General Background Information

These course policies are pretty long. With the exception of the collaboration policy we're not expecting you to read the entire thing in one sitting. However, you'll hopefully find the answers to any logistics questions you may have somewhere in this document.

Welcome to CS 61B

The CS 61 series is an introduction to Computer Science, with particular emphasis on software and machines from a programmer's point of view. CS 61A covered high-level approaches to problem-solving, providing you with a variety of ways to organize solutions to programming problems as compositions of functions, collections of objects, or sets of rules. In CS 61B, we move to a somewhat more detailed (and to some extent, more basic) level of programming.

In CS 61A, the correctness of a program was our primary goal. In CS 61B, we're also concerned with engineering. An engineer, it is said, is someone who can do for a dime what any fool can do for a dollar. Much of CS 61B will be concerned with the tradeoffs in time and memory for a variety of methods for structuring data. We'll also be concerned with the engineering knowledge and skills needed to build and maintain moderately large programs.

Background Knowledge

This class assumes you have taken CS 61A, CS 88, or E 7, or have equivalent background to a student who has taken one of these courses. The course is largely built upon the assumption that you have taken CS 61A. CS 88 and E7 students may find the beginning of the course to be a bit scarier, particularly when it comes to object oriented programming.

We assume you are coming in with zero Java experience. Nonetheless, we will move through basic Java syntax very quickly. Though the syntaxes of Java, Python, MATLAB, Scheme, etc. are enormously different, the underlying computational models are surprisingly similar.

If you already have Java experience, great! We hope that you'll help out your fellow students in discussion, lab, and on our class forum, particularly in the opening weeks when everyone is catching up on Java.

Is this the right course for me?

This is a course about data structures and programming methods. It happens to also teach Java, since it is hard to teach programming without a language. However, it is not intended as an exhaustive course on Java, creating Android apps, user interfaces, graphics, or any of that fun stuff.

Some of you may have already taken a data structures course, and simply want to learn Java or C++. For you, self-study may be a better option. CS 9F (C++ for programmers) and CS 9G (Java for programmers) are both one-unit self-paced courses that will teach you more of what

you want to know in less time. There is no enrollment limit for that course, and you work through it at your own pace after the first and only lecture.

Finally, the 1-unit self-paced course CS 47B is for students with "sufficient partial credit in 61B," allowing them (with instructor's permission) to complete the CS 61B course requirement without taking the full course. The 47B guide is at this link.

Lectures, Discussion and Lab Sections

Rather than presenting lecture material during our officially scheduled lecture time of MW, 1-2 PM, Josh Hug will lead a live Q&A session with no specific pre-planned activities. On Friday, the live Q&A will be at 2 PM.

Lecture material will instead be presented in pre-recorded videos, currently available on the course website. However, Lecture 1 and some other lectures will be presented live during the regularly scheduled period. We will announce on our course forum when live lectures will take place.

Each week there is a 1-hour Discussion section and a 2-hour Lab section headed by a TA.

Discussion sections feature a worksheet with problems reviewing the material learned in the past week. In discussion sections, TAs will help students review the material and solve the problems on the worksheet. Discussion sections will be delivered on Zoom.

Lab sections feature coding assignments that students submit for credit. In these sections, TAs will introduce the assignment, review relevant material, and answer students' questions. In some weeks, Lab TAs will also go over the staff's solution to the lab assignment. Lab sections are also supported by academic interns. Labs will take place on Discord servers. To learn how to use our Discord servers, please refer to our Discord Guide.

Information about the staff running each section can be found on the staff page. Section signups, the Zoom meeting links, and Discord information for each TA will be available on our course forum, Ed, shortly before classes start.

The schedule of all sections and lectures can be found at the bottom of our course website's main page.

Discussion and lab attendance are not mandatory. However, there are 2 (EDITED 2/16) labs for which attendance is mandatory for full credit: Lab 5 (Project 1 code review lab), and Lab 15 (Project 3 demo lab). These labs include an assignment for which you will need to get checked-off by a TA to receive credit. These labs cannot be submitted late (see lateness policy below).

Sections are meant to provide an opportunity for students and TAs to engage in meaningful interactions, so students who attend discussion and/or lab sections will be required to have their video turned on for the entirety of the section. If your video is turned off for more than a few minutes, you will automatically be removed from the section.

