CS 210 – Fundamentals of Programming I  
Fall 2006 – Syllabus

Instructor  
Dr. Deborah Hwang  
KC-264, 488-2193, hwang@evansville.edu  
Home page: http://csserver.evansville.edu/~hwang

Office Hours: See instructor's home page.

Course Home Page  
Announcements regarding handouts and assignments will be made in class. Assignments will be available only at the course home page (http://csserver.evansville.edu/~hwang/f06-courses/cs210.html). It is your responsibility to consult the course home page on a regular basis. Grades will be posted to Blackboard (http://acebb.evansville.edu).

Catalog Data  
Emphasizes problem solving techniques used in the analysis and design of software solutions including structured top-down design, abstraction, good programming style, debugging, and testing. Programming constructs covered include control structures, functions, and basic, aggregate, and user-defined data types. Introduction to object-oriented design and recursion.

Objectives  
Learning problem solving techniques used in programming software solutions including structured design, good programming style, testing strategies, and debugging strategies. Introduction to abstract data types and object-oriented design. Note that you will be learning and using C++ as a vehicle towards these goals, but this course is not about learning C++ in its entirety.

Prerequisites: None.

Required Textbook  

Daily Requirements  
Assigned daily reading assignments. Daily in-class exercises on material up to and including the reading assigned for the day. These exercises may be both written and programming exercises, and may be individual or group exercises.
**Homework, Programming Assignments, and Projects**

There will be weekly written homework exercises and short programming assignments that will focus on a single key concept. Generally these will be assigned on Thursday and due the following Thursday. Each programming assignment will consist of a written analysis and design and a C++ implementation.

In addition, there will be 2 larger programming projects of 2–3 weeks in duration that will combine key concepts. For each project, a written analysis and design of the program will be due one week after it is assigned, followed by an implementation of the program in C++ due 1–2 weeks later.

**Exams and Evaluation**

There will be two written in-class exams during the term, a 2-hour practical lab midterm exam, and a comprehensive written final exam. The in-class exams are tentatively scheduled for September 21 and November 7. The practical exam is tentatively scheduled for October 3. Since there are more students than computers in KC-267 there will be exam sessions at 1:30-3:30pm (normal class time) and 3:30-5:30pm. The final exam is Friday, December 8, at 2:45pm.

Final grades will be based on the following weighted distribution:

- 20% Two in-class written exams (10% each)
- 10% 2-hour practical exam
- 20% Comprehensive final exam
- 5% Written homework exercises
- 10% In-class exercises
- 15% Weekly programming assignments
- 20% Two programming projects (10% each)

Final grades are based on the final weighted percentage with adjustments depending on class distribution. Historically, the A/B line falls around $88\% \pm 2\%$ with subsequent grade levels every 10%.

**Grading of Short Programming Assignments and Programming Projects**

Short programing assignments (in general) and programming projects will be divided into two parts:

- 30% Analysis and design
- 70% Implementation

For short programming assignments, the analysis and design and the implementation are assigned and due together. For programming projects, the analysis and design will be assigned separately at least one week before the implementation is assigned. The criteria used in grading analysis and design include appropriate structure and modularity for the problem being solved. See handout *An Analysis and Design*.  

08/18/2006

2
Guideline for the appropriate format used in this course.

The implementations of short programming assignments are due at the same time as the analysis and design. The implementations of projects will be due 1-2 weeks after the analysis and design. They will be graded using the following criteria with the weights as shown.

- 60% Correct results
- 10% Appropriate error checking
- 30% Style, observed coding guidelines, originality

See handout A C++ Programming Style Guideline for details of the coding style guideline used in this course. This style is substantially the same as the one in the textbook.

Missed Classes and In-class Exercises
Due to the integrated lecture and lab format of the course, attendance is mandatory. In-class exercises may or may not be made up depending on the nature of the exercise. However, excused absences will be noted and taken into consideration when assigning final grades.

