Eric Roberts
CS 106B

Handout #2
January 5, 2015

CS 106B — General Information

Professor: Eric Roberts

E-mail:	eroberts@cs.stanford.edu
Offices:	Gates 202
Phone:	723-3642
Drop-in hours:	Tuesdays, 9:30–11:30 A.M.
	Wednesdays after class in Bytes Café

TA: Kevin Miller

E-mail:	kmiller4@stanford.edu
Office:	Gates 160
Phone:	714-768-4954
Office hours:	Wednesdays, 12:30–2:00 P.M.
	Thursdays, 1:00–3:00 P.M.

Lectures

Lectures are scheduled for Monday, Wednesday, and Friday at 2:15 P.M. in Hewlett 200. The schedule for individual lectures appears on pages 5 and 6 of this handout. Although CS 106B lectures are usually videotaped only in the spring, I was able to convince the Stanford Center for Professional Development to record this quarter's lectures as well.

Discussion sections

In addition to lecture, you **must** sign up for a weekly 50-minute section. In order to take CS 106B, you must sign up for a section between 5:00 P.M., Thursday, January 8, and the close of enrollments at 5:00 P.M. on Sunday, January 11. Please use the signup form at http://cs198.stanford.edu/section/. **Important note: Do not use Axess to sign up for sections; section enrollments made through Axess are simply ignored.**

Units

If you are an undergraduate, you are required to take CS 106B for 5 units of credit. If you are a graduate student, you may take CS 106B for 3 or 4 units if it is necessary for you to reduce your units for administrative reasons. Taking the course for reduced units does not imply any change in the course requirements. **Important note: The default on Axess is to sign you up for 3 units; make sure that you change this entry to 5.**

Drop/add deadlines

You may not add or drop a course after 5:00 P.M. on Friday, January 23. The deadline for withdrawing from a course or for changing the grading status is 5:00 P.M. on Friday, February 27.

Web page

The web page for CS 106B is http://www.stanford.edu/class/cs106b/. All course materials will be posted on this site, so be sure to check it frequently.

Text and handouts

The text for this course is a near-final draft of the forthcoming textbook *Programming Abstractions in C++*, which is available from the Stanford Bookstore. The textbook is also available on the class web site, so that you can refer to it online. You will, however, almost certainly want a copy because you will need it for the open-book exams.

As a supplement to the text, we will distribute additional material in the form of class handouts. The handouts are also available in PDF® format on the CS 106B web site. If you miss a handout in class, you can print your own copy from the web.

Examinations

The midterm examination will be administered at two different times on **Tuesday, February 10: from 2:15 to 4:15 P.M. and from 7:00 to 9:00 P.M.** The final examination will be offered only at the special "Introductory CS courses" group time: **Tuesday, March 17 from 8:30–11:30 A.M.** The examinations themselves are written so that you should be able to complete them in less than the allotted time: the midterm is designed to take one hour and the final is designed to take 100 minutes. All examinations are open-book, and you may use any notes or materials from the class.

Programming assignments

CS 106B requires seven programming assignments, which are due on the dates given in the syllabus. Except for Assignment #7 (which is due at the very end of the quarter), your section leader will grade each of your assignments and then discuss how well you have done (and how you can improve on future assignments) in an interactive, one-on-one session. Assignments are graded separately for functionality and style using the following scale:

++ An absolutely fantastic submission of the sort that will only come along a few times during the quarter. To ensure that this score is given only rarely, any grade of ++ must be approved by the instructor and TA. Since your section leader would almost certainly want to show off any assignment worthy of a ++, this review process should not be too cumbersome.

\+ A submission that exceeds our standard expectation for the assignment. The program must reflect additional work beyond the requirements or get the job done in a particularly elegant way.

✓+ A submission that satisfies all the requirements for the assignment—a job well done.

✓ A submission that meets the requirements for the assignment, possibly with a few small problems.

✓– A submission that has problems serious enough to fall short of the requirements for the assignment.

– A submission that has extremely serious problems, but nonetheless shows some effort and understanding.

– – A submission that shows little effort and does not represent passing work.

From past experience, we expect most grades to be ✓+ and ✓. Dividing the grades into categories means that your section leader can spend more time talking about what you need to learn from the assignment and not have to worry about justifying each point.

For each assignment, you must make an appointment with your section leader for an interactive-grading session. Your section leader will explain in section how to schedule these sessions and go over the grading process in more detail.

Late policy

Each of the assignments is due at 5:00 P.M. on the day specified in the syllabus. Assignments are submitted electronically using the procedure described in the first assignment. **Important note: all assignments are due at 5:00 P.M. sharp on the dates indicated on the assignment handout. Assignments submitted after 5:00 will be considered late.**

Because each of you will probably come upon some time during the quarter where so much work piles up that you need a little extra time, every student begins the quarter with two free "late days." To avoid any ambiguity, a "day" is defined as a class day. Thus, if your assignment was due on Friday but turned in the following Monday, that assignment would be one day late. (Note: Assignment #7, which is due at the end of the quarter after classes have finished, has its own late-day policy in which "days" revert to calendar days.) After your late days for the quarter are exhausted, programs are assessed a late penalty of one category point per late day used (a ✓+ turns into a ✓, and so forth). Late days are valuable, and it pays to keep some around for the harder assignments toward the end of the quarter. In all cases, assignments must be turned in within a calendar week of their published due date.

In special circumstances (such as extended medical problems or other emergencies), extensions may be granted beyond the late days. To request an extension, send e-mail to the TA (Kevin Miller **<kmiller4@stanford.edu>**) no later than 24 hours before the program is due. Only the TA is authorized to approve such extensions. In particular, do not ask your section leader.

