CS16: Problem Solving with Computers I
Summer 2012 (Session C)

Instructor: Kyle Dewey (kyledewey@cs.ucsb.edu)
Teaching Assistant: Joshua Dickinson (dickinson@mat.ucsb.edu)
Grader: Praveen Kumar Rajamani (praveenkumar@umail.ucsb.edu)

Website: http://cs.ucsb.edu/~kyledewey/cs16
Lecture: Tuesday / Thursday 2:00 PM - 3:15 PM in Phelps 1401
Lab: Wednesday 2:00 PM - 3:15 PM; 3:30 PM - 4:45 PM in ESB 1003 (Cooper Lab)

Textbook: Engineering Problem Solving with C, 3rd Edition (by Delores M. Etter) (Not strictly required, but **strongly** recommended. The majority of the book will be covered, and it reinforces the material well.)

Course Office Hours: TBA; refer to http://cs.ucsb.edu/~kyledewey/cs16/hours for up-to-date information. I am also available by appointment.

Course Description:
(In my own words) A survey of the fundamentals necessary for basic computer programming. C is used as the language of choice, though the techniques learned should be broadly applicable.

(From the course catalog) Fundamental building blocks for solving problems using computers. Topics include basic computer organization and programming constructs: memory CPU, binary arithmetic, variables, expressions, statements, conditionals, iteration, functions, parameters, recursion, primitive and composite data types, and basic operating system and debugging tools.

Prerequisites:
- Math 3A (may be taken concurrently)
- Students with no experience with computer programming are encouraged to take Computer Science 5 or 8 before Computer Science 16.
- Also a “legal repeat” of CS10

Grading:
- Exams: 3 @ 20% apiece = 60%
- Labs (including pre-labs): 9 @ 3% apiece = 27%
- Projects: 3 @ (variable apiece) = 13%

Course Outline: (subject to change)

Week	Day	Topic	Event
1	6/26	Syllabus / Editors / UNIX	Lab 1 due 7/3 (pre-lab in lecture if not in lab)
1	6/28	int, char,variables, data representation	
2	7/3	printf/scanf, variable assignment, expressions	
2	7/5	floating point, external libraries, errors	
3	7/10	basic functions, statements vs. expressions	Lab 2 due 7/11 (pre-lab in lab)
3	7/12	selection: if/else/switch, exam review	Proj 1 assigned
4	7/17	**Exam 1**	Lab 3 due 7/18 (pre-lab in lab)
4	7/19	repetition: while/for/do-while	Proj 1 due 7/23
5	7/24	File I/O	Lab 4 due 7/25 (pre-lab in lab), Proj 2 Assigned
5	7/26	Arrays and pointers	
6	7/31	Scope and lifetime, macros, testing	Lab 5 due 8/1 (pre-lab in lab)
6	8/2	Selection sort, basic structs	
7	8/7	More complex data	Lab 6 due 8/8 (pre-lab in lab), Proj 2 due 8/10
7	8/9	Recursion, exam review	
8	8/14	**Exam 2**	Lab 7 due 8/15 (pre-lab in lab)
8	8/16	Strings, multidimensional arrays, dynamic memory allocation	Proj 3 Assigned
9	8/21	Assertions, more with structs, enums	Lab 8 due 8/22 (pre-lab in lab)

Week	Day	Topic	Event
9	8/23	Bit manipulation, efficiency	
10	8/28	Exam review	Lab 9 due 8/29 (pre-lab in lab)
10	8/30	**Exam 3**	Proj 3 due 8/31

On Learning to Program:
While it is assumed that students have been somehow exposed to programming before, this course is intended to be the first real "dive" into the subject. Learning to program can be a fun, even exhilarating experience, but it can also be difficult, frustrating, and above all, time-consuming. It sits is at the intersection of art, science, and engineering, and it requires a different way of thinking that takes time to develop. Be forewarned: it will not be easy, and it will demand a serious time investment.

On Pair Programming:
Some of the labs will encourage students to do pair programming. In pair programming, one person is the "driver" who actually writes code, and another person "navigates" the driver through communication. People can switch between the two roles dynamically if they so choose. In my own experience, pair programming is excellent for tackling difficult problems that are simply too complex to keep contained in only one head. One person can focus on the details (usually the driver) while another can focus on the big picture (usually the navigator), and the end result is faster, less stressful development.

This, of course, is the ideal. This tends not to work well when two people of very different skill levels attempt it, and one of them is more anxious to "just get it done". In light of this, when selecting a partner for pair programming, it is best to select someone of approximately the same skill level.

On Collaboration:
In actual software development, very little work is done as an individual effort. Most modern software systems are simply too big to be undertaken in this style, so collaboration is key. This is one of the reasons why we are encouraging pair programming for some of the labs - this is a real, marketable skill.

