CS 210 - Fundamentals of Programming I
Fall 2019 - Syllabus

Instructor

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

Office Hours: See instructor's home page.

Class Hours:

TTh, 9:00am - 10:50am, KC-267

Course Home Page

Announcements regarding handouts and assignments will be made in class. Assignments will be available only at the course home page (https://csserver.evansville.edu/~hwang/f19-courses/cs210.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 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 and aggregate data types. Introduction to recursion and dynamic allocation.

Objectives

Learning problem solving techniques used in programming software solutions including structured design, good programming style, testing strategies, and debugging strategies. Note that you will be learning and using the C language as a vehicle towards these goals, but this course is not about learning C in its entirety. More specifically:

- Students will be able to write programs using selection, repetition, functions, and dynamic allocation in one major high level language.
- Students will be able to write programs using the aggregate data types arrays, strings, and structs.
- Students will be introduced to the fundamentals of functional decomposition and design.
- Students will be introduced to testing and debugging strategies.
- Students will be able to construct software solutions that use structured analysis and design with good programming style.
- Students will be proficient using a Microsoft Windows IDE, such as Visual Studio or Code Blocks, to implement and debug programs.
- Students will be introduced to contemporary professional issues

Prerequisites: None.

Required Textbook

Jeri Hanly and Elliot Koffman, *Problem Solving and Program Design in C, 8/e*, Pearson, 2016. ISBN 978-0-13-401489-0

Daily Requirements

You are expected to read the assigned chapter(s) from the textbook. There is not enough time during lecture to cover all the material in the depth necessary for mastery. You are responsible for all of the material in assigned chapters even if it is not explicitly covered in lecture, unless otherwise noted.

Most classes will have in-class exercises on material up to and including the reading assigned for the day. These exercises may be written and/or programming exercises, and may be individual or group exercises. Generally, these exercises are due by the end of the class period.

Attendance Policy - Missed Classes and In-class Exercises

Due to the integrated lecture and lab format of the course, **attendance is mandatory**. Graded in-class exercises may not be made up, regardless of reason for an absence. However, excused absences will be noted and taken into consideration when assigning final grades. 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.

In addition, many classes will end with more than adequate time to complete any in-class exercises. The rest of the time is for you to work on the currently assigned programming project. **You are expected to use the time to the end of the scheduled class period to work on the assignment.** The only valid reason for leaving early is if you have already completed the outstanding assignment. During this time the instructor is available to answer your questions regarding the program and its design. This is incorporated into the design of this course and if you do not take advantage of this, several key concepts may be missed (and your grade will probably end up reflecting this).

Homework and Programming Projects

There will be weekly written homework exercises. Generally homework exercises will be assigned on Thursday and due the following Tuesday. There will be programming projects of varying lengths and difficulty. Generally, they will be assigned on Thursday and due the following Thursday or the Tuesday after that.

Exams and Evaluation

There will be one in-class **written** midterm exam, two practical **programming** exams, and a comprehensive **written** final exam. The purpose of the practical **programming** exams is to demonstrate mastery in using the C programming language sufficient to continue on to CS 215. **Therefore, it is necessary to score a minimum of 70% on the second practical programming exam to pass the course (grade of C- or better) regardless of performance on other assignments.** Students who fail to do so and are otherwise passing the course will be allowed an opportunity to substitute a makeup practical programming exam at the end of the term with a 10% penalty.

Final grades will be based on the following weighted distribution with some adjustments depending on class distribution. Historically, the A/B line falls around 88% +/- 2% with subsequent grade levels every 10%.

20% Two in-class practical exams (5% and 15%, respectively)

15% Written midterm exam

20% Comprehensive written final exam

10% In-class exercises and homework (weighted as indicated)

35% Programming projects (weighted as indicated)

Graded work scores will be posted to BlackBoard. The total weighted score is the current weighted percentage of the currently graded work based on the assignments for the **entire** course. Thus as the course proceeds and more work is graded, the maximum weight percentage available increases until it is 100% after the final exam. Periodically, the instructor will announce the current maximum weighted percentage available.

