CS 215 - Fundamentals of Programming II
Spring 2017 - 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 will be made in class. However, most handouts will be available only at the course home page (http://csserver.evansville.edu/~hwang/s17-courses/cs215.html). It is your responsibility to consult the course home page on a regular basis. Grades will be posted to Blackboard (http://bblearn.evansville.edu/).

Catalog Description

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 and Outcomes

The main objective of this course is to continue the study of problem solving techniques used in programming software solutions with emphasis on abstract data types, and to become proficient in the use of the UNIX operating system and development tools. Specific outcomes for this course include:

- Students will be able to write programs using selection and repetition constructs, functions, and arrays.
- Students will be able to use dynamic allocation and recursion to solve problems.
- Students will be able to use design and implement abstract data types (ADTs) using simpler data structures such as multi-dimensional arrays and vectors, and use these ADTs to solve problems.
- Students will be able to design and implement abstract data types (ADTs) using linked data structures such as linked lists and binary trees, and use these ADTs to solve problems.
- Students will be able to implement several sorting algorithms.
- Students will be introduced to rudimentary algorithm analysis.
- Students will be able to do basic generic programming and use library structures such as stacks and queues to solve problems.
- Students will complete at least 4 assignments requiring design of an application using one or more data structures, testing, and debugging.
- Students will be proficient using the UNIX operating system.
- Students will be proficient using a UNIX editor, such as emacs or vim.
- Students will be proficient using relevant GNU tools for software development, including g++ (compiler) and make (program build utility).
- Students will be introduced to contemporary professional issues.

Prerequisites: Grade of C- or better in CS 210

Required Textbook
Dale, Weems, & Richards, *C++ Plus Data Structures, 6/e*, Jones & Barlett Learning, 2018, ISBN: 978-1-284-08918-9

Daily Requirements
Assigned daily reading assignments. Weekly homework assignments. Homework assignments may include both written exercises and short programming problems.

Programming Projects
There will be 7-8 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.

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, makefile

Programming projects must be submitted electronically as explained in the handout *Submission Instructions for CS 215*.

Exams and Evaluation
There will be one 2-hour written mid-term exam and a comprehensive 2-hour final written exam. In addition, there are two 2-hour practical programming exams. The purpose of the practical programming exams is to demonstrate mastery in using the C++ programming language and the UNIX environment. T**herefore, it is necessary to score a minimum of 70% on the second practical programming 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 second practical programming exam at the end of the term with a 10% penalty. Final grades will be based on the following weighted distribution:

20%	2-hour comprehensive written final exam
15%	2-hour written midterm exam
20%	Two 2-hour practical exams (5% and 15%)
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%.

Missed Exams, Late Homework, Late Projects

Written homework is due at the instructor's office by 4:30pm on the date specified unless otherwise noted. Any such assignments arriving after 4:30pm are considered late. Programming homework and projects must be submitted electronically by 11:59pm on the date specified. The following automatic late penalties will be applied:

10%	if handed in or submitted one day late
20%	if handed in or submitted two days late
30%	if handed in or submitted 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 to Monday 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, **send an email to the instructor with the details**. 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. In cases of excessive absences, the instructor reserves the right to require official documentation

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 unexcused 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.

Credit Hour Policy

This course meets the federal requirements of 15 in-class hours plus an expected 30 hours of out-of-class work per credit hour.

Disability Policy

It is the policy and practice of the University of Evansville to make reasonable accommodations for students with properly documented disabilities. Students should contact the Office of Counseling and Health Education at 812-488-2663 to seek services or accommodations for disabilities. Written notification to the instructor from the Office of Counseling and Health Education is required for academic accommodations.

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** (including programming 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 are to 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 also is 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. This includes, but is not limited to, other students in the class, previous students in the class, and the Internet. 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 csserver 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.) Also, studying printouts left in CS Lab (or any other lab) is not acceptable.

  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. Also note that such incidences are recorded by the EECS department and multiple violations across courses also may result in 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 the first few weeks of the term. A more complete schedule will be forthcoming.

Week of	***Monday***	***Wednesday***	***Friday***
01/09	Overview, basic Unix, `clang++`, console I/O	C++ types, namespaces, file streams, `argv`. `argc`	Output formatting, reference parameters
01/16	MLK, Jr. Day No class	Chapter 7.1-7.6 Recursive functions	Chapter 7.8, 7.13-7.15: Recursive functions
01/23	Chapter 7 Case Study: Recursive maze searching	Chapter 1, 2.1-2.3: Abstraction, classes	Separate compilation, `make`
01/30	Chapter 2.5: Assertions, exceptions	Testing, debugging, `gdb`	Chapter 2.6: Algorithm analysis, Big-O notation

Tentative Exam dates (exams are 2 hours; times TBA):
2/15 – Practical Exam I
2/27 – Midterm Exam Review
2/29 – Midterm Exam
3/29 – Practical Exam II

Final exam period for this class is Monday, May 1, 11:00am-1:00pm