

# Introduction to programming in C++

## Syllabus - CSC 100 - Lyon College - Spring 2025

Marcus Birkenkrahe

January 6, 2025

### General Course Information

- Meeting Times: Monday/Wednesday/Friday, 9:00-9:50 AM
- Meeting place: Lyon Building Room 104 (computer lab)
- Professor: Dr. Marcus Birkenkrahe
- Office: Derby Science Building 210
- Phone: (870) 307-7254
- Office hours: By appointment via [tinyurl.com/sp25-booking](https://tinyurl.com/sp25-booking)
- Textbooks (optional + free + online):
  1. **Think C** by (2019): Starts easy then gets harder fast. [tinyurl.com/think-C-book](https://tinyurl.com/think-C-book) (also available in German!)
  2. **The Rook's Guide to C++** (2013): [tinyurl.com/rooksguide-cpp](https://tinyurl.com/rooksguide-cpp) - with my fixes to the exercises: [tinyurl.com/rooksguide-cpp-notes](https://tinyurl.com/rooksguide-cpp-notes)
  3. **Beej's Guide to C Programming (2024)**: [beej.us/guide/bgc/](https://beej.us/guide/bgc/) - assumes that you already know another programming language.

### Standard and course policies

**Standard Lyon College Policies** are incorporated into this syllabus and can be found at: [lyon.edu/standard-course-policies](https://lyon.edu/standard-course-policies).

The **Assignments and Honor Code** and the **Attendance Policy** are incorporated into this syllabus also and can be found at: [tinyurl.com/LyonPolicy](https://tinyurl.com/LyonPolicy).

In addition to these rules, please read and observe my guide to Using AI to code (written in the Fall 2024): [tinyurl.com/Using-AI-to-code](https://tinyurl.com/Using-AI-to-code).

## Objectives

This course introduces you to programming using C. We cover C++ as an extension. C is a system programming language of pure power: it enables you to converse with the computer at a level unknown to users other high level languages many of which come from C. You also learn about: compilers, working on the command line, text editors Emacs, using C for Internet of Things (IoT) devices, cybersecurity, using pseudocode and process modeling. You get a foundation in critical thinking in concert with two of the four currently most popular languages (cp. TIOBE Index 2024): the next three languages are Java, C# and JavaScript, which you can learn in a fortnight after this course). You'll learn some great tools: we use the "hacker's editor" Emacs, the world's foremost software engineering platform, GitHub, and BPMN for process modeling and pseudo code generation. For those interested in computer and data science, this course is a bridge to CSC 240, Data structures with C++, and CSC 265, Algorithms (with Python).

## Student Learning Outcomes

Students who complete CSC 100.01 "Introduction to programming in C++", will be able to:

- Learn sequential programming skills (conditional statements, loops, functions, input/output, use of data types, pointers, arrays).
- Explain the basic components of a procedural programming language.
- Apply programming to solve a variety of quantitative problems.
- Learn to use computing infrastructure (compiler, editor, shell).
- Know how to effectively present assignment results in written form.
- Be ready for "Data structures with C++" (CSC 240).
- Get ready for "Algorithms" (CSC 265).
- Improve data literacy, computational and critical thinking skills.

## Course requirements / prerequisites

- MTH 110 College algebra or permission from the instructor.
- No prior programming or computing knowledge is required.
- Experience with computers is useful but not critical.
- Curiosity is essential, and willingness to try new things!
- To get the most out of this course you must code, code, code!

## Grading system

You should be able to see your current grade at any time using the Canvas gradebook for the course.

| WHEN   | DESCRIPTION             | IMPACT |
|--------|-------------------------|--------|
| Weekly | Programming assignments | 50%    |
| Weekly | Multiple choice tests   | 25%    |
| TBD    | Final exam (optional)   | 25%    |

Notes:

- **To pass:** 60% of all available points.
- **Tests:** weekly online quizzes based on classroom lectures and practice.
- **Final exam:** random selection of the known test questions. **Note:** You only have to write the final exam if you want to improve your grade at the end of the course. If the final exam result is below your final grade average up to this point, it will be ignored.
- There will be numerous (optional) **bonus assignments** to improve your midterm and final grades.

## Rubric

| Component                    | Weight | Excellent  | Good  | Satisfactory   | Needs Improvement   | Unsatisfactory  |
|------------------------------|--------|--|---|--|---|---|
| Participation and Attendance | 0%     | Consistently attends and actively participates in all classes.                         | Attends most classes and participates in discussions.           | Attends classes but participation is minimal.                        | Frequently absent and rarely participates.                                    | Rarely attends classes and does not participate.                      |
| Programming assignments      | 50%    | Completes all assignments on time with high accuracy (90-100%).                        | Completes most assignments on time with good accuracy (80-89%). | Completes assignments but with some inaccuracies or delays (70-79%). | Frequently late or incomplete assignments with several inaccuracies (60-69%). | Rarely completes assignments and shows minimal understanding (0-59%). |
| Tests                        | 25%    | Demonstrates thorough understanding and application of concepts (90-100%).             | Shows good understanding with minor errors (80-89%).            | Displays basic understanding with some errors (70-79%).              | Limited understanding with several errors (60-69%).                           | Minimal understanding and many errors (0-59%).                        |
| Final Exam (Optional)        | 25%    | Demonstrates comprehensive understanding and application of course concepts (90-100%). | Shows strong understanding with minor errors (80-89%).          | Displays adequate understanding with some errors (70-79%).           | Limited understanding with several errors (60-69%).                           | Minimal understanding and many errors (0-59%).                        |

::

## Grading table

This table is used to convert completion rates into letter grades. for the midterm results, letter grades still carry signs, while for the term results, only straight letters are given (by rounding up). This table is also used in Canvas to compute your grades. Transitions between letter grades are rounded (e.g. 89.5% to 90%).

| Percentage       | LETTER GRADE*    |
|------------------|------------------|
| 100% to 89.5%    | A (very good)    |
| < 89.5% to 79.5% | B (good)         |
| < 79.5% to 69.5% | C (satisfactory) |
| < 69.5% to 59.5% | D (passed)       |
| < 59.5% to 0%    | F (FAILED)       |

## Schedule and Workload

For **important dates**, see the 2024-2025 Academic Calendar at: [catalog.lyon.edu/202425-academic-calendar](https://catalog.lyon.edu/202425-academic-calendar).

**Workload** (estimated):

- Time in class: 48 hrs.
- Time outside of class: 42 hrs.
- Time for tests [1 hrs/test]: 14 hrs.
- Time for coding assignments [2 hrs/pgm]: 28 hrs.
- Total number of hrs in term: 90.
- Weekly workload (outside of class): 5.625 (2.625)

Note: If you invest less than 1 hour per week in the course (outside of class), you will not pass unless you've got a secret that I don't know of, or if you already know C++ well. If you invest 1-2 hours per week, you may pass. If you invest 2-3 hours per week, you can pass with an A easily.

## Course Outline

- Week 1: Orientation
- Week 2: The Way of the Program
- Week 3: Variables and Types
- Week 4: Constants and Names
- Week 5: Input and Output
- Week 6: Operators
- Week 7: Control Flow
- Week 8: Iteration
- Week 9: Arrays
- Week 10: Functions
- Week 11: Pointers

- Week 12: Structures
- Week 13: Dynamic Memory Allocation
- Week 14: Classes and Abstraction
- Week 15: Separate Compilation
- Week 16: Standard Template Library