CS 215 - Fundamentals of Programming II  
Fall 2008 - 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/f08-courses/cs215.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  
Project and problem-solving course emphasizing the use of classes for encapsulation of abstract data types and abstract data structures. Topics include classes, templates, dynamic allocation, searching and sorting, recursion, and exception handling.

Objectives  
To continue the study of problem solving techniques used in programming software solutions including abstract data types, templates, exception handling, dynamic allocation, recursion, sorting and searching algorithms, and basic algorithm analysis. To continue the study of the C++ language features that support these techniques. Exposure to the UNIX operating system and development tools.

Prerequisites: CS 210

Required Textbook  

Recommended Textbook  
Daily and Weekly Requirements
Assigned daily reading assignments. Weekly homework assignments. Homework assignments may include short programming problems.

Programming Projects
There will be 7-9 programming projects of 1-2 weeks in duration each. See handout *A C++ Programming Guideline for CS 215* for appropriate code format used in this course. It is similar, but not the same, as the style used in the textbook.

Programming projects will be graded using the following criteria with the weights as shown:

- **65%** Correct results, including command line arguments and file I/O
- **10%** Error checking, including proper use of exceptions
- **25%** Style, observed coding guidelines, originality

In addition, there will be an up to 3-point penalty if an appropriate Makefile is not submitted when required.

Programming projects should be submitted both on-line by email and in hard-copy printout as explained in the handout *Submission Instructions for CS 215*.

Exams and Evaluation
There will be two in-class written exams and a comprehensive final written exam. In addition, there is an evening lab programming practical exam that is scheduled for the week after midterm. The purpose of the practical exam is to demonstrate mastery in using the C++ programming language and the UNIX environment. Therefore, it is necessary to score a minimum of 60% on the practical exam to pass the course (grade of C- or better). Students who fail to do so and are otherwise passing the course will be given a second opportunity to pass the practical exam at the end of the term with a 10% penalty. Final grades will be based on the following weighted distribution:

- **20%** Comprehensive final exam
- **20%** Two in-class exams (10% each)
- **15%** Lab programming practical exam
- **5%** Homework (weighted as indicated in assignment)
- **40%** Programming projects (weighted as indicated in assignment)

Final grades are based on the final weighted percentage with some adjustments depending on class distribution. Historically, the A/B line falls around 88% +/- 2% with subsequent grade levels every 10%.
**Late Projects, Late Homework**

Homework and programing 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 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 in borderline cases. However, the instructor reserves the right to reduce a final grade in this course for excessive absences. Students will be warned prior to such action. 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. The following specific guidelines are in force for this course.

- **Homework exercises** (including programming 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. This includes the programming exercises, which should be the result of your own typing. 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 projects 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. For example, asking, "What does this compiler error mean?" or "Do I have the correct class syntax here?" is acceptable. Whereas asking "Can I see how you coded your stack?" is not acceptable.

  In particular, since UNIX systems tend to be open by default, it is absolutely forbidden to "rummage" around the cserver file system looking at anyone else's work even if they have not set the file permissions to prevent such observation. (For those that would rather not rely on the integrity of others, it is suggested that all work for this class be put into a subdirectory that has its permissions set to owner only.)

  Giving or receiving unauthorized aid on a programming project will result in a 0 for the project on the first offense. Any subsequent violations will result in an F for the course and possibly formal disciplinary action.

- Exams, of course, are to be solely your own work. Giving or receiving any type of unauthorized aid on any exam will result in a final grade of F and possibly formal disciplinary action.

If there is any doubt as to whether assistance is acceptable, consult the instructor.
Course Schedule
Here is a tentative schedule showing the daily reading assignments and exams for this term. Adjustments will be made as needed.

