Syllabus

version 1.4

Instructor

David J. Malan '99
malan@post.harvard.edu
http://www.cs.harvard.edu/malan/
+1-617-523-0925

Description

Introduction to the intellectual enterprises of computer science and the art of programming. This course teaches students how to think algorithmically and solve problems efficiently. Topics include abstraction, encapsulation, data structures, databases, memory management, software development, virtualization, and websites. Languages include C, PHP, and JavaScript plus SQL, CSS, and XHTML. Problem sets inspired by real-world domains of biology, cryptography, finance, forensics, and gaming. Designed for concentrators and non-concentrators alike, with or without prior programming experience.

This course, when taken for a letter grade, meets the General Education requirement for Empirical and Mathematical Reasoning or the Core area requirement for Quantitative Reasoning.

Expectations

You are expected to attend all lectures and sections, complete nine problem sets, take two quizzes, and produce a final project.

Grades

Final grades will be determined using the following weights; remarkable effort will not go unnoticed.

Problem Sets (best 8 out of 9):	60%
Quizzes:	30%
Final Project:	10%

You may take this course pass/fail. Realize, though, that some departments (Computer Science, Applied Math, *et al*.) do not allow courses taken pass/fail to satisfy concentration requirements. Moreover, you'll certainly learn more by immersing yourself in this course, in which case you might just want to take it for a grade. But it's up to you.

Know that CS 50 draws quite the spectrum of students, including "those less comfortable," "those more comfortable," and those somewhere in between. However, what ultimately matters in this course is not so much where you end up relative to your classmates but where you, in Week 12, end up relative to yourself in Week 0.

Website

The address of this course's website is:

```
http://www.cs50.net/
```

Visit the course's website to access the course's bulletin board, enter the virtual terminal room, check your grades, watch videos, download handouts and software, and follow links to other resources.

Lectures

Lectures take place in Sanders Theater on Mondays and Wednesdays from 1:00pm until 2:30pm[*].

Each lecture is filmed and made available within 72 hours via podcast (for download to iTunes and iPods) and via the course's website in Flash, MP3, and QuickTime formats. Once posted, these recordings remain available until semester's end. You are welcome to watch or listen to a recording if you are unable to attend some lecture in person. You are encouraged to watch or listen to these recordings for the sake of review.

A schedule of lectures, subject to change, appears below.

Week 0
Wed 9/2, Fri 9/4

> Introduction. Bits. Binary. ASCII. Programming. Algorithms. Scratch. Statements. Boolean expressions. Conditions. Loops. Variables. Threads. Events.

Week 1
Wed 9/9, Fri 9/11

> C. Source code. Compilers. Object code. SSH. SFTP. GCC. Functions. Comments. Standard output. Arithmetic operators. Precedence. Associativity. Local variables. Types. Casting. Standard input. Libraries. Boolean expressions, continued. Conditions, continued. Loops, continued.

Week 2
Mon 9/14, Wed 9/16

> Functions, continued. Global variables. Parameters. Return values. Stack. Frames. Scope. Arrays. Strings. Command-line arguments. Cryptography.

Week 3
Mon 9/21, Wed 9/23

> Linear search. Binary search. Asymptotic notation. Recursion. Pseudorandomness. Bubble sort. Selection sort. Insertion sort. Merge sort. Debugging.

Week 4
Mon 9/28, Wed 9/30

> Structures. Dynamic memory allocation. Stack and heap. Pointers. Debugging, continued.

[*] This course will also meet Fri, 1–2:30pm on 9/4/09 and 9/11/09 only. Students with conflicts should watch those lectures online.

Week 5
Mon 10/5, Wed 10/7

File I/O. Forensics. Linked lists. Stacks. Queues.

Week 6
Wed 10/14

Quiz 0 on Wed 10/14.

Week 7
Mon 10/19, Wed 10/21

Valgrind. Bitwise operators. Hash tables. Trees. Binary search trees. Tries. Huffman coding.

Week 8
Mon 10/26, Wed 10/28

HTTP. XHTML. PHP. SQL.

Week 9
Mon 11/2, Wed 11/4

CSS. Inheritance. JavaScript. Events, continued. Ajax.

Week 10
Mon 11/9

Preprocessing. Compiling. Assembling. Linking. CPUs.

Week 11
Mon 11/16, Wed 11/18

Guest lectures.

Quiz 1 on Wed 11/18.

Week 12
Mon 11/23

Exciting conclusion.