Problem sets

In addition to the programming assignments, there will also be two problem sets during the quarter that consist of pencil-and-paper problems of the sort that you are likely to see on the midterm and the final. The first problem set covers the C++ memory model and is due on Friday, February 6. The second covers the mechanics of various algorithms for trees and graphs and is due on Friday, March 6. Late days may not be applied to the problem sets.

Contests

As shown on the calendar handout, there are three contests scheduled at different points during the term. The point of these contests is to give you a chance to show creativity and initiative beyond what is formally required by the course. Rules for each contest will be distributed in class when they are announced. The grand prize in each contest is a perfect score on one of the graded components of the course, typically the final exam.

Beyond the contests there are additional incentives to encourage your participation. First, every serious entry gets you one virtual ticket in a random drawing for a special grand prize at the end of the quarter. Second, winning a runner-up prize or honorable mention in a contest or having an assignment judged as a ++ candidate gives you additional chances in the random drawing, which will take place at the beginning of the review for the final exam.

Grading

Your final grade in this course will be calculated from your scores on the individual assignments using the following weights:

Programming assignments	50%
Problem sets	5%
Final examination	20%
Midterm examination	15%
Section participation	10%

Each of the exams is scaled to take account of the difficulty of the exam, which is nearly impossible to assess in advance. The other components and the final grade are not curved. If everyone does brilliantly in this class, everyone gets an A.

Computer facilities

Programming assignments can be implemented on Macs, Windows machines, or Linux workstations, using an open-source implementation platform called Qt Creator. Information on how to download and use this software will be provided in separate handouts. Many of you will do your work in the Tresidder LaIR computer cluster, which is staffed most evenings with section leaders who can help if you get stuck.

The Stanford Honor Code

One of the sad realities of teaching in this field is that computer science courses account for a disproportionately large share of Honor Code cases at Stanford. What's more, certain forms of cheating—particularly copying posted solutions from the web without attribution—seem to be increasingly prevalent over time. Further details about what is and is not permissible under the Honor Code appear in Handout #3.

Given recent history, we believe that it is essential for this course to take concerted action to deter such behavior. To this end, we will run the MOSS (Measure of Software Similarity) program on all major assignments to identify suspicious code. If we discover cases that represent clear transgressions of academic integrity, the course staff will pursue those cases aggressively through the Office of Community Standards.

CS 106B Calendar

Monday	Wednesday	Friday
January 5 Course overview The big ideas in CS 106B The C++ language C++ *vs.* Java **Read: Chapter 1**	7 Functions in C++ Call by reference Libraries and interfaces Recursive functions **Read: Chapters 2 and 7**	9 Using the `string` class File streams Class hierarchies **Read: Chapters 3 and 4**
12 Abstract data types Using `Vector` and `Grid` Stacks and queues **Read: Sections 5.1-5.3**	14 `Map`, `Set`, and `Lexicon` Iterating over a collection **Read: Sections 5.4-5.6**	16 Designing classes The `TokenScanner` class **Read: Chapter 6** **Due: HW #1 (Simple C++)**
19 Martin Luther King, Jr. Day Optional film: Dr. King's 1963 speech "I Have A Dream"	21 Procedural recursion The Towers of Hanoi Graphical recursion **Read: Chapter 8**	23 Recursive backtracking Solving a maze **Read: Section 9.1**
26 Backtracking and games The minimax algorithm **Read: Sections 9.2-9.3** **Due: HW #2 (ADTs)**	28 Algorithmic efficiency Big-O notation Sorting algorithms **Read: Chapter 10**	30 The C++ memory model Pointers **Read: Chapter 11** **Due: RandomWriter contest**
February 2 Dynamic allocation **Read: Sections 12.1-12.8**	4 The `editor.h` interface **Read: Sections 13.1-13.3** **Due: HW #3 (Recursion)**	6 Implementing editors **Read: Sections 13.4-13.5** **Due: Problem set #1**

Monday	Wednesday	Friday
February 9 Templates Implementing stacks **Read: Sections 14.1-14.2**	11 Implementing queues Implementing vectors Midterm exam Tuesday, February 10 2:15 or 7:00 P.M. **Read: Sections 14.3-14.4**	13 The `StringMap` class The idea of hashing Implementing `HashMap` **Read: Chapter 15** **Due: HW #4 (Boggle)**
16 Presidents' Day (no class)	18 Binary search trees Balanced trees Implementing `Map` **Read: Sections 16.1-16.4** **Due: Recursion contest**	20 Sets in mathematics Implementing sets **Read: Sections 17.1-17.3**
23 Graphs Standard traversals **Read: Sections 18.1-18.4** **Due: HW #5 (PQueue)**	25 Graph algorithms Shortest-path algorithms Minimum spanning trees **Read: Sections 18.5-18.6**	27 Inheritance in C++ Defining shape classes **Read: Sections 19.1-19.2**
March 2 Expression trees Representing expressions **Read: Section 19.3**	4 Parsing strategies Overview of BASIC **Read: Section 19.4** **Due: HW #6 (MazeMaker)**	6 C++ in the real world Using the STL The mysteries of `const` **Due: Problem set #2**
9 Advanced algorithms Google's Page Rank DAWGs and lexicons Heaps and priority queues **Read: Sections 16.5, 18.7**	11 Strategies for iteration Function pointers Function objects The `<algorithm>` library **Read: Chapter 20**	13 Further adventures in CS (optional) **Due: HW #7 (BASIC)** **Due: BASIC contest**
Final Exam Tuesday, March 17 8:30–11:30 A.M.		