COURSE OVERVIEW

CSC 100 - Introduction to programming in C++ - Spring 2025

Marcus Birkenkrahe, Lyon College

January 6, 2025

Who is your lecturer?

Figure 1: Durer, Artist drawing the portrait of a man, 1519 .

- Not cats OR dogs, but cats AND dogs

- Physicist by training (making sense of this)
- Teacher by profession (making sense of this)
- Data scientist by desire (making sense of this)
- Lyon College by choice (2021 from Berlin, Germany)

Why am I teaching this class?

Figure 2: Durer, Rhinocerus, 1515 .

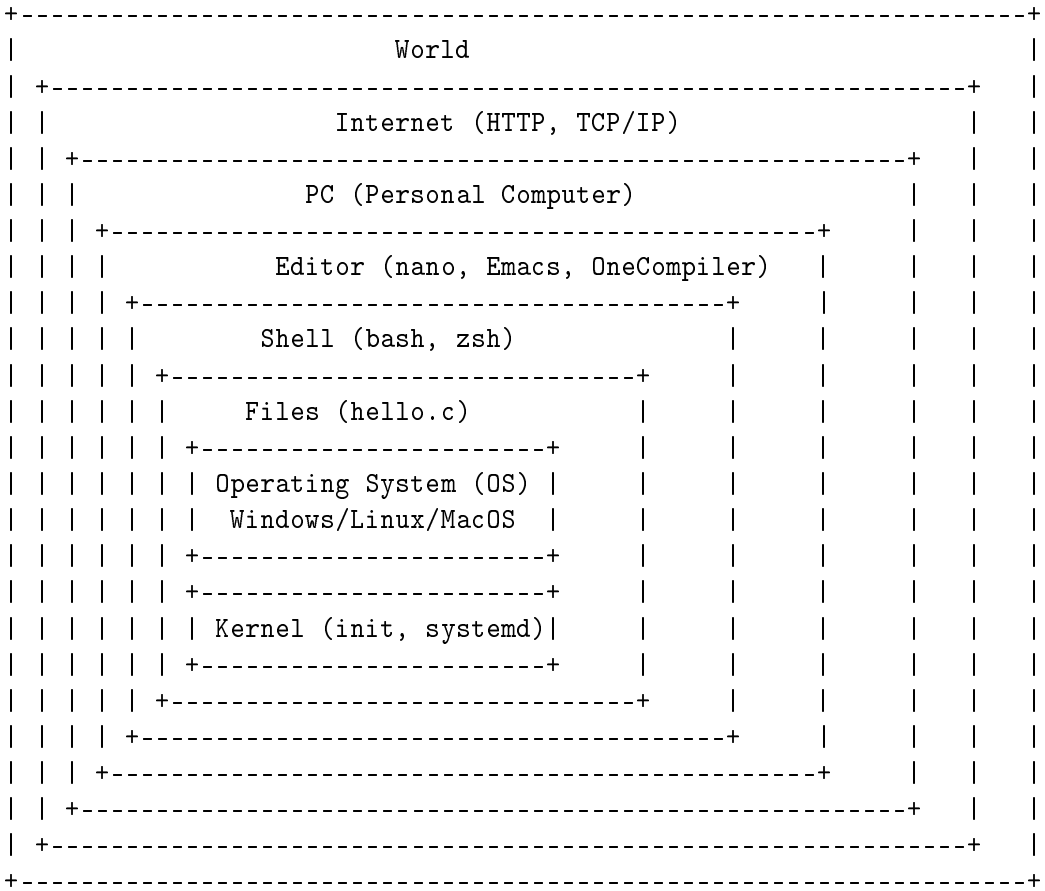
- How many programming languages are there?
- My first programming languages:
  - BASIC - Beginner's All-purpose Symbolic Instruction Code (1979)
  - FORTRAN - FORMula TRANslator (1985)
  - C++ - "C with classes" (1993)
- My last programming languages:

  - R - statistical programming language (2019)
  - Python - all-purpose programming language (2022)
  - Lisp - the first AI programming language (2024)

- Some languages that I picked up in between:
  - HTML/CSS - web layout (1991)
  - Shell - operating system control (1992)
  - SQL/SQLite - relational databases (2005)
  - Snap! - visual block-based coding (2020)

The importance of infrastructure

We will take a "holistic", or complete view of computing:



The image shows eight layers that your computing infrastructure has to manage (there are more in fact) - most of the software enabling you to do this (without you noticing) is written in C.