For students who prefer to not attend section due to this requirement, the discussion worksheet, solutions, and recorded videos of TAs walking through the solutions will be available on the course website every week after all discussion sections have been taught. Lab assignments will also be available on the course website, and students who wish to not attend a lab section can get help in Office Hours and on our course forum, Ed.

If you choose to not sign up for a lab to attend regularly, for the mandatory attendance labs, you will have the following options:

- Lab 5: You will be able to attend any lab that works with your schedule. Mandatory video-on will not be enforced for this lab only, but will be encouraged.
- Lab 15: There will be official sign ups for this checkoff, in which you and your partner will present the work you completed for Project 3. The signups will be available shortly before the last week of classes. Mandatory video-on will not be enforced for this checkoff, but will be encouraged.

Mentor GSIs

All students will be assigned a mentor TA. Your mentor TA will keep an eye out for you, and will also be your point-of-contact for issues when you need help with course policies, strategies for studying, and any other advice. Please refrain from asking your mentor TA for help with course assignment, as it is against our staff policy to offer help outside of sections, Office Hours, or our course forum, Ed.

During the second week of classes you will be able to choose your mentor TA, who does not necessarily have to be your discussion or lab TA. If you don't choose one, you will be randomly assigned a mentor TA. At various points throughout the class, your mentor TA will reach out to you.

Online Resources

The course home page will provide one-stop shopping for course information. The course schedule as well as all handouts, homework, labs, FAQs, etc., will be posted there.

Our discussion forum this semester will be Ed Discussions. For most questions about the course, Ed is the right place to ask them. The course staff reads it regularly, so you will get a quick answer. Furthermore, by posting online as opposed to emailing us directly, other students benefit by seeing the question and the answer. Don't forget to check Ed before asking your question, just in case someone else has already posted it. If you have a question about something pertaining to your own code that shouldn't be shared with the class, or if you have a

question about a personal matter, you can make a private post on Ed, which will only be seen by Josh and the TAs.

Please read our Ed Guide and Policies. We will only respond to questions that adhere to our policies of using Ed.

The e-mail address cs61b (at) berkeley.edu will send a message to the course staff (Josh and the head TAs). You can use it for correspondence that you don't want to seen on our class forum. The head TAs and Josh all read it, so you will usually get a reply within a few days. If you send a question that is of general interest, we may post the response on Ed (we will keep personal information out of it, of course). If you have any problems that require an exception to course policy (e.g. medical emergencies or sudden necessary travel that result in extended absences), please contact cs61b (at) berkeley.edu. **Please do not email Josh for exceptions. Email cs61b (at) berkeley.edu.**

In the beginning of the semester, we will issue each student a Github repository on which you will save your work for the different CS 61B assignments. You will use this repository to submit the assignments as well. We will also use Github to share skeleton files with you. Finally, for every partnership you establish you will get a dedicated "partners' repository" from which you will submit assignments on which you collaborated on with another student (more details below). Information on how to get started with your Github repository will be available in Lab 1.

After you submit an assignment through Github, you will be able to see the results for a submission you made on Gradescope, where we run our autograders. Information on how to use Gradescope in CS 61B will be available in Lab 1.

In general whenever we release an assignment on the course website, we also release its grader on Gradescope. You can only run the grader 3 times an hour. This is to dissuade you from using the grader to debug your code, only making small changes between submissions. Some projects will have different release schedules and restrictions for their graders. For example, the grader of a project may only be released shortly before the deadline, and/or allow only a few submissions per day. Whenever we release an assignment with irregular grader details, we will specify them on Ed.

Rather than using bcourses, we will be using our own custom learning management system called Beacon located at beacon.datastructur.es. You can use Beacon to keep track of your grades and late assignments, and you will also use it to specify your lab partner and mentor GSI. You can also read our full guide on Beacon here.

Office Hours

In Office Hours, you can get help from our staff and academic interns with the different assignments, exam preparation, logistical matters, and any advice you may need. You will be able to find the link to the Office Hours Discord Server on our course forum, Ed. To learn how to use our Discord servers and how Office Hours works please read our Discord Guide. We will

use the online Office Hours queue to keep track of students in Office Hours. Staff will always skip tickets on the queue that do not adhere to our Office Hours policies.

Course Materials

There is no required textbook for the class.

There is an online textbook written by myself and a large team of former TAs. It can be found at https://joshhug.gitbooks.io/hug61b.