That said, collaboration is **not** simply taking the credit for someone else's work. When this happens, **everyone** loses. The obvious loser is whoever did all the work. The less immediate loser is the one who took half of the credit for the work, because that person did not learn whatever the assignment was trying to teach. This can (and I have seen it happen during my own undergrad!) snowball long-term. Eventually the student who took the credit without working is truly completely lost, often without realizing it, and is held up entirely by the work of other students. This is a worst-case scenario, but in my experience it is far too common.

The point is this: collaboration means to work together, not to blindly take someone else's work or to give your own work away.

Attendance policy:

Attendance will not be taken in lecture, and there will be no graded assignments given during lecture. However, any material covered in lecture is ultimately **your** responsibility, regardless of whether or not the lecture was attended.

Due Dates / Late Policy:

For electronic submissions, all due dates are at midnight (technically 11:59 PM) on the same day. For pre-labs, the paper **must** be turned in during the lab (or lecture with the first lab), except in extenuating circumstances (explained below). If you cannot physically attend the lab, it is acceptable to have another student turn in your pre-lab for you (it must still be your work!).

For electronic submissions, as with labs and projects, each person has 24 hours worth of "grace" time in total. For example, if someone were to submit the first lab 4 hours late and the first project 6 hours late, then a total of 10 "grace" hours have been used. Both submissions would be accepted without incident, and there would be 14 "grace" hours remaining. For a pair submission, the grace time counts against both students individually. For example, if a pair submission were turned in 5 hours late, and one student had 20 hours of grace time remaining and the other 5 hours, then both students receive credit. The first student ends up with 15 grace hours remaining and the second is left without any grace time remaining. Except in extenuating circumstances, electronic submissions for students who have gone beyond their grace time will **not** be accepted.

A little background on this policy - the grace time is intended to be used as a sort of last-minute "oops" relating to a poor time estimate of (what should be) final touches. This policy tries to reduce the number of submissions hastily done just to meet a deadline, and to prevent issues of submissions that missed the deadline by a relatively small amount of time. It is not intended to be used as a way to extend the deadline for one assignment for a day, although it certainly can be used that way. Be forewarned: once it's gone it's gone, so use it wisely!

Extenuating Circumstances:

"Extenuating circumstances", for the purpose of this class, is defined as anything beyond our immediate control. In these cases, at my discretion I can grant an extension. To be absolutely clear, there is **no** guarantee that I will do so, and I am not obligated to grant them. For the things we can predict (i.e. trips), I expect to be contacted at least a week in advance. For the things we cannot predict (i.e. illness), I need official documentation explaining the situation (i.e. a doctor's note).

Communication Policy:

I have two office hours per week, though I may increase this to 3-4 if questions abound. If these hours do not work for you, then I am available by appointment as well (don't hesitate to ask!).

With email, assume that I will take at least 24 hours to respond. Typically my response time is much, much faster than this, but I do occasionally take this long. Historically, this has only been an issue the last hours before a project deadline, and only for students who started far too late. The point being: start early!

Academic Honesty:
In as few words as possible, cheating and plagiarism will **not** be tolerated. I understand that the temptation may be high ("it's just this one assignment" or "I just need this class"), but this is **no** excuse. At the very least, this is unfair to all the students who did not resort to such unethical means, who instead took the time and struggled through. I will be following UCSB's Academic Conduct policy on this (from http://www.sa.ucsb.edu/Regulations/student_conduct.aspx, under "General Standards of Conduct"), quoted below for convenience:

> It is expected that students attending the University of California understand and subscribe to the ideal of academic integrity, and are willing to bear individual responsibility for their work. Any work (written or otherwise) submitted to fulfill an academic requirement must represent a student's original work. Any act of academic dishonesty, such as cheating or plagiarism, will subject a person to University disciplinary action. Cheating includes, but is not limited to, looking at another student's examination, referring to unauthorized notes during an exam, providing answers, having another person take an exam for you, etc. Representing the words, ideas, or concepts of another person without appropriate attribution is plagiarism. Whenever another person's written work is utilized, whether it be a single phrase or longer, quotation marks must be used and sources cited. Paraphrasing another's work, i.e., borrowing the ideas or concepts and putting them into one's "own" words, must also be acknowledged. Although a person's state of mind and intention will be considered in determining the University response to an act of academic dishonesty, this in no way lessens the responsibility of the student.

Note that collaboration with a non-lab partner also constitutes cheating. Any incident of cheating **will** be reported. While this may sound steep, real-life cases of cheating (i.e. taking someone else's code without permission) have led to job termination and lawsuits among other things, so this is not unrealistic.

If you are absolutely stuck, and you have somehow acquired code that does what you need, **cite** this, be it a website or a student. It is unlikely that any credit will be awarded, but this is at least honest, and arguably more was learned this way than by giving up and submitting something blank.