

## . CSCI 202 Computer Science II Spring 2007

More information is in the Generic Syllabus

. See <http://csci.csusb.edu/dick/syllabus.html>

and on the World Wide Web (WWW) site for this class

. See <http://csci.csusb.edu/dick/cs202/>

You need to find the above sites for links to updates, grades, corrections, assignments, projects, labs, etc..

### . Description

Computer science is about predicting how pieces of software and hardware work together to solve people's problems. This is part of creating useful software. CS202 is the second class in the core of all the Computer Science majors. It covers the second half of the ACM CS1 and the first part of the ACM CS2. This class is required by other B.S. and BA degrees, and the Computer Systems Administration Certificate. This class is a prerequisite for CSCI320, 330, 372, 375.

### . Prerequisites

All tests, lectures, and work assume you have passed a class equivalent to our CSCI201. You need to have a Satisfactory ELM. Previous lab experience of UNIX is very helpful.

### . Goals

You will develop problem solving skills used in Object-Oriented programming and software engineering. You will use a languages, operating system, and libraries that (1) are popular in industry and (2) force you to think about what you are doing. You will learn about new control structures and new types of data. The course includes the Unified Modelling Language, multi-file programming, data abstraction, encapsulation, templates, inheritance, polymorphism, and the standard library.

### . Reading

**Required Text:** Skansholm, C++ from Beginning, ISBN 0-201-72168-6. In bookstore . We will cover every chapter and appendix but we will take the first 4 very quickly since they overlap with CSci201. Start by reviewing chapters 1 thru 4.

**REFERENCE BOOKS:** My WWW site has many useful pages including the draft specification of the **C++ language**. You may benefit from referring to the CS201 book (Horstman) if Skansholm is confusing on a topic. C++ books in the library are in the QA76.73.C153 section. A former CS202 student recommends "Conquering C++ pointers" by Robert J. Traister, ISBN 0-12-697420-9.

There will be handouts on the **UML** but Fowler's "UML Distilled" is an excellent reference in practice.

If you haven't used **Linux** much then Daniel J Barrett, "Linux pocket Guide," O'Reilly Feb 2004 \$9.95 ISBN 0-596-00628-4 is an inexpensive and useful reference. Also go to the Computer Science Club's Linux sessions. For more try "Linux in a Nutshell" and "Learning the VI Editor" both from O'Reilly & Associates, Inc.

### . Work

You will study the text. You will do much supervised and independent programming. You will do independent offline work (mainly reading and thinking) as well. Some of this work is submitted for grading. Note. I can't tell you how much work the project work will take: good programmers produce code 5 times faster than bad ones!

### . Meetings (20pts=4%, 1 pt per session)

Attend all class meetings from beginning to end or make up the work yourself. Exception - medical and other documented emergencies. \*Note: *mobile phones and wireless devices* -- Turn off audio alarms. Do not use wireless devices during class time.

### . Assigned Work (20pts=4%, 1 pt per session)

Read the part of the text assigned in the schedule and write one question on it on a piece of paper. Print your name in the top right hand corner. Hand in at the start of each class except the first.

### . Quizzes(100pts=20%, 120 pts Max)

There will be four quizzes each worth about 30 points each. Points above 100 will be used to make up points lost elsewhere. See the schedule for dates and topics.

### . Labs(100pts, 20%, 10pts per lab)

You can only attend the lab that you have enrolled in. The instructions are on the WWW site. I may change them until just before the class just before the lab. I will give you grades/scores for each lab by the **end of the lab**. Scores vary from full credit(A=10 points) down to zero. Seven (7) points (a D) will be given for being present, working hard, and making demonstrable progress during the whole lab period.

### . Comprehensive Final(200pts, 40%)

The final exam will be on June 13th from 2pm to 3:50pm. It will go over material covered in quizzes, labs, projects, lecture/discussions, assigned work, and notes.

### . Tests

Tests will be closed-book and supervised. **No wireless communication! You should prepare a 11.5in<8in sheet of notes to use in each test.** You may use a calculator, but not a computer. A computer is any device that lets you compile and run C++ programs or browse the web. The quizzes and final will require you to (1) read, interpret, correct, and complete pieces of C++ programs, and (2) write correct C++ code.

### . Project Work(100pts, 20%, 20 points max per project)

You will be writing five(5) programs (see the schedule below and on the web site) based on exercises in the book. The programs must be your own work and unlike anybody else's.

