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.

61A primarily uses the Python 3 (http://en.wikipedia.org/wiki/Python_(programming_language)) 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 (http://en.wikipedia.org/wiki/Scheme_(programming_language)) programming language in the latter half of the course.

Mastery of a particular programming language is a very useful side effect of 61A. However, 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

Mathematics 1A is a corequisite for 61A (i.e. may be taken concurrently).

Prior programming experience is *not* a prerequisite for 61A. However, students without prior experience historically spend a large amount of time each week on the course.

Alternatives

If you don't feel ready for 61A, we recommend CS 10: The Beauty and Joy of Computing (http://cs10.org/), which provides a bird's-eye-view of the field of computer science. The course teaches students how to program using Snap (one of the friendliest programming languages ever invented) and Python. You'll also learn about some Big Ideas of computer science, as well as the societal transformations and challenges that come along with the technical knowledge.

Course Format

All course content (e.g. lectures, handouts, and assignments) will be posted on the course website at cs61a.org (/~cs61a/su16/).

Lectures: There are four 90-minute lectures per week. Slides will be posted the night before each lecture. In-lecture quizzes will be administered at the beginning of Thursday lectures.

Lab and Discussion Sections: There are two lab and two discussion sections each 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.

> If you wish to switch sections, simply start attending the section you wish to join. We are often able to accommodate these wishes, although seating priority will be given to people who are officially enrolled in that section. You do not need to officially update your section. It's best to find a permanent section in the first 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. All members of the instructional staff hold office hours each week.

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. Lab exercises are graded on completion.

There will be two lab assignments each week, each worth two points. We will drop your two lowest lab scores over the course of the semester, for a total of 20 points.

Homeworks

Homeworks 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. 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. If you are unable to solve a problem, submit evidence of your effort (e.g. partially-working solutions).

There will be ten homework assignments. Each is worth three points, for a total of 30 points.

In addition to programming homework assignments, we will also have three surveys throughout the course. Each of these will be worth one extra credit point. Therefore, missing one homework can be made up by filling out all three surveys.

Projects

Projects are larger assignments intended to combine ideas from the course in interesting ways. You are encouraged to complete projects in pairs; your partner can be anyone else enrolled in 61A. We recommend finding a partner in your section. Your TA will help. You may also work alone, although this is not recommended.

There will be four projects, for a total of 100 points. Projects are graded on correctness, as well as for code composition (i.e. code clarity and legibility).

Checkoffs

During every lab section, lab assistants will run checkoffs for homework and lab assignments. These will consist of a few short questions meant to assess your understanding of the material. If you made mistakes on the assignment but later reviewed the solutions, you should be fine.

Each week, you *must* choose to check off at least one of the week's homework or lab assignments during lab. Checkoffs are worth one point of the week's total assignment score. Therefore, if you fail to get checked off for the week, one point will be deducted from your overall grade. To reduce administrative overhead, please try to attend the same lab section each week for your checkoff. Please note that checkoffs on lab assignments will not ask questions on the optional questions.

Checkoffs are a great opportunity to get one-on-one feedback and gain a deeper understanding of the material, so you are welcome to check off multiple assignments each week. However, you are only required to check off one assignment per week, and checking off more than one will not earn you any additional points.

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. You are allowed to drop two lab assignments.
- **Homeworks**: We will not accept any late submissions of homework.
- **Projects**: Submissions that are 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.

Quizzes

Written quizzes will be administered every week in the first 25 minutes of Thursday lectures (i.e. from 11:10 - 11:35 AM). Each quiz will contain 1-2 exam-level questions. They are closed-book.

Take-home coding quizzes will be assigned as well. You will have 24 hours to complete them. They are open-book, but you may not consult other students or the Internet.

There will be six written quizzes and three coding quizzes. Each quiz will be worth five points, and we will drop your lowest quiz score. These quizzes are equivalent to one exam, but having several quizzes will make it harder for one bad day to significantly impact your grade.

> If you have a conflict with any written quiz, you must let us know by the first week of the course to be assigned an alternate quiz time. Additional exceptions will only be made for extraordinary circumstances.

Exams

The midterm exam will be held on Thursday, July 14 from 5 - 8 PM in 2050 VLSB. You are permitted to bring one double-sided letter-sized cheat sheet.

The final exam will be held on Friday, August 12 from 5 - 8 PM in 155 Dwinelle. You are permitted to bring two double-sided letter-sized cheat sheets.

> If you have a conflict with either exam, you must let us know by the first week of the course. We may be able to offer an alternate exam time. Additional exceptions will only be made for extraordinary circumstances.

More exam details will be released as the exam approaches.

Discussion Participation

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 discussion sections. Therefore, participation is not required, but it is tracked and potentially beneficial.

Discussions 1 through 6 (up to 5) will count toward midterm recovery points. Discussions 8 through 13 (up to 5) will count toward final recovery points.

We compute your recovery points with the following formula:

```
def recovery_points(score, discussions):
    participation = min(5, discussions)
    return max(score, score / 2 + participation * 2) - score
```

This means if you score over 20 points on an exam, your score remains the same. If you score below 20 points on an exam, you have the chance to recover a few points. The more you participate in discussion, the more points you can earn back.

Grading

Your course grade is computed using a point system with a total of 300 points. Half of those points will come from assignments; the other half will come from quizzes and exams.

- Labs, worth a total of 20 points.
- Homework, worth a total of 30 points.
- Projects, worth a total of 100 points.
- Quizzes, worth a total of 40 points.
- Midterm, worth 40 points.
- Final, worth 70 points.

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

```
A+  ≥ 294    A  ≥ 283    A-  ≥ 275
B+  ≥ 260    B  ≥ 240    B-  ≥ 230
C+  ≥ 220    C  ≥ 210    C-  ≥ 200
D+  ≥ 190    D  ≥ 180    D-  ≥ 170
```

Notice that this scale is nonlinear; the steps are wider in the B range.

> This grading formula implies that *there is no curve*; your grade will depend only on how well you do, and not on how well everyone else does.

You can check your grade in the class at any time by logging into your CS 61A lab account.

Learning Cooperatively

With the obvious exception of exams and quizzes, 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.

> 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.

Cheating Policy

Cooperation has a limit, and in 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 choose to cheat, you will receive a score of -100% of the assignment score. For example, if an assignment is worth 10 points, you will receive a -10 out of 10. 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.

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 will be automatically assigned a CS 61A lab 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

Zhen Qin is our staff member in charge of DSP accommodations. If you are enrolled in DSP and would like an accommodation, please email her at zhenqin@berkeley.edu. We will post detailed information on Piazza as the semester progresses for critical matters such as scheduling an alternate exam time.

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 61A this summer, and we're looking forward to having such a large and enthusiastic group of students this semester. We want all of you to be successful here. Welcome to 61A!