Grading of Programming Projects

Programing projects grading will be divided as follows:

75% Implementation - correct results, appropriate error checking

15% Analysis and design - appropriate structure and modularity as specified by the assignment and indicated by appropriate commenting

10% Coding format guidelines

The analysis is be included as comment as part of variable and parameter declarations. The design steps are indicated by numbered in-lined comments. Programs also are required to be formatted and commented as described in handout *A C Programming Style Guideline*. This style is substantially the same as the one in the textbook.

All programs will be submitted electronically for testing using the submission system at https://submission.evansville.edu. Programs may be submitted multiple times with the final submission being the graded submission. **Ignore the score!** The results of the submission system testing is only part of an assignment's grade. Also, the late penalties are computed differently than given below. (Dr. Roberts wrote the submission system, so it uses his late penalty scheme, which is different than that given below.)

Missed Exams, Late Homework, Late Projects

In-class exercise written sheets must be submitted at the end of the class period. **In-class exercise programs** must be submitted electronically by 11:59pm on the day assigned. As noted previously, late in-class exercises will not be accepted.

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

10% if handed in one day late

20% if handed in two days late

30% if handed in 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. (For example, an

assignment due on Thursday may be submitted for late credit until the following Tuesday.) Please note that the purpose of the automatic late extension is to allow students leeway when needed. It is usually better to submit something late and completed than on-time and incorrect. However, chronically submitting late work will lower your final grade.

Valid excuses for missing exams, missing classes, and submitting late assignment 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.** However, the instructor reserves the right to request written documentation in cases of chronic absences. 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.

Credit Hour Policy

This course meets the federal requirements of 45-75 total hours of student work (combined classroom plus 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 unauthorized assistance. Specific guidelines in force for this course are:

- **Written homework and in-class 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 and in-class problems on your own before finding a solution from another source.

- **Programming assignments (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 y**ou 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:

- "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."

- "What's the design step for loops again? I want a loop that accesses each element of an array." "The format is 'For each element in the items array ...' "

- "What's the C code for array accessing loops again? I want a loop that accesses each element." "The code is '`for (index = 0; index < n; index++)...`' "

- "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..."

- "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.

- Solutions shall not be copied from the Internet or any other sources including previous students' work. Discovery of such will result in a 0 for that assignment for all parties involved for the first offense. A second offense will result in failure of the course.

- **Any** attempt to compromise the automatic testing system will result in immediate failure of the course.

- **Exams and quizzes, 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.

Also note that violations of the Honor Code are recorded by the EECS department and multiple violations across courses also may result in formal disciplinary action. If there is any doubt as to whether assistance is acceptable, consult the instructor.

Reading Schedule

This is a tentative schedule of topics for this course. You are expected to have read the assigned material before coming to class.

Week of	*Mon*	*Tuesday*	*Wed*	*Thursday*	*Fri*
08/19				Chapter 1 & 2: Introduction, Overview of C	
08/26		Chapter 2 Overview of C		Chapter 3: Functions	
09/02	Labor Day	Chapter 3: Functions		Chapter 4: Selection	
09/09		Chapter 5: Repetition, aka Loops		Chapter 5: Repetition	
09/16		Chapter 5: Repetition		Chapter 6: Pointers, Reference Parameters	
09/23		**Practical Exam 1**		Chapter 11: File pointers, textfiles	
09/30		TBD		Written Midterm Exam Review	Out of town
10/07		FALL BREAK NO CLASS		**Written Midterm Exam**	
10/14		Chapter 7: Arrays		Chapter 7: Arrays	
10/21		Chapter 8: Strings		Chapter 8: Strings	
10/28		**Practical Exam 2**		Chapter 7.8: Multi-dimensional Arrays	
11/04		Chapter 10: Structs		Chapter 12: Separate Compilation	**Last day to drop**
11/11		Chapter 13: Dynamic Allocation		Chapter 13: Dynamic Arrays	
11/18		Chapter 9: Recursion		Chapter 9: Recursion	
11/25		TBD		THANKSGIVING BREAK NO CLASS	
12/02		**Final Exam Review**		READING/STUDY DAY	
12/09					