EECE.2160: ECE Application Programming

Summer 2016

Syllabus

Course Meetings

Section 011: MWTh 8-10:20, Kitson 302

Course Website

Main page: http://mjgeiger.github.io/eece2160/sum16/

Schedule: http://mjgeiger.github.io/eece2160/sum16/schedule.htm

Course Discussion Group

All course announcements will be posted on the discussion group—you are responsible for checking the board regularly or enabling direct e-mail updates from Piazza.

Sign up link: http://piazza.com/uml/summer2016/eece2160

Instructors

Dr. Michael Geiger
E-mail: Michael_Geiger@uml.edu
Office: Perry Hall 118A
Phone: 978-934-3618 (x43618 on campus)
Office hours: TBD

During my office hours, student questions are my top priority. I am available by appointment at other times.

Feel free to stop by my office, e-mail me questions, or schedule a one-on-one appointment. Office hours are subject to change.

Textbook

K.N. King, *C Programming: A Modern Approach*, 2nd edition, 2008, W.W. Norton.
ISBN: 978-0-393-97950-3

Course Overview

Catalog Description: Introduces C programming for engineers. Covers fundamentals of procedural programming with applications in electrical and computer engineering and embedded systems. Topics include variables, expressions and statements, console input/output, modularization and functions, arrays, pointers and strings, algorithms, structures, and file input/output. Introduces working with C at the bit manipulation level. Laboratories include designing and programming engineering applications. 3 credits.

Course Objectives: By the end of this course, you should understand and be able to use all of the following:

1. **Basic C Language Concepts:** constants, variables, operators, expressions and assignment statements
2. **Input and Output:** Reading data from the keyboard and displaying formatted results on the screen
3. **Flow of Control 1 – Decisions and selection:** `if` and `switch` statements
4. **Flow of Control 2 – Repetition:** `while`, `do-while`, and `for` loops
5. **Functions**: Defining and calling functions. Using arguments to pass data to a function. Using arguments to obtain results from a function. Return values.
6. **Data Structures 1**: One and two-dimensional arrays. character strings.
7. **Data Structures 2**: Structures, collections of data components of differing types.
8. **File Input / Output**: Writing programs which obtain input from a file rather than the keyboard, and which write results to a file rather than to the screen

Grading: Grades will be computed on an A to F scale; no A+ grades will be assigned, in accordance with UMass Lowell policy. The weights assigned to the various items are:

Programming assignments	60%
Exam 1	10%
Exam 2	15%
Exam 3	15%

Incomplete grades will only be given in exceptional situations, and the student must be passing the class at the time the grade is requested.

The following rubric describes how grades will be assigned if no grading curve is applied. A grading curve may be used at the instructor's discretion, depending on the overall course average at the end of the term. Grades will not be curved down, meaning that the table below describes the minimum letter grade you will earn for a final average in each of the ranges shown:

Range	**Grade**	**Range**	**Grade**
> 92	A	78-79	C+
90-92	A-	73-77	C
88-89	B+	70-72	C-
83-87	B	68-69	D+
80-82	B-	60-67	D
		< 60	F

Programming assignments: Typically, you will have 3 to 4 days to complete each assignment. All assignments will be graded according to the program grading guidelines, to be distributed separately. Late assignments will lose 4^{n-1} points per day, including weekends and holidays. You will submit your work by uploading your files directly to your Dropbox folder.

For each assignment, you will be allowed one resubmission to improve your grade without penalty. You must resubmit your code by the given deadline for that assignment; late penalties will apply to late resubmissions. Note that the resubmission policy does not allow you to avoid penalties when the original submission is late (e.g., an assignment losing 4 points for a late initial submission has a maximum possible score of 96 for the resubmission). See the grading guidelines for more details.

Exams: Make-up exams will only be offered in exceptional circumstances. You must notify your instructor as early as possible in order to determine an appropriate make-up date.

Class participation: You are responsible for all material discussed or announced in class. You are expected to attend class regularly and participate in any in-class discussions, as such exercises are essential to your learning. Although lecture attendance is not explicitly required, regular attendance will improve your understanding of the course concepts.

Academic Honesty

All assignments and exams must be completed individually unless otherwise specified. You may discuss concepts or material covered in class, but may not share any details of your solutions to assigned problems, including algorithms and code. Plagiarism (in this course, copying code from an outside source) is also unacceptable and will be treated as an instance of cheating.

Students are allowed to discuss assignments in general terms and to help one another fix specific errors—examples include compiler errors or output formatting. In this case, students are required to note that they received assistance from a classmate by listing that person's name and the nature of their assistance as part of their assignment header. However, any sharing of code—even when used strictly to help a classmate solve a specific error—is a violation of the academic honesty policy.

Any assignment or portion of an assignment that violates this policy will receive a grade of zero for all parties concerned. Depending on the severity of the infraction, or in cases of repeat violations, additional penalties may be given at the instructor's discretion, up to and including a failing grade in the course.

Further information on the university Academic Integrity policy can be found at:

http://www.uml.edu/Catalog/Undergraduate/Policies/Academic-Policies/Academic-Integrity.aspx

Course Schedule

This schedule contains a tentative schedule of topics we will cover throughout the term; the course website will contain the most up-to-date version. The web page will also describe which section(s) of the textbook are associated with each lecture. This schedule is subject to change.

Please note that several days are denoted as "PE#"—in these classes, we will do an in-class programming exercise. While students will be able to participate even if they do not have a computer, I suggest anyone with a laptop bring it to class on these days.

The exams are currently scheduled as shown below. The first exam will be held on **Thursday, May 26 in class**, the second exam will be held on **Monday, June 13 in class**, and the third exam will be held **Monday, June 27 in class**.

Lecture	Date	Lecture Topics	Programs
1	M, 5/16	Course introduction/overview Basic C program structure; data in C	Program 1 (due 5/19)
2	W, 5/18	Operators Output with printf(); input with scanf()	
3	Th, 5/19	PE1 (Flowcharts, debugging) Conditional statements	Program 2 (due 5/23)
4	M, 5/23	Loops (while, do/while, for)	Program 3 (due 5/27)
5	W, 5/25	PE2 (Loops and conditionals) Exam 1 Preview	
	Th, 5/26	**EXAM 1**	Program 4 (due 6/1)
No Monday lecture—Memorial Day			
6	W, 6/1	Functions Pointers; pointer arguments	Program 5 (due 6/6)
7	Th, 6/2	Pointers (continued) PE3 (Functions)	
8	M, 6/6	Arrays	Program 6 (due 6/9)
9	W, 6/8	Character arrays and strings	
10	Th, 6/9	Structures Exam 2 Preview	Program 7 (due 6/15)
	M, 6/13	**EXAM 2**	
11	W, 6/15	PE4 (Structures) Dynamic memory allocation	Program 8 (due 6/20)
12	Th, 6/16	Dynamically allocated data structures	
13	M, 6/20	File, character, and line I/O	Program 9 (due 6/24)
14	W, 6/22	Bitwise operators	
15	Th, 6/23	Topics TBD; Exam 3 Preview	
	M, 6/27	**EXAM 3**	