If you find these notes insufficient, you might consider consulting Paul Hilfinger's (free) Java Reference or Head First Java, 2nd Edition by Sierra and Bates (O'Reilly, 2005). These are not required for the course.

The optional textbook for the weeks 5-14 of the course is Algorithms, 4th Edition by Wayne and Sedgewick.

All textbooks for this course are optional. Homework will not be assigned from them. Only alternate readings will be provided from them when possible.

The official description of the Java core language is available online in The Java Language Specification (Java SE 15 Edition) by James Gosling, Bill Joy, Guy Steele, Gilad Bracha, Alex Buckley, Daniel Smith, and Gavin Bierman. It's extremely thorough and easy to read (once you understand how to read it).

Software

This official coding environment and text editor for the course is the Integrated Development Environment (IDE) called IntelliJ. At your own discretion, you may instead use Vim, Emacs, Sublime, or IDEs like Eclipse, Netbeans, etc. Whatever you use, however, your submitted solutions must conform to our expected style guide. We strongly recommend that you use IntelliJ starting as soon as possible. We will not officially support any editing / programming environment other than IntelliJ.

This semester, we will use Java 15.

You will be able to do any work you'd like on any Windows, Mac OS X, or Linux computer. Information for setting up your own computer is linked in Lab 1.

We'll be using the version-control system Git this semester. Version-control systems allow you to maintain a series of "snapshots" of your files at various points in their development. Used properly, this provides you some back-up protection, so that you can recover previous states of your work when something goes wrong. Also for team-oriented projects (as well as in the real world), version-control systems help manage collaborative work.

You will be learning and using Git for this course to track your work and submit your assignments. In addition to the advantages above, using Git will allow the staff to track your progress in the course and maybe even help you out when you're stuck on bugs. The first lab will teach you the basics of what you will need to know. Feel free to also read official Git documentation.

Expected Coursework

There are five required aspects of the course for which you earn points:

1. Weekly Surveys
2. HWs
3. Lab Assignments
4. Projects
5. Exams

In addition, there are many opportunities to earn extra credit.

Weekly Survey

While lecture and section attendance is not required, nor even expected, we do expect you to stay up to date with material. To help us keep track of your progress and sentiment about the course, there will be 14 weekly surveys due on Mondays at 11:59 PM.

Each weekly survey is worth 32 points, for a total of 320 points. Only your top 10 weekly surveys are counted (out of 14). No late surveys will be accepted.

HW and Lab Assignments

There are 14 weeks of lab in the course, as well as 3 required homework.

During Phase I of the course (Weeks 1 through 5), labs will provide you with help getting your computer set up and teach you how to use essential Java programming tools, and will also include a peer code review after we're done with Project 1.

During Phases II and III of the course (Weeks 6 - 14), labs will usually involve implementation of some data structure or algorithm described in lecture. All labs should take no more than two hours to complete, though some may run slightly longer. You will turn in everything electronically using GitHub, and your results will be available on Gradescope.

Labs will receive full credit for "reasonable effort," as evaluated by a small number of relatively simple correctness tests. Alternatively, some lab assignments may not require you to pass all the tests on Gradescope to get full credit. There are no hidden tests.

Homework assignments will be conceptual assignments normally due shortly before each exam to help you study for the exams. They will be available on Gradescope and include a

combination of multiple choice and fill-in-the-blanks questions. You can expect to have to work out these problems on paper before filling in your answer on Gradescope.

Out of the 14 labs, only 10 will entail an assignment you need to submit. Each of these 10 labs will be worth 64 points (for a total of 640 points), and each homework will be worth 320 points (for a total of 960 points). No homework or labs will be dropped.

Partnerships

Coronavirus makes the world lonely. This semester, we are allowing lab partners for lab assignments. You will be able to select your own lab partner or request to be paired up with another student. You and your lab partner will be able to submit the exact same code for lab assignments under some restrictions. Having a partner is optional for labs.

For Project 3, all students will be required to work with a partner. You will either be able to choose a partner to work with or request to be paired up with another student who has similar working habits and goals as you. If you have a lab partner, you can either continue to work with your lab partner on Project 3, choose to work with a new partner, or request to be paired up with another student.

One of the main goals of CS 61B is to give you the tools to become a successful software engineer, one of which is the ability to work effectively with others. Barring extreme circumstances, we will not approve for students to officially work alone on Project 3. If you think you have a compelling reason for working alone on Project 3, we will release form that you can fill out closer to the release of the project (keep an eye on Ed for this) - we will let you know if your request is approved shortly before the release of Project 3.