<table>
<thead>
<tr>
<th>Week of</th>
<th>Monday</th>
<th>Tues</th>
<th>Wednesday</th>
<th>Thurs</th>
<th>Friday</th>
</tr>
</thead>
<tbody>
<tr>
<td>08/25</td>
<td></td>
<td></td>
<td>Introduction UNIX, g++, make</td>
<td></td>
<td>C string tutorial argv, argv</td>
</tr>
<tr>
<td>09/01</td>
<td>File streams</td>
<td></td>
<td>Chapter 1.1-1.5 ADTs &amp; Classes</td>
<td>Chapter 1.7-1.8 C++ strings</td>
<td></td>
</tr>
<tr>
<td>09/08</td>
<td>Chapter 2.1-2.3 Exception handling String streams</td>
<td></td>
<td>Chapter 2.4 Overloading</td>
<td>Chapter 3.1-3.2 Selection sort Search</td>
<td></td>
</tr>
<tr>
<td>09/15</td>
<td>Chapter 3.3-3.4 Algorithm analysis</td>
<td></td>
<td>Chapter 3.6 Recursion</td>
<td>Chapter 3.7 Recursion</td>
<td></td>
</tr>
<tr>
<td>09/22</td>
<td>Chapter 3.5, 4.1-4.3 Templates, vectors</td>
<td></td>
<td>Exam 1 Review</td>
<td>Exam 1 Material to 9/19</td>
<td></td>
</tr>
<tr>
<td>09/29</td>
<td>Chapter 4.3-4.4 Using vectors Insertion sort</td>
<td></td>
<td>Chapter 5.1-5.2 Pointers dynamic arrays</td>
<td>Multi-dimensional dynamic arrays</td>
<td></td>
</tr>
<tr>
<td>10/06</td>
<td>Chapter 5.3-5.4 Dynamic classes</td>
<td></td>
<td>Chapter 5.5-5.6 Dynamic classes</td>
<td>Debugging, gdb</td>
<td></td>
</tr>
<tr>
<td>10/13</td>
<td>Fall Break No class</td>
<td></td>
<td>Chapter 6.1-6.2 List ADT, Iterators</td>
<td>Chapter 6.3-6.4 List operations</td>
<td></td>
</tr>
<tr>
<td>10/20</td>
<td>Chapter 7.1-7.5 Stack ADT</td>
<td></td>
<td>Practical Exam 2-HR TBA</td>
<td>Chapter 7.4-7.5 Chapter 8.1-8.3 Stacks, Queue ADT</td>
<td></td>
</tr>
<tr>
<td>10/27</td>
<td>Chapter 8.4-8.5 Queues</td>
<td></td>
<td>Chapter 9.1-9.3 Linked Lists</td>
<td>Chapter 9.4 Using linked lists</td>
<td></td>
</tr>
<tr>
<td>11/03</td>
<td>Chapter 9.5-9.6 Doubly-linked lists</td>
<td></td>
<td>Exam 2 Review</td>
<td>Exam 2 New material to 10/31</td>
<td></td>
</tr>
<tr>
<td>11/10</td>
<td>Chapter 10.1-10.3 Binary trees</td>
<td></td>
<td>Chapter 10.3-10.4 Traversing binary trees</td>
<td>Chapter 10.5 Binary search trees BST ADT</td>
<td></td>
</tr>
<tr>
<td>Week of</td>
<td>Monday</td>
<td>Tues</td>
<td>Wednesday</td>
<td>Thurs</td>
<td>Friday</td>
</tr>
<tr>
<td>--------</td>
<td>---------------------------------------------</td>
<td>------</td>
<td>------------------------------------------</td>
<td>--------------------------------------------</td>
<td>---------------------------------------</td>
</tr>
<tr>
<td>11/17</td>
<td>Chapter 10.6-10.7 Using BSTs</td>
<td></td>
<td>Chapter 14.1-14.2 Heaps, heapsort</td>
<td>Chapter 8.6, 14.3 Priority Queue ADT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>BST class</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11/24</td>
<td>Chapter 15.1 Quicksort</td>
<td></td>
<td>Thanksgiving Break</td>
<td>Thanksgiving Break</td>
<td>Thanksgiving Break</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>No class</td>
<td>No class</td>
<td>No class</td>
</tr>
<tr>
<td>12/01</td>
<td>Chapter 15.1 Comparing sorting algorithms</td>
<td></td>
<td>Chapter 11.1-11.2 Associative containers</td>
<td>Chapter 11.3 Maps</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Sets</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12/08</td>
<td>Course review</td>
<td></td>
<td>Reading/Study Day</td>
<td>Final Exam 8:00am</td>
<td></td>
</tr>
</tbody>
</table>