Sections

Lectures are supplemented by weekly, 90-minute sections led by the teaching fellows. Sections provide you with an opportunity to review and discuss course materials in a more intimate environment, with only your teaching fellow and a handful of classmates present. Moreover, the teaching fellows supplement material from lecture with additional examples and implementation details as well as provide further guidance for problem sets and quizzes.

Different sections are offered for those less comfortable, those more comfortable, and those somewhere in between.

A schedule of sections appears on the course's website.

Walkthroughs

On Sunday nights from 7:00pm until 8:30pm, the staff hold a "walkthrough" for the current week's problem set during which you receive direction on where to begin and how to approach the week's challenges.

Each walkthrough is filmed and made available within 48 hours via podcast (for download to iTunes and iPods) and via the course's website in Flash, MP3, and QuickTime formats. You are encouraged to watch or listen to these recordings before asking questions at office hours.

Office Hours

Throughout the week, the staff hold office hours in Science Center B14 (the "terminal room") during which you can receive hands-on, one-on-one assistance with problem sets. The staff also hold virtual office hours in the course's "virtual terminal room," where you can receive remote, one-on-one assistance via the Web. Thanks to technology, the staff can help you troubleshoot bugs by observing or sharing control of your screen while chatting with you via IM or VOIP, whether you're in your dorm room, at Brain Break, at Starbucks, or beyond.

A schedule of office hours appears on the course's website.

Problem Sets

Nine problem sets are assigned during the semester. Each is due via electronic submission seven or more days after its date of distribution. However, you have nine "late days" that you may "spend" during the semester, each of which provides you with an extension of twenty-four hours. In other words, you may submit one problem set nine days late, each problem set one day late, *etc*. It is expected that you alert your teaching fellow to your use of any late days for some problem set prior to or upon its actual deadline. Lateness of electronic submissions is determined down to the minute by submissions' timestamps. Submitting one minute late is considered equivalent to submitting twenty-

four hours late. If you choose not to submit some problem set at all, you will not be charged any late days, but you will receive a score of zero. Late work is not accepted once you have exhausted your nine late days, except in cases of emergency. Technical difficulties are not considered emergencies. These late days cannot be spent on the course's final project.

In order to accommodate students with different backgrounds, some problem sets are released in two editions: a standard edition intended for most students and a "Hacker Edition" intended for some students. Both editions essentially cover the same material. But the Hacker Edition typically presents that material from a more technical angle and poses more sophisticated questions. Hacker Editions are graded separately from standard editions, but those students who submit the former do not receive any form of extra credit outright. When awarding letter grades at term's end, however, we do bear in mind submissions of Hacker Editions.

To be clear, we encourage most students (including aspiring computer scientists) to tackle the standard editions. However, you may choose, week to week, which edition to submit. You may not submit both or some amalgam of the two.

When final grades are computed, your lowest score on these nine problem sets is dropped; your eight highest scores are weighted equally.

A schedule of problem sets, subject to change, appears below.

Problem Set 0: Scratch
due by 7:00pm on Fri 9/11

Create your own animation, game, or interactive art.

Problem Set 1: C
due by 7:00pm on Fri 9/18

Meet Linux and C.

Problem Set 2: Crypto
due by 7:00pm on Fri 9/25

Encrypt and decrypt sensitive information.

Problem Set 3: The Game of Fifteen
due by 7:00pm on Fri 10/2

Implement a party favor.

Problem Set 4: Sudoku
due by 7:00pm on Fri 10/9

数字は独身に限る。

Problem Set 5: Forensics
due by 7:00pm on Fri 10/23

Recover lost photos. Solve a murder mystery.

Problem Set 6: Mispellings
due by 7:00pm on Fri 10/30

Implement a spell-checker that's faster than your classmates'.

Problem Set 7: C$50 Finance
due by 7:00pm on Fri 11/6

Design a database. Build a dynamic website.

Problem Set 8: Mashup
due by 7:00pm on Fri 11/13

Google Maps meet Google News. And Ajax.

Quizzes

The course has two 75-minute quizzes. These quizzes are "closed-book," but you may utilize during each quiz one two-sided page (8.5" × 11") of notes, typed or written, and a pen or pencil, nothing else.

When final grades are computed, your scores on these two quizzes are weighted equally.

A schedule of quizzes, subject to change, appears below; these quizzes take place in lieu of lectures on these dates.

Quiz 0
Wed 10/14

Covers weeks 0 through 5.

Quiz 1
Wed 11/18

Covers weeks 0 through 10 with emphasis on 7 onward.

Unless arranged with the staff in advance, quizzes may not be taken at alternative times even if missed by accident, except in cases of emergency.