Full details on partnerships can be found here.

Programming Projects

In addition to the HWs and labs, there will be 4 programming projects. In these projects you will build an entire system. All projects except project 3 must be completed on your own. For project 3 you will be required to work with a partner, unless you specifically request otherwise. If you think you can't work with a partner on this project, we will have a form for you to fill out requesting to work alone closer to the project release date. Note that requests to work alone are unlikely to be approved.

Project 0 and 1 will be relatively easier than projects 2 and 3, taking less time and with greater levels of scaffolding. Project 2 will be a difficult project (on par with what you might expect from a Hilfinger project), though it will be spread over a long period of time. Project 3 will be a capstone project in which you will design a project from scratch. Depending on how ambitious you are, it might end up being much more work than project 2.

Each project has a specific theme:

Project 0 (2048): Introduction to Java
Project 1 (Deques): Basic Design, Testing, and Code Review
Project 2 (TBA): Design, Large Scale Implementation
Project 3 (BYOW): Large Scale Design
Projects will have different grader release schedule and restrictions on the number of allowed submissions per a time interval. Full grader details will be provided on Ed with the release of each project.
Project 3 will have an autograded portion and a checkoff portion. All tests will be released for the autograder portion.

Projects 0, 1, 2, and 3 will be worth 640, 640, 1600, and 1600 points respectively.

Exams

Midterm 1 (Wed, 2/10, 8-10 PM CA Time) will be worth 1280 points, midterm 2 (Wed, 3/17, 7-9 PM CA Time) will be worth 1920 points, and the final exam (Tue, 5/11, 8-11AM CA Time) will be worth 3200 points.

You will be allowed to use unlimited handwritten sheets of notes on all exams. You will not be required to turn in these sheets, and you may reuse them from exam to exam.

Inspired by the great Paul Hilfinger tradition, exam questions may cover any material whatsoever. For fear of our lives, exams will almost exclusively test material covered in the course.

CS 61B exams will be proctored. You will need to record yourself taking every exam using Zoom. It is your responsibility to know and follow our proctoring policy closely. All the details about our proctoring process can be found in our Exam Proctoring Guide (will be released leading up to Midterm 1). We will not grade exams for students who did not follow the process described in this guide exactly, regardless of the circumstances. If you do not follow our proctoring policy correctly, or you had any proctoring issue with a midterm exam, that exam will be clobbered by your final exam score (see “exam clobbering” below). If you have a proctoring issue on the final exam, you may need to take the class for an Incomplete grade and take the final exam in a future semester (Summer 2021 or Fall 2021). Please thoroughly read the Exam Proctoring Guide before making any inquiries about the proctoring process.

There will be no alternate midterm exams. If you miss an exam, your score will be reweighted with your performance on other exams (see “exam clobbering” below). Students with disabilities that require alternate exam timing will be honored, so long as you can make a time that overlaps the official time. If you have a disability that prevents your ability to make such a time, we will discuss alternate arrangements with you directly. If you are traveling on official UC business, and have a proctor available, we will allow remote exams to be taken at the same time as the official exam.

Alternate finals will be given in case of a direct final conflict only. If you miss a final exam due to illness, you will be given an Incomplete grade in the course and will have to complete the final exam during a future semester.

We release grades for exams on Gradescope. If you believe we have misgraded an exam, request a regrade on Gradescope with a note explaining your complaint. You should check the official solutions first to make sure that this regrade will make your total score go up as it is possible to lose points from a regrade request. More details about regrade requests will be provided when we release exam grades.

Due to the university level change in drop deadlines, Midterm 1 grades will be unavailable before the drop deadline. If you're a prospective CS major and you are worried about dropping the course in the time before the drop deadline, please contact your mentor GSI and they may be able to provide you with some advice.

Exam Clobbering

For those of you who miss an exam, have a bad night, or make major improvements over the semester, the exam clobbering policy gives you a chance to replace potentially both of your midterm exam scores.

Specifically, if it helps your score, we will replace your midterm scores by their "final statistical equivalent" (FSE). We compute the FSE of an exam as follows:

Let F be the number of standard deviations above the mean that you score on the final. For example, if you are 0.3 standard deviations below the mean, F = −0.3. Let M be the class-wide mean (not including zeroes) on an midterm. Let σ be the class-wide standard deviations (not including zeroes) on a midterm. Your FSE for that exam is σ * F + M. The FSE cannot go above the maximum possible score for that exam.