More details will be put on the WWW site.

It is better to give me something on time than give me a perfect project late. Earn a **bonus** of 5pts (1%) by handing in a complete project at the start of the class before the deadline. For example, if a program is due Wednesday then it must be handed in before class on the previous Monday to get a bonus. A bonus can make up for points lost elsewhere but not in the final..

**Re-submission.** You can resubmit projects P1, P2, P3, and P4 **once** as long as the original scored a **B** or less. Any bonuses will be lost if a project is resubmitted. High light the changes you have made. Resubmit on the next due date -- See Project Schedule. Note: you can hand in a new and a resubmitted project at one time. The last project P5 may not be resubmitted.

### Start Early!

Aim to complete each project before the deadline because **things go wrong**. First think out a solution and roughly sketch it on paper. **Start coding the program** with a comment that has your name plus the number of the exercise. Add a short description of what the program demonstrates or does. Add the necessary "#include"s and an empty "main" function and make it compile. Add a set of tests of the classes and functions you need -- and compile it and watch it fail. Then add missing stuff or fix broken parts until all the tests pass. Then tidy up.

### CS202 Project Requirements

In a *good software company*, your bosses and colleagues want your work to be (1) on time, (2) correct, and (3) understandable. Late or confusing work is penalized. So, if the work is late it gets no points and if I can't understand it, then it loses points. Good projects also meet *academic standards*: (1) They show the skills and knowledge covered in the course at the time. (2) They must not pirate other people's work. **Warning: I'm good at spotting errors and borrowed work.** I have tools that can find anything that you download from the web.

**Work must be on time.** Partial credit is given for incomplete work including programs with *documented* errors. You won't lose points for missing features or bugs, *if you document them in the code*. (2) The code must *explain what it does and how it does it*. (3) All copied code must be documented in comments.

**What to hand in:** I want to read your code and know that it will run. Print it out, staple it and hand it in! **No** cover sheets, folders, etc. **No** test runs. Each file should have a comment with your name and a description of the project as a comment at the top. It should have more comments explaining anything complicated or buggy. Later projects may need an UML class diagram.

The schedule lists a chapter. Pick one exercise from the end. All earn the same score. However, I expect a simple project to be done better than a more complex one.

### Project Schedule

#	Due on	One exercise from end of
P1	April 11	Chapter 5 but not exercise 5.18
P2	April 18	Chapter 7
P3	May 2	Chapter 9
P4	May 16	Chapter 11
P5	June 4	Chapter 13, exercises 13.1 to 13.11 only

### Hints

(0) **Always Seek Knowledge (ASK):** Come to see me with questions about what is needed early.

(1) **Do the simplest thing that could possibly work.**

(2) **If in doubt, take a small step** in a useful direction. **If stuck, do something different**, and come back later. If you are coding, draw a picture. If drawing, try code. **Visit the Comp Sci Club** <http://club.csci.csusb.edu/forums/>. **Take a break:** go shopping, eat something, get some exercise, come to my office, etc.. **Read & Think:** WWW, the book, the Pfau library, the Internet, ... . **Find a tutor** in the learning center. **Talk to a student** who is doing a different project.

(3) Unexpected bugs? **Bring printouts of code** to my office. Talk to a lab assistant or any CSci. teacher.

(4) **Spare time?** Refactor the code -- reorganize and retest, one simple safe step at a time, so that it is easier to understand. Use the DRY="Don't Repeat Yourself" rule to make code better.

(5) **Stop at the deadline. Hand in what you've got.**

### . How I will grade projects

I read the program from beginning to end and then assign a letter grade and/or a score. As I read it I will mark up problems. An **A**(100%, 20 points) program identifies who did it what it does. It is clear that it does what it says it does. There are no spelling mistakes or grammatical/syntax errors. It shows that you have understood the material in the course up to that time. Any bugs are described in the code. A **B**(90%, 18 points) is like an **A** program but is not as clear and easy to understand. A **C**(80%) program still identifies who did it what it does. But it may not do what you think it does(bug) or may have other mistakes. A **D**(70%) program has undocumented errors, spelling mistakes, grammatical goofs, syntax errors, etc.. It shows that you didn't understand the material in the course up to that time. An **E**(60%) program is worse than a **D** but is still an honest attempt. I may have to use unlettered percentages for really bad work. An **F**(0%) program is either late or has some plagiarized material in it. I may use grades like "A-"(19 points) for odd points and intermediate quality work.