CS102 Syllabus

15 December 2011

Description

Welcome to CS 102. In this course, you will learn to program computers, using primarily the 'C/C++' language family. Although this is an older language, it is still tremendously useful, and many of the skills we learn will transcend the language itself. Programming can be difficult to learn, but as your skills develop with effort and practice, you'll find it can be great fun, and rewarding too!

> Problem solving, algorithmic design, and implementation using the C++ programming language are presented. Topics include fundamental data types and associated array types, I/O processing, conditional and loop constructs, use and implementation of functions. A brief overview of structures is given. Throughout the course, good programming styles and sound program construction are emphasized.

Tuesday, Thursday 9–10:50am in LLC 207
Three credits, prerequisite: CS101

Contact information

Instructor: Prof. Christopher League, Ph.D.

Email: christopher.league@liu.edu – please include course number in subject.

Messaging: chrysleague (AIM), league@contrapunctus.net (MSN), chrisleague (Skype), cleague@gmail.com (GTalk/XMPP)

Phone: +1 718 488 1274 (office), +1 646 450 6278 (Google voice)

Office hours: Monday 12–1; Tuesday, Wednesday 11–12; other times by appointment.

Office location: LLC 206

Resources

Software: Microsoft Visual C++ Express Edition or Apple Xcode or similar tools.

Web sites: `https://blackboard.liu.edu/` **and** `https://liucs.net/cs102f11/`

Text: *A First Book of ANSI C* by Gary J. Bronson, 4th edition, ISBN 1–4188–3556–0

Library: Campus library resources tailored for computer science are available online.

Tutoring is available from graduate assistants in the Computer Science department. The hours are Monday 11–6, Tuesday 9–2, Wednesday 11–1 and 2–5, Thursday 9–2. Check signs posted around the department.

Goals and objectives

Upon completion of the course, students should be able to…

1. demonstrate proficiency in basic algorithms and data structures.
2. understand the mathematical and logical foundations of computing.
3. understand the role of programming languages in software architecture.
4. use tools such as a compiler, editor, and development environment.
5. work with simple data models in a programming language.
6. exhibit awareness of professional organizations and technical opportunities.
7. productively attend seminars and workshops outside of classwork.

Assessment of learning

This course is part of a B.S. program in Computer Science. This section relates programmatic objectives with objectives and assessment instruments used in this course. At the completion of the B.S. program…

1.1: Students will demonstrate proficiency in basic algorithms and data structures. Introduced with course objective 1.

1.2: Students will understand the mathematical and logical foundations of computing. Introduced with course objective 2.

2.1: Students will master the fundamentals of programming languages and software architecture. Introduced with course objective 3.

2.2: Students will effectively use tools for software development.

3.2: Students will develop a data modeling design for a proposed database application. Introduced with course objective 5.

5.1: Students will exhibit awareness of professional organizations and technical opportunities. Introduced with course objective 6.

5.2: Students will attend seminars and workshops outside of classwork. Practiced with course objective 7.

Requirements

Your grade will be computed based on assignments, exams, quizzes, and participation. There are a total of 1,000 points available, broken down as follows:

- There will be **12 assignments** during the semester (roughly one per week). Assignments are worth **60 points each**, but I will **drop the lowest two scores** so only 10 will count, for a total of **600 points**.

- There are **8 quizzes** scheduled throughout the semester, to make sure you are following along and reviewing your notes after each meeting. Quizzes are worth **30 points each**, but I will **drop the lowest two scores** so only 6 will count, for a total of **180 points**.
- There will be a midterm and final exam, worth **100 points each** for a total of **200 points**.
- The remaining **20 points** are for your attendance and participation in class.

On the 1,000-point scale, you can expect the following letter grades:

		≥ 870:	**B+**	≥ 770:	**C+**	≥ 670:	**D+**
≥ 930:	**A**	≥ 830:	**B**	≥ 730:	**C**	≥ 600:	**D**
≥ 900:	**A–**	≥ 800:	**B–**	≥ 700:	**C–**	else:	**F**

In the end, I may choose to adjust the scale slightly to compensate for assignments or questions that turned out to be trickier than I intended. Such adjustments would never *lower* your grade from what is designated in the above table; if you get 930 points, you are guaranteed an **A**.