If your FSEs is better than your original midterm score, we will use the FSE instead. If both are better (e.g. you do much better on the final than either midterm), then we will replace both of the midterm scores. If both of your FSE are worse, nothing happens (i.e. doing badly on the final won't hurt your earlier exam scores).

Your grade can only go up with the clobbering policy, so if replacing your midterm scores with their FSE makes your grade lower we will not do that. Exam clobbering happens automatically after we have your final exam score. You don't have to request exam clobbering to be applied.

In pseudocode, clobbering works as follows:

```
F = (your_final_score - final_mean) / final_stddev

FSE_m1 = min(m1_stddev * F + m1_mean, 1280) // the max score for midterm 1 is 1280
FSE_m2 = min(m2_stddev * F + m2_mean, 1920) // the max score for midterm 2 is 1920
```

score_with_m1_replaced = FSE_m1 + your_m2_score + your_final_score
score_with_m2_replaced = your_m1_score + FSE_m2 + your_final_score
score_with_m1_and_m2_replaced = FSE_m1 + FSE_m2 + your_final_score
score_with_no_replacements = your_m1_score + your_m2_score + your_final_score

your_total_exam_score = max(score_with_m1_replaced, score_with_m2_replaced, score_with_m1_and_m2_replaced, score_with_no_replacements)

Extra Credit

Here is a list of extra credit opportunities in the course:

- 32 points: Taking the staff created pre-semester survey.
- 2 points: Submitting Project 0 a day before the deadline or earlier.
- 16 points: Submitting a portion of Project 1 early (details will be available on the Project spec).
- 32 points: Completing the extra-credit portion of Project 1 (details will be available on the Project spec).
- 32 points: Taking the staff created mid-semester survey.
- 16 points: Submitting a portion of Project 2 early (details will be available on the Project spec).
- 32 points: Completing the extra-credit portion of Project 2 (details will be available on the Project spec).
- 32 points: Completing the extra-credit portion of Project 3 (details will be available on the Project spec).
- 32 points: Taking the staff created end-of-semester survey.
- 32 points: Taking the official university end-of-semester survey.

Point another way, there are 258 total points of extra credit:

- 128 points for the special surveys (not the same thing as the weekly surveys).
- 34 points for completing portions of assignments before the deadline.
- 96 points for completing extra-credit portions of projects.

Grades

Your letter grade will be determined by the total points out of the possible 12,800. In other words, there is no curving in this course, other than the clobbering policy above. Your grade will depend solely on how well you do, and not on how well everyone else does. Unlike other lower division CS courses, the grading bins for 61B generally do not get tweaked at the end of the semester.

Category	Percentage	Points
Homework/Labs	12.5%	1600
Surveys	2.5%	320
Projects	35%	4480
Midterms	25%	3200
Final Exam	25%	3200
Total	100%	12800

A+	A	A-	B+	B	B-	C+	C	C-	D+	D	D-	F
12400	11600	10900	9900	9200	8600	8000	7400	6000	5100	4400	3200	0

These bins were designed to comply with departmental guidelines that the average GPA for a lower-division required course be in the range 2.8 - 3.3 (including students who drop or take the class for a P/NP grade, given the special circumstances surrounding the Coronavirus pandemic). The design process involved setting of specific standards I expect students to achieve for the A, B, and C bins, with numbers adjusted and other bins interpolated based on a model that I built of predicted student performance. At the end of the semester, we might make the bin boundaries slightly friendlier, though as noted above, unlike other lower division CS courses, I don't typically move them very much, if at all.

Since the class is not curved, and we provide all the grading details above, occasionally students will only be "a few points away" from the next grading bin after they receive their final total points value. Please do not contact us in this case. We cannot "round up" your grade or give you credit for work you did not complete. If you ever find yourself unable to complete an assignment, please refer to our friendly lateness policy (details below), or contact us before the deadline.

We will grant grades of Incomplete 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. Do not try to get an incomplete simply as a way to have more time to study or do a project. That is contrary to University policy. Before requesting an Incomplete grade, please contact a college advisor or review your college's Incomplete grade policies to understand if this is a right option for you.

