not engage in behavior that is actively harmful to others.

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.

You are also welcome to code collaboratively with others in your lab and solve lab problems in small groups.

Learning cooperatively is different from sharing answers. You shouldn't be showing your code to other students or looking at others' code, except:

- During lab, you can share all you want as long as you're all learning.
- For a project that allows partners, you can share anything with your partner.
- If you've finished a problem already, you can look at others' code to help them finish.

If you are helping another student, don't just tell them the answer; they will learn very little and run into trouble on exams. Instead, try to guide them toward discovering the solution on their own. Problem solving practice is the key to progress in computer science.

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/berkeley/spring2020/cs61a), 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 cs61a+sp20@berkeley.edu, denero@berkeley.edu, or your TA directly.

Academic Honesty

Cooperation has a limit, and in CS 61A that limit is reading others' homework or project solution to a problem before you solve that problem on your own. You are free to discuss the problems with others beforehand, but you must write your own solutions. You may share code with your project partner.

If you are unsure if what you are doing is cheating, please clarify with the instructor 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 sources such as Stack Overflow, Pastebin, and public repositories on GitHub.
- Do not post your solutions publicly during or after the semester.

If you find a solution online, please submit a link to that solution anonymously (https://goo.gl/forms/nL2yOj1Z81HcQYDi2). 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.

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!