Policies

No late assignments will be accepted, because we will discuss and evaluate your work promptly after the deadline. This helps to ensure that everyone receives timely feedback, and that you can learn from mistakes while they are still fresh in your mind.

There will be no extra credit. Students usually ask for extra credit late in the semester after they have already messed up their original opportunities. Be sure to start your work early, so that we can detect and solve any problems before they can affect your grade.

Plagiarism is the use or presentation of ideas, words, or work that is not one's own and that is not common knowledge, without granting credit to the originator. Plagiarism is a practice that is not only unacceptable, but which is to be condemned in the strongest terms possible on the basis of moral, educational and legal grounds. Under University policy, plagiarism may be punishable by a range of penalties from a failing grade in the assignment or course to dismissal from the School of Business, Public Administration and Information Sciences. All students are required to read the handbook on avoiding plagiarism by visiting `http://bit.ly/lVShWN`

Cheating includes, but is not limited to the following: falsification of statements or data; listing sources that have not been used; having another individual write your paper or do your assignments; writing a paper or creating work for another student to use without proper attribution; purchase of paper or research work for one's submission as his/her own work; using written, verbal, or electronic or other sources of aid during an examination (except when expressly permitted by the instructor, depending on the nature of the examination) or knowingly providing such assistance to aid other students.

Showing up on time to class every week is extremely important. If you must be absent or more than 5 minutes late, please try to notify me in advance. I will be keeping track of whether you are in class, and when you arrive. A few missed classes will not count against you, but habitual absence will significantly hurt your grade. Additionally, there will be no make-up quizzes. I do not distinguish between 'excused' and 'unexcused'

absence. Unless you miss an *exam* due to a severe medical emergency, I don't want to see a doctor's note. If you do miss an exam, the make-up exam will be different – and probably *not* easier.

Long Island University seeks to provide reasonable accommodations for all qualified persons with disabilities. This University will adhere to all applicable federal, state and local laws, regulations and guidelines with respect to providing reasonable accommodations as required to afford equal educational opportunity. It is the student's responsibility to register with Special Education Services (SES) as early as possible and to provide faculty members with the formal communication from SES for suitable accommodations. All accommodations must be approved through SES. Contact Information: 718 488 1221 or 718 488 1044.

Time commitment

New York State defines one credit as a total of 15 hours instructional time, plus 30 hours of student preparation. Thus, a typical three-credit course will amount to 45 hours instruction plus 90 hours preparation. (For these computations, an 'hour' actually consists of 50 minutes.)

This is a lab course, for which you will have to spend a significant amount of time both inside and outside of class to succeed. In addition to spending about **1 hour** preparing (reading, reviewing, practicing) for each hour of class time, your work on the assignments is a crucial part of the learning experience. Some time will be set aside in class for supervised work on the assignments, but it will not be sufficient.

The productivity of computer programmers varies widely, depending on the project and skill level, by a factor of ten or more. (In other words, the most productive programmer can accomplish the same task in one-tenth the time taken by the least productive programmer.) This factor comes from studies of professional programmers; for beginners, the effect is probably amplified further.

For this reason, I am reluctant to estimate the number of hours a 'typical' student will need to spend on each assignment. However, the state of New York requires it, so here we go. On average, expect to spend **6–7 hours per assignment** (keeping in mind that earlier assignments will require less time than later ones), or a total of **60 hours per semester**.

- Lecture time: 4 hours per week × 15 weeks = 60 hours
- Preparation time (mostly reading): 4 hours per week × 15 weeks = 45 hours
- Programming: 6.25 hours per assignment × 12 assignments = 75 hours
- **Total: 180 hours**

You may find you need less time, or you may find you need spend substantially more time, in order to achieve the educational goal. So please don't get discouraged if you find yourself working even more than this. With practice, you will get there. Nothing worth doing is easy.