Policies on Collaboration, Cheating, and Lateness

Deadlines can be stressful, and we know that under extreme pressure, it becomes tempting to start rationalizing actions that you would otherwise consider inappropriate. Perhaps you'll find yourself facing a CS 61B project deadline, and under all this stress you'll convince yourself that you're just going to cheat for the moment so you can get the points, and that you'll come back later and really learn the thing you were supposed to have learned in order to restore your karmic balance (I've heard something along these lines a few times).

This is a terrible idea. Obviously it's important to learn how to deal with deadlines, but far more important than that, giving into this sort of pressure under not-so-dire circumstances is going to do some damage to your moral compass. Someday, when the consequences are higher than potentially losing a 1/3rd of a letter grade, you may find yourself committing dishonest acts at the cost of someone else's livelihood or life.

Plagiarism on any homework, lab or project will result in a score of -200 on that assignment, which will likely reduce your letter grade by several bins. A second instance of plagiarism on a homework, lab, or project will result in an F in the course. All incidents of plagiarism will be referred to the Office of Student Conduct, including carelessly leaving code up on GitHub. Given our friendly lateness policy (see below), there will be no exceptions to this rule.

During the Spring 2017 semester, we compiled a series of incident reports written by students who were caught plagiarizing. If you find yourself tempted to cheat, you might turn to the words of others who have made the wrong choice for guidance.

In CS 61B, we have three types of assignments: homework, labs, and projects. The entire point of homework and labs is to learn. For homework or labs, you should feel free to collaborate with others however you choose, though keep in mind that greater independence is likely to give you a better learning experience (as long as you aren't totally stuck). Even though we will allow close collaborations on HW and labs, the solutions you submit should still be your own work! Identical or near identical submissions will be treated as plagiarism. Lab partners will be allowed to submit identical code as long as the submission is made from their shared repository (details above).

By contrast, the projects were designed not just for learning (particularly how to be self-reliant in the context of large unfamiliar systems), but also for the dual purpose of evaluating your mastery of the course material. As such, they are intended to be completed primarily on your own (or with your partner on the last project), particularly when it comes to writing the actual code.

For exams, we will be absolutely unforgiving. Any incident will result in a failing grade for the course, though Berkeley will let you retake CS 61B next semester. As above, all incidents of cheating will be referred to the Office of Student Conduct.

What constitutes cheating? **The golden rule of academic dishonesty is that you should not claim to be responsible for work that is not yours.**

This is obviously open to some interpretation, and you'll be getting some help from instructors, the internet, other students, and more throughout the course. This is OK, and we hope that the class is an open, welcoming, collaborative environment where we can help each other build the highest possible understanding of the course material.

To help (but not entirely define) the bounds of acceptable behavior, we have three important rules for projects:

1. By You Alone: All project code that you submit (other than skeleton code) should be written by you (and if applicable, lab partner on lab assignments and/or your partner on project 3) alone, except for small snippets that solve tiny subproblems (examples in the Permitted section below).
2.
3. Do Not Possess or Share Code: Before you've submitted your final work for a project, you should never be in possession of solution code that you (or your partner) did not write. You will be equally culpable if you distribute such code to other students or future students of 61B (within reason). DO NOT GIVE ANYONE YOUR CODE – EVEN IF THEY ARE DESPERATELY ASKING. DO NOT POST SOLUTIONS TO PROJECTS ONLINE (on GitHub or anywhere else)! If you're not sure what you're doing is OK, please ask.
4.
5. Cite Your Sources: When you receive significant assistance on a project from someone else, you should cite that assistance somewhere in your source code with the @source tag as described in Lab 1. We leave it to you to decide what constitutes 'significant'.

For clarity, examples of specific activities are listed below:

Permitted:

- Discussion of approaches for solving a problem.
- Giving away or receiving significant conceptual ideas towards a problem solution. Such help should be cited as comments in your code. For the sake of other's learning experience, we ask that you try not to give away anything juicy, and instead try to lead people to such solutions.
- Discussion of specific syntax issues and bugs in your code.
- Using small snippets of code that you find online for solving tiny problems (e.g. googling "uppercase string java" may lead you to some sample code that you copy and paste into your solution). Such usages should be cited as comments in your hw, lab, and especially project code!