Final Project

The climax of this course is its final project. The final project is your opportunity to take your knowledge of programming out for a spin and develop your very own piece of software. So long as your project draws upon this course's lessons, the nature of your project is entirely up to you, albeit subject to the staff's approval. You may implement your project in any language(s) as long as the staff approves. You are welcome to utilize infrastructure other than `nice.fas.harvard.edu` and `cloud.cs50.net`, provided the staff ultimately has access to any hardware and software that your project requires. All that we ask is that you build something of interest to you, that you make something useful, that you solve an actual problem, or, best yet, that you somehow impact campus. Strive to create something that outlives this course.

This semester will conclude with The CS 50 Fair, a campus-wide exhibition of final projects.

Inasmuch as software development is rarely a one-person effort, you are allowed an opportunity to collaborate with one or two fellow students for this final project. Needless to say, it is expected that every student in any such group contribute equally to the design and implementation of that group's project. Moreover, it is expected that the scope of a two- or three-person group's project be, respectively, twice or thrice that of a typical one-person project. A one-person project, mind you, should entail more time and effort than is required by each of the course's problem sets.

Guidelines for the final project will be distributed by Mon 10/26. A schedule, subject to change, appears below.

Pre-Proposal
due by 11:00am on Mon 11/9

Proposal
due by 11:00am on Mon 11/16

Status Report
due by 11:00am on Mon 11/30

Implementation
due by 11:00am on Mon 12/7

The CS 50 Fair
from 1:00pm until 4:30pm on Tue 12/8

Extensions on the final project are not granted, except in cases of emergency. Technical difficulties are not considered emergencies. Problem sets' late days cannot be spent on the final project. Late submissions may be penalized 1% per minute late up to a maximum of 100%. Lateness of submissions is determined by submissions' timestamps.

Books

No books are required for this course.

However, you may want to supplement your preparation for or review of some lectures with self-assigned readings relevant to those lectures' content from either of the books below. The first is intended for those inexperienced in (or less comfortable with the idea of) programming. The second is intended for those experienced in (or more comfortable with the idea of) programming. Both are available for purchase at the Coop and at sites like Amazon.com. Both of these books have been placed on reserve at Cabot Science Library. Realize that free, if not superior, resources can be found on the course's website.

For Those Less Comfortable

Absolute Beginner's Guide to C, Second Edition
Greg Perry
Sams Publishing, 1994
ISBN 0-672-30510-0

For Those More Comfortable

Programming in C, Third Edition
Stephen Kochan
Sams Publishing, 2004
ISBN 0-672-32666-3

The book below is recommended for those interested in understanding how their own computers work for personal edification. It is also available for purchase at the Coop and at sites like Amazon.com. It, too, has been placed on reserve.

How Computers Work, Ninth Edition
Ron White
Que Publishing, 2007
ISBN 0-7897-3613-6

This last book below is recommended for aspiring hackers, those interested in programming tricks and low-level optimization of code for applications beyond the scope of this course. It is also available for purchase at the Coop and at sites like Amazon.com. It, too, has been placed on reserve.

Hacker's Delight
Henry S. Warren Jr.
Addison-Wesley, 2003
ISBN 0-201-91465-4

Academic Honesty

All work that you do toward fulfillment of this course's expectations must be your own unless collaboration is explicitly allowed (*e.g.*, by some problem set or the final project). Viewing or copying another individual's work (even if left by a printer, stored in an executable directory, or accidentally shared in the course's virtual terminal room) or lifting material from a book, website, or other source—even in part—and presenting it as your own constitutes academic dishonesty, as does showing or giving your work, even in part, to another student.

Similarly is dual submission academic dishonesty: you may not submit the same or similar work to this course that you have submitted or will submit to another. Nor may you provide or make available solutions to problem sets to individuals who take or may take this course in the future. Moreover, submission of any work that you intend to use outside of the course (*e.g.*, for a job) must be approved by the staff.

You are welcome to discuss the course's material with others in order to better understand it. You may even discuss problem sets with classmates, but you may not share code. In other words, you may communicate with classmates in English, but you may not communicate in, say, C. If in doubt as to the appropriateness of some discussion, contact the staff.

You may even turn to the Web for instruction beyond the course's lectures and sections, for references, and for solutions to technical difficulties, but not for outright solutions to problems on problem sets or your own final project. However, failure to cite (as with comments) the origin of any code or technique that you do discover outside of the course's lectures and sections (even while respecting these constraints) and then integrate into your own work may be considered academic dishonesty.

All forms of academic dishonesty are dealt with harshly.