Schedule

Tue 13 Sep Meetings 1–2: Introduction, motivation, tools, identifiers, `main`, and `printf`. Hello, world! Integer and floating-point data types, arithmetic operations. *Read §2.1–2.5.*

Thu 15 Sep Meeting 3: Variable declaration and initialization, practice with all of the above, interpreting error messages, and assignment. *Read §2.6–2.7, 3.1.*

Fri 16 Sep Assignment 1 due at 6pm.

Tue 20 Sep Meeting 4: More on assignment, Input/output, and constants. *Read §3.3–3.5, 3.7.* **Quiz 1.**

Thu 22 Sep Meeting 5: Boolean expressions, logical operators, `if`, and `else`. *Read §4.1–4.2.* **Assignment 2 due at 6pm.**

Mon 26 Sep Assignment 3 due at 6pm.

Tue 27 Sep Meeting 6: Nested conditional statements, the `if/else` chain and `switch`. *Read §4.3–4.4.*

Thu 29 Sep Meeting 7: Practice with conditionals. *Read §4.5.* **Quiz 2.**

Mon 3 Oct Assignment 4 due at 6pm.

Tue 4 Oct Meeting 8: More practice with compiler errors, testing and debugging. *Read §4.6, 4.8.*

Thu 6 Oct Meeting 9: Basic loop structures: counter-controlled, condition-controlled, and the `while` syntax. *Read §5.1–5.2.*

Mon 10 Oct Assignment 5 due at 6pm.

Tue 11 Oct Meeting 10: Computations using a `while` loop. A look at `break` and `continue`. *Read §5.3.* **Quiz 3.**

Wed 12 Oct Assignment 6 due at 6pm.

Thu 13 Oct Meeting 11: The `for` statement. *Read §5.4.*

Tue 18 Oct Meeting 12: Practice with loop programming. *Read §5.5.* **Quiz 4.**

Wed 19 Oct Assignment 7 due at 6pm.

Thu 20 Oct Meeting 13: More practice and nested loops. *Read §5.6.*

Tue 25 Oct Meeting 14: Even more loops: `do/while` *Read §5.7–5.8.*

Thu 27 Oct Meeting 15: Midterm exam.

Tue 1 Nov Meeting 16: Function and parameter declarations. *Read §6.1.*

Wed 2 Nov Assignment 8 due at 6pm.

Thu 3 Nov Meeting 17: Practice with functions and return values. *Read §6.2–6.3.*

Tue 8 Nov Meeting 18: Practice with standard library functions. *Read §6.4–6.5.* **Quiz 5.**

Wed 9 Nov Assignment 9 due at 6pm.

Thu 10 Nov Meeting 19: Variable scope and storage class. *Read §7.1–7.2.*

Tue 15 Nov Meeting 20: Pass by reference (the C++ way). **Quiz 6.**

Thu 17 Nov Meeting 21: Recursion. *Read §7.5–7.6.*

Mon 21 Nov Assignment 10 due at 6pm.

Tue 22 Nov Meeting 22: Array concept and initialization. *Read §8.1–8.2.*

Tue 29 Nov Meeting 23: Practice with arrays as function arguments. *Read §8.3–8.4.* **Quiz 7.**

Thu 1 Dec Meeting 24: Two-dimensional arrays and more practice. *Read §8.5–8.6.*

Mon 5 Dec Assignment 11 due at 6pm.

Tue 6 Dec Meeting 25: Searching and sorting algorithms. *Read §8.8.*

Thu 8 Dec Meeting 26: Strings are arrays too. *Read §9.1.* **Quiz 8.**

Tue 13 Dec Meeting 27: Practice with string library. *Read §9.2–9.3.*

Wed 14 Dec Assignment 12 due at 6pm.

Thu 15 Dec Meeting 28: More practice with strings and arrays. *Read §9.4–9.6.*

Wed 21 Dec Final exam: 1:20–3:20pm in the usual lab.