Permitted with Extreme Caution:

- Looking at someone else's project code to assist with debugging. Typing or dictacting code into someone else's computer is a violation of the "By You Alone" rule.
- Looking at someone else's project code to understand a particular idea or part of a project. This is strongly discouraged due to the danger of plagiarism, but not absolutely forbidden. We are very serious about the "By You Alone" rule!
- Working on a project alongside another person or group of people. Your code should not substantially resemble anyone else's!

Absolutely Forbidden:

- Possessing another student's project code in any form before a final deadline, be it electronic or on paper. This includes the situation where you're trying to help someone debug. Distributing such code is equally forbidden.
- Possessing project solution code that you did not write yourself (from online (e.g. GitHub), staff solution code found somewhere on a server it should not have been, etc.) before a final deadline. Distributing such code is equally forbidden.
- Posting solution code to any assignment in a public place (e.g. a public git repository, mediafire, etched into stones above the Mediterranean, etc). This applies even after the semester is over.
- Working in lock-step with other students. Your workflow should not involve a group of people identifying, tackling, and effectively identically solving a sequence of subproblems.

We have advanced cheating detection software, and we will routinely run this code to detect cheating. Every semester, we catch and penalize a significant number of people (roughly 100 cases per semester). Do not be one of them. If you find yourself at such a point of total desperation that cheating begins to look attractive, contact one of the instructors and we can maybe help somehow. Likewise, if 61B is causing massive disruption to your personal life, please contact us directly.

If you admit guilt to an act of plagiarism before we catch you, you will be given zero points on that assignment, and we will not refer your case to the university administration.

Obviously, the expressive power of Java is a subset of the English language. And yes, you can obviously obey the letter of this entire policy while completely violating its spirit. However, this policy is not a game to be defeated, and such circumventions will be seen as plagiarism.

Lateness

The expected deadlines for each assignment are posted on the class website. Our assumption is that you will complete each assignment by its due date.

However, we understand that life gets in the way, and especially with the Coronavirus, unexpected circumstances may arise.

It is possible to submit work late by requesting an "extension token" for the assignment using the Extension tab on Beacon. You can find the full instructions on submitting extensions on the Beacon guide. The system will grant an extension token as long as:

1. You have not used up your slip time.
2. The extension is for less than 72 hours.

Every student will starts with a total of 6 days worth of slip time that can be used on labs or projects (including extra credit assignments). In order for us to release the solutions to conceptual homework assignments on time, you may not use any slip days on homework.

For example, consider a student who has:

1. Submitted HW2 3 hours late.
2. Submitted Project 2 35 hours late.
3. Submitted Lab 7 16 hours late.
4. Realizes four days after the deadline that they forgot to submit Project 2.

This student has used 2.25 of their slip days and they have 3.75 remaining. This student will be unable to submit project 2, because it is already more than 3 days late, and will therefore receive a score of zero on project 2.

Slip days are intended to cover life events and disability accomodations. Students who have dire life circumstances that require additional time beyond the 6 slip days should email our staff email account (cs61b@berkeley.edu) if their supply becomes too low, and additional slip days may be granted if appropriate.

You can request an extension for the extra credit portions of the different projects, if available. They usually appear at the end of every project spec. However, you cannot request extensions on project checkpoints. These are special graders that allow you to submit a subset of the assignment early (more detalis on each project's spec). Similarly, you can get extra credit for submitting Project 0 a day early, and will not be able to request an extension for this opportunity.

Additionally, you will not be able to request extensions for homework assignments or Project 3 Part B.

Finally, you will be able to use a maximum of 2 slip days on Project 1.

Note that while you can submit assignments late under these rules, staff will not provide help with an assignment passed its deadline. Thus, all Ed posts, Gitbugs, and Office Hours tickets concerning a past-due assignment will be ignored.

Auditing CS61B

This is for students who are unofficially auditing this class. Maybe you are a non-Berkeley student who wants to brush up on your programming knowledge or even a high school student who wants to get a head start on your programming career.

You can follow along with our lectures and assignments, getting all the code from our skeleton as described in lab 1. The only difference is that you won't have a CS61B GitHub repository, you'll have to make your own.

We also have an autograder on Gradescope set up just for you! The Gradescope course activation code is P5WVGW. We'll release the autograder for assignments about 3 days after it is due for Berkeley students.

Acknowledgements
Some course handout material derived from Paul Hilfinger's CS61B handout
https://inst.eecs.berkeley.edu/~cs61b/fa14/handout0.pdf