

CS 2000: Introduction to Programming in C

Administrative Details and Syllabus

Fall 2008

Description. A computer program is a set of instructions, written in a specific programming language, which a computer follows in processing data, performing an operation, or solving a logical problem.

This course provides an introduction to essential programming concepts using the C language. Students will learn about problem solving, the decomposition of programs into functional units, control structures, fundamental data structures of C, recursion, dynamic memory management, and low-level programming, among other things. There will also be some exposure to C++.

The co-requisite for this course is CS 1010 (Introduction to Unix). This course is intended for non-CS/CE majors.

Instructor. Ganesh Gopalakrishnan. *Office:* 3428 MEB. *Email:* ganesh@cs.utah.edu. *Office Hours:* See the class web page.

Teaching Assistant. Niladrish Chatterjee (nil@cs.utah.edu). See the class web page for the TA consulting schedule.

Class Meetings. Tuesdays and Thursdays 2:00-3:20p in 2250 WEB.

Communication. The class web page is <http://www.coe.utah.edu/~cs2000>. It will be updated throughout the semester with the lecture schedule, programming assignments, laboratory exercises, links to course handouts, and more.

For questions outside of class and office hours, students are encouraged to use email.

The course staff will use the class mailing list (cs2000@eng.utah.edu) to send urgent messages to everyone in the class, such as corrections to assignments or changes in due dates. Students are not able to send mail to the class list.

Students *must* familiarize themselves with the class mailing list; please consult <https://sympa.eng.utah.edu/sympa/info/cs2000>.

We have added all your uNID emails to this mailing list. Please set up mail forwarding as you may wish. Information on mail forwarding is found at these places:

- <https://unid.utah.edu>
- <http://www.it.utah.edu/services/email/umail/index.html>

Students who would like to ask a question of the course staff should use the staff mailing list (teach-cs2000@eng.utah.edu). The instructor or teaching assistant will respond to each question directly.

Text. The required course text is *Applications Programming in ANSI C* by Richard Johnsonbaugh and Martin Kalin, 3rd edition, Prentice-Hall, 1996 (ISBN: 0-02-361141-3).

Lectures. This class will meet for lecture two times a week for one hour and twenty minutes. The instructor will make use of slides during lecture, and the slides covered during each lecture will be posted on the class web page following the lecture. Students are encouraged to take notes in class and should not expect to rely solely on posted slides to recall the material covered in each lecture.

Reading. The lecture schedule is posted on the class web page. Following each class meeting, the schedule will be updated to reflect the material actually covered by the lecture that day and to indicate the reading assignment for the next lecture. By the end of the semester, the lecture schedule will be a record of all the material covered in this course.

Laboratory Practice. Laboratory sections meet Fridays 9:40-10:30a or 10:45-11:35a in 224/226 WEB (CADE Lab #4). Each week, the lab will cover topics recently discussed in lectures, allowing students to practice applying these concepts. The labs are intended to prepare students for programming assignments and examinations. A student's participation in labs contributes to her final grade (see below).

Assignments and Grading. The final course grade will be based on seven programming assignments (45% total), two midterm examinations (30% total), a final examination (20%), and participation in labs (5%). Examination material is cumulative and will be based on lectures, programming assignments, and labs.

The following scale is used to assign letter grades.

100-93	A	89-87	B+	79-77	C+	69-67	D+	59-0	E
92-90	A-	86-83	B	76-73	C	66-63	D		
		82-80	B-	72-70	C-	62-60	D-		

In addition to the seven programming assignments that count toward the final course grade (i.e., *for credit*), there will be six programming assignments that do not count toward the final course grade (i.e., *not for credit*). Of the seven *for-credit* assignments, the lowest assignment grade will be dropped. All programming assignments must be electronically submitted (using the **handin** program on a CADE machine) by 11:59p on the due date. **Late assignments will not be accepted without a written medical excuse.** Assignments *for credit* must be completed individually.

All programming must be done in ANSI C, and all programs must compile and run in the CADE Lab using gcc version 3.4.5. Lectures and lab sections will provide instruction on using the appropriate editor, compiler, and debugger in the CADE Lab. You may choose to develop programs in the CADE Lab or on a Mac or PC. However, programs not developed in the CADE Lab should be tested there to ensure that they will compile and run during grading. *Only those programs which compile and run in the CADE Lab will be graded.* Also note that the TA is not required to help you with program development in an environment other than that of the CADE Lab.

The final exam for this class will be on Wednesday, December 17, from 1:00-3:00p in 2250 WEB.

Time Requirement. Note that for most students this course requires a substantial amount of time outside of class meetings and lab sections to complete the programming assignments. It is impossible to estimate the time necessary to finish each assignment. A program is not finished simply because the code is written. The testing and debugging phase can take quite a long time. The best way to deal such unpredictability is to start each programming assignment early.

Working Together. Working together is good way to learn a complex skill such as computer programming. Students are free to work together on programming assignments that are *not for credit*, as they are solely learning exercises. A *for-credit* programming assignment is both a mechanism for evaluating performance in this course and a learning exercise. Therefore, work on *for-credit* assignments is strictly individual. Any student who submits material that is clearly not her own or knowingly supplies code (or other information) to another student on *for-credit* assignments will receive a failing grade for the *entire course*. Of course, there must be no collaboration during examinations. Please see the University of Utah Student Code (<http://www.admin.utah.edu/ppmanual/8/8-10.html>) for a detailed description of the university policy on cheating.

Students with Disabilities. The University of Utah seeks to provide equal access to its programs, services, and activities for people with disabilities. If you need accommodations in this class, reasonable prior notice needs to be given to the Center for Disability Services, 162 Olpin Union Building, 581-5020 (V/TDD). CDS will work with you and the instructor to make arrangements for accommodations.

Syllabus. The following are the key topics planned for study, the approximate number of lectures devoted to each, and the corresponding chapters in the course text.

Getting Started (3 lectures) Chapter 0

Administrative details

Overview of computer programs, program development, and programming languages

Problem solving

Internal representation of data

Introduction to C (3 lectures) Chapter 1

Basic structure of a C program

Identifiers

Simple control flow

I/O and files

Operators and Control Flow (4 lectures) Chapters 2-3

Variables (characters, integers, and reals)

Operators (arithmetic, relational, logical, assignment, and more)

Statements (**for**, **break**, **switch**, and more)

Conditional expressions

Casting

Bitwise operations

Functions and Program Structure (2 lectures) Chapter 4

Function calls

Arguments and parameters

return statement

Function declarations and definitions

Call by value

Variable scoping

Preprocessor (file inclusion, macros, and directives)

Recursion

Arrays (2 lectures) Chapter 5
Array indexing
Character strings
String-handling functions
Multidimensional arrays

Pointers (2 lectures) Chapter 6
Pointer variables
Levels of indirection
Call by reference
Pointers and arrays
Command-line arguments

Storage Classes (2 lectures) Chapter 7
Blocks and nested blocks
Storage classes (`auto`, `extern`, `static`, and `register`)
Multiple source files
Type qualifiers (`const` and `volatile`)

Input and Output (2 lectures) Chapter 8
Opening and closing files
Character and string I/O
Formatting I/O

User-defined Data Structures (2 lectures) Chapter 9
Structures (`typedef` construct, pointers, nesting, and more)
Unions
Enumerated types

Data Structures and Algorithms (3 lectures) Chapter 10
Linked lists
Stacks and queues
Graphs and trees
Tree traversals

Introduction to C++ (1 lecture) Chapter 11