Late Homework, Late Programming Assignments, Late Projects
All other assignments (homework, programming assignments, programming projects) are due at the instructor's office and/or electronically as appropriate by 4:30pm on the date specified unless otherwise noted. Any assignments arriving after 4:30pm are considered late. The following automatic late penalties will be applied:

- 10% if handed in by 4:30pm, one day late
- 20% if handed in by 4:30pm, two days late
- 30% if handed in by 4:30pm, three days late

Unexcused late work will not be accepted for credit after three days after the due date without prior arrangements. For the purpose of counting days, Friday 4:30pm to Monday 4:30pm is considered one day. Please note that the purpose of the automatic late extension is to allow students leeway when needed. It is usually better to hand in something late and completed than on-time and incorrect. However, chronically handing in late submissions will lower your final grade.

Valid excuses for missing exams, missing classes, and handing assignments in late include illness, family emergencies, religious observances, official UE events such as varsity games and concerts, etc. They do not include (most) work conflicts, studying for other classes, leaving a day early or staying home an extra day over a weekend or holiday, etc. In general, an excused absence is one caused by circumstances beyond your control.
The instructor will rely on your integrity for getting work excused. If you have a valid excuse, put it in
writing, sign your name to it, and give it to the instructor. For religious observances and official UE
events, you must inform the instructor that you will be absent **before** the absence occurs, otherwise it will
be considered an unexcused absence.

Excused work must be made up within one calendar week from the original due date for full credit. Late
excused work will not be accepted. Exceptions will be made for serious or prolonged illness, or other
serious problems. **Please note:** It is your responsibility to take care of missed or late work.

**Attendance Policy**
Attendance is important and expected. Attendance records will be maintained in accordance with Federal
Law, but will not be used in the determination of grades, except to the extent it affects the in-class
exercise portion of your grade and in borderline cases. Students are responsible for all material covered
in class. If you miss a class, find out what was covered from another student. You are responsible for
checking the course home page for new assignments even if you miss class.

**Honor Code**
All students are expected to adhere to the University's Honor Code regarding receiving and giving
assistance. Two specific guidelines are in force for this course.

- Written homework exercises are for you to gain experience and practice. You may collaborate
  with your classmates, but each student should submit a solution in his/her own words that reflect
  his/her understanding of the solution. Ultimately you will be required to demonstrate your
  proficiency of the material on exams. Therefore, it is highly recommended that you attempt all
  homework problems on your own before finding a solution from another source.

- Programming assignments and programming projects (including analysis and design) are to be
  your own work unless otherwise noted. Discussing the meaning and general solution techniques
  of an assignment with other students is permitted. For example, discussing “How is this
  assignment similar or different from problems presented in the text or in lecture?” is acceptable.

  Asking another person for assistance on **specific** items in your own analysis and design or code is
  also permitted, but you may not observe another person’s solution or code in its entirety for the
  purposes of studying or copying it, with or without that student's permission. Examples of
  acceptable assistance include:

  - “What's the design format for counting loops again? I want a loop that has a counter that
    counts from 1 to 100”

  “The format is ‘For counter from 1 to 100 by 1 do …’
“What's the C++ code for counting loops again? I want a loop that has a counter that counts from 1 to 100”

“The code is ‘for (int counter = 1; counter <= 100; counter++)...’”

“The compiler is complaining about a missing semicolon. I can’t see where I’m missing one. Could you look at my code and see if you can find where it is?”

“Here is it. Remember, it’s usually the line above the error that’s missing the semicolon.”

“My function doesn’t compute the correct answer. It seems to be going through the loop one too many times. Could you look at this?”

“Your loop condition is not quite correct; it should be i < n, not i <= n. Remember, array indexes start at 0 and end at n - 1.

Note in the last case, the assistance is acceptable because the requester has asked about something specific. Examples of unacceptable assistance include:

“How did you write the Sort function?”

“I passed an array and a size, and wrote nested for loops...”

“My function doesn’t work...”

“Here’s what I did...”

“Can I look at your analysis and design (or program) and see what you did?”

“Sure...”

“Hey, here’s Jane Student’s CS 210 program listing. Let’s look at her program...”

“Hey, Joe Student left himself logged into this machine. Let’s look at his program...”

Note in the last two cases that just looking at another student’s code, even if he/she left it unprotected, is not permitted.

If there is any doubt as to whether assistance is acceptable, consult the instructor.