CS 102 Syllabus

9 September 2015

Welcome to CS 102, an introduction to problem solving, algorithmic design, and implementation using the C++ programming language. 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.

"Computer programming is an art, because it applies accumulated knowledge to the world, because it requires skill and ingenuity, and especially because it produces objects of beauty." — Donald Knuth, 1974

When: Monday, Wednesday 11am–12:50pm
Where: LLC 207
Credits: 4
Prerequisites: CS101

Contact Info

Instructor: Prof. Christopher League, Ph.D.
Email: christopher.league@liu.edu — please include the course number (CS102) in the subject. I have several email addresses, but all messages end up in the same place, so use only one.
Google Hangout: cleague@gmail.com
Office hours: Monday, Wednesday 2–2:50 or make an appointment at https://liucs.net/bookme
Office phone: +1 718 488 1274
Office location: LLC 206, LIU Brooklyn

Resources

Web sites: We will use several web resources:

- https://liucs.net/cs102f15/ — notes, schedule, assignment handouts
- https://piazza.com/liu/fall2015/cs102 — discussion, Q&A
- http://www.gradechamp.com/ — grade reports

If you have a question or problem that might also apply to other students, *please* ask on Piazza rather than by email. Then the GA and other students

can help you too, and the solution is available for all to see. Try to use email only for personal matters such as your grades.

Text: We will use *Programming in C and C++,* an interactive text from Zyante. To obtain access:

1. Sign up at http://zybooks.com/
2. Enter zyBook code: LIUBrooklynCS102Fall2015
3. Subscribe using any credit card.

An access code may be purchased from the campus bookstore instead.

In addition, if you'd like a traditional textbook for offline reading, try *How to Think Like a Computer Scientist* http://greenteapress.com/thinkcpp/

Library: Campus library resources tailored for computer science are available at https://liucs.net/u1

Tutoring: Additional office hours and tutoring are available from my graduate assistant, Janki Vasoya, Thursday 3–5 and Friday 1–3. You can find her in the CS Department (LLC 206) or GA Room (beside the lab).

Requirements

Your grade will be computed based on projects, exams, quizzes, and readings. There are a total of 1,000 points available, composed as follows:

- There will be **13 weeks of reading assignments** in the interactive textbook. You will complete the activities at the site to earn **10 points per week,** but I will **drop the lowest score** so only 12 will count, for a total of **120 points.**
- There will be **10 programming projects** during the semester. Some class time will be devoted to guided work on the projects. They are worth **60 points each,** for a total of **600 points.**
- There are **6 quizzes** scheduled throughout the semester, to make sure you are following along, and to provide some representative questions for exams. Quizzes are worth **20 points each,** but I will **drop the lowest two scores** so only 4 will count, for a total of **80 points.**
- There will be a midterm and final exam, worth **100 points each** for a total of **200 points.**

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 achieve 930 points, you are guaranteed an **A**.

Policies

It is important to **complete readings and projects on time,** so you don't fall behind. This is especially true in a programming course, where every concept and skill builds on what came before. Reading surveys will have a couple points deducted for lateness, but you can still get most of the credit. A missed quiz will just result in a zero (no make-up quizzes), but the lowest two will be dropped. If you need to miss an exam, try to notify me in advance so we can make arrangements to make it up. **Late projects** will be graded as follows.

This formula specifies a *lateness factor* f that is multiplied by your earned score to determine a late score. The variable h represents the number of hours the submission is late.

$$f = \frac{8.5 - \log_2\left(\frac{h}{24}\right)}{10}$$

There will be no extra credit. Students usually ask for extra credit late in the semester after they have already squandered 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 `https://liucs.net/u2`

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.

In a course with programming assignments, it is usually okay to work with and learn from other students to **some** extent, but what you submit in the end needs to be

your own. The most reliable way to do that would be to set aside whatever code you created together, and then recreate it from scratch on your own.

Showing up on time to class 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,** whether psychological, neurological, chronic medical, learning, sensory, or physical. The 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 Student Support Services as early as possible and to provide faculty members with the formal communication for suitable accommodations. Visit Pratt 410, call 718 488 1044, or visit http://www.liu.edu/Brooklyn/SSS

I participate in the **LIU Safe Zone** program. Representatives of the program serve as contacts for individuals on campus with questions or concerns related to sexual orientation and gender identity, whether of self or of a friend or family member. The goal of the program is to promote a safe and free campus for all students. Safe Zone areas can be identified by a sticker with the LIU Safe Zone logo.

The **Family Educational Rights and Privacy Act** (FERPA) gives students control over the disclosure of their educational records. During this course you may have the opportunity to create accounts or register with certain public online services. In these cases, you need not make any personally identifying information public. You may use a pseudonym or online handle, as long as you identify yourself to the instructor.

Time commitment

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 **1–2 hours** preparing (reading, reviewing, practicing) for each hour of class time, your work on the programming projects is a crucial part of the learning experience. Some time will be set aside in class for supervised work, but it will not be sufficient.

The productivity of computer programmers varies widely, depending on the project and skill level. For this reason, I am reluctant to estimate the number of hours a 'typical' student will need to spend on each project. However, the state of New York requires it, so here we go. On average, expect to spend **6 hours per project** (keeping in mind that earlier projects will require less time than later ones), or a total of **60**

hours per semester. 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.

Goals and objectives

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

- demonstrate proficiency in basic algorithms and data structures (1.1, introductory level).
- understand the mathematical and logical foundations of computing (1.2, introductory level).
- understand the role of programming languages in software architecture (2.1, introductory level).
- use tools such as a compiler, editor, and development environment (2.2, introductory level).
- work with simple data models in a programming language (3.2, introductory level).
- exhibit awareness of professional organizations and technical opportunities (5.1, introductory level).
- productively attend seminars and workshops outside of class work (5.2, practicing level).

Schedule

Wed Sep 9 **Meeting 1** at 11 am. Introduction – languages, compilers, and tools.

Mon Sep 14 **Meeting 2** at 11 am. Basic input and output. *Read §1.1, 1.2, 1.3, 1.4.*

Wed Sep 16 **Meeting 3** at 11 am. Comments, spaces, errors, and warnings. *Read §1.5, 1.6, 1.8, 1.9.* **Project 1** due at 23:59.

Mon Sep 21 **CANCELED - Meeting 4** at 11 am. Identifiers, integer variables, and assignments. *Read §2.1, 2.2, 2.3.* **Project 2** due at 23:59.

Wed Sep 23 **CANCELED - Meeting 5** at 11 am. Integer and floating-point expressions. *Read §2.4, 2.5.* **Quiz 1.**

Mon Sep 28 **Meeting 6** at 11 am. Constants, math functions, and type conversions. *Read §2.6, 2.8, 2.9, 2.10.*

Wed Sep 30 **Meeting 7** at 11 am. Characters and strings. *Read §2.11, 2.12, 2.13.* **Project 3** due at 23:59.

Mon Oct 5 **Meeting 8** at 11 am. Other numeric types, randomness, and debugging techniques. *Read §2.14, 2.15, 2.16, 2.17, 2.19.*

Wed Oct 7 **Meeting 9** at 11 am. Branches and relational operators. *Read §3.1, 3.2.* **Quiz 2.**

Mon Oct 12 **Meeting 10** at 11 am. Multiple if-else blocks, and logical operators. *Read §3.3, 3.4.*

Wed Oct 14 **Meeting 11** at 11 am. Switch statements and string comparisons. *Read §3.5, 3.6, 3.7.* **Project 4** due at 23:59.

Mon Oct 19 **Meeting 12** at 11 am. String and character operations, conditional expressions. *Read §3.8, 3.9, 3.10, 3.11, 3.12.* **Quiz 3.**

Wed Oct 21 **Meeting 13** at 11 am. While loops. *Read §4.1, 4.2, 4.3.* **Project 5** due at 23:59.

Mon Oct 26 **Meeting 14** at 11 am. For loops and counting. *Read §4.4, 4.5.*

Wed Oct 28 **Meeting 15** at 11 am. Nested loops, break/continue, and incremental development. *Read §4.6, 4.7, 4.8.*

Mon Nov 2 **Meeting 16** at 11 am. Do-while and enumerations. *Read §4.9, 4.10.* **Midterm exam.** **Project 6** due at 23:59.

Wed Nov 4 **Meeting 17** at 11 am. Formatting and validating iostreams. *Read §8.1, 8.2, 8.3.* **Quiz 4.**

Mon Nov 9 **Meeting 18** at 11 am. Arrays and vectors. *Read §5.1, 5.2, 5.3, 5.4.*

Wed Nov 11 **Meeting 19** at 11 am. Iterating through arrays. *Read §5.5, 5.6, 5.7, 5.8, 5.9.*

Mon Nov 16 **Meeting 20** at 11 am. Strings as character arrays. *Read §5.11, 5.13, 5.14, 5.15.* **Project 7** due at 23:59.

Wed Nov 18 **Meeting 21** at 11 am. Functions, parameters, and results. *Read §6.1, 6.2, 6.3.* **Quiz 5.**

Mon Nov 23 **Meeting 22** at 11 am. Functions with branches and loops; using functions for unit testing. *Read §6.4, 6.5, 6.6.*

Wed Nov 25 **Project 8** due at 23:59.

Mon Nov 30 **Meeting 23** at 11 am. Pass by value, pass by reference, and other details about functions. *Read §6.7, 6.8, 6.9.*

Wed Dec 2 **Meeting 24** at 11 am. String parameters, scope, and default parameters. *Read §6.10, 6.11, 6.12.*

Mon Dec 7 **Meeting 25** at 11 am. Separate compilation. *Read §6.13, 6.14, 6.15.* **Quiz 6.**

Wed Dec 9 **Meeting 26** at 11 am. String streams. *Read §8.4, 8.5.* **Project 9** due at 23:59.

Mon Dec 14 **Meeting 27** at 11 am. Structured data. *Read §7.1, 7.2, 7.3, 7.4.*

Wed Dec 16 **Meeting 28** at 11 am. Conclusions, further examples. **Project 10** due at 23:59.

Wed Dec 23 **Final exam** at 10:15 am.