Which careers do our graduates have?

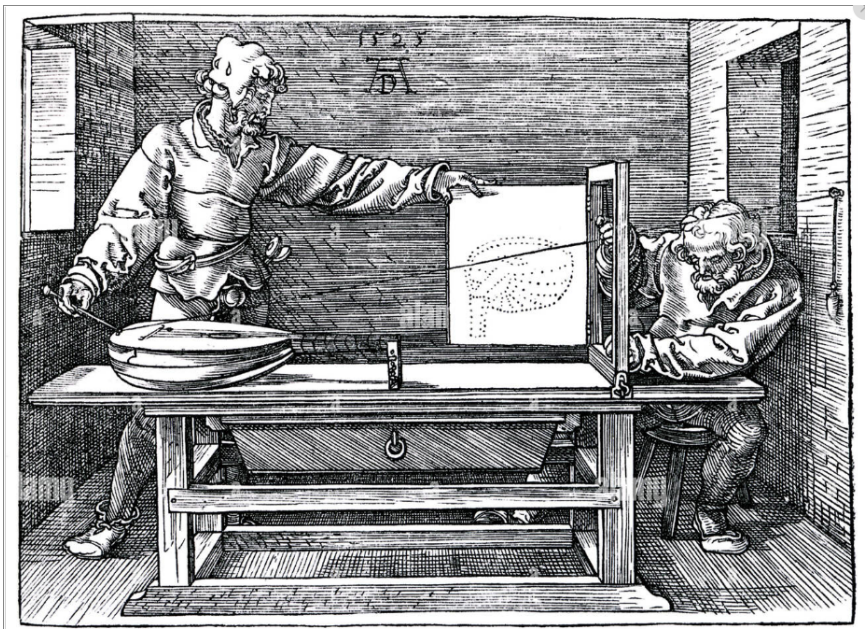


Figure 3: Durer, Perspective drawing (1525)

Where our CS and DS graduates work (2021-2024):

- Full stack developer @OneBrief (military planning software)
- Computer Aided Design at Bad Boy Mowers (comp sci/math/engineering)
- Software Engineer at Bad Boy Mowers (data science/math/comp sci)
- Data analyst at Future Fuel Chemical (data science/comp sci)
- Database specialist at the Department of Homeland Security
- Software engineer at Riot Games (League of Legends/game company)
- Cybersecurity engineer at Sierra Nevada (aircraft engineering)

- Head of data analysis at Enterprise Inc. (rental cars)
- Programmer at Outlaw Snax Chips (snack foods)
- Manager Enterprise Applications at Lyon College (IT Services)

None of this is for you? Just learn C/C++ for the fun of it!

What will you do in this course?

You will learn to:

1. Reason - using pseudocode and the C/C++ programming language/s
2. Render - using an online editor set up for C programming
3. Run - using an online command-line tool

Reason & run using pseudocode (BPMN)

Reason using pseudocode:

1. Open the BPMN online editor (bpmn.io)
2. Create a BPMN diagram
3. Pick a participant/process pool
4. Model a process using the BPMN language
5. Save model diagram as `.bpmn` or `.svg` file

Render & run using OneCompiler

Render and run code using OneCompiler

1. Open the OneCompiler dashboard (onecompiler.com/c)
2. Write source code in the editor panel (`Main.c`)
3. Optionally, add input in the `STDIN` panel
4. Compile & `RUN` code to see `Output` or an error report.

Figure 4: Durer, Angel playing lute (1497).

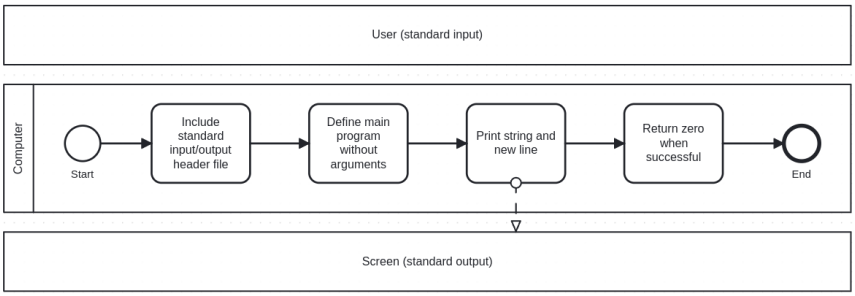


Figure 5: BPMN modeling diagram for a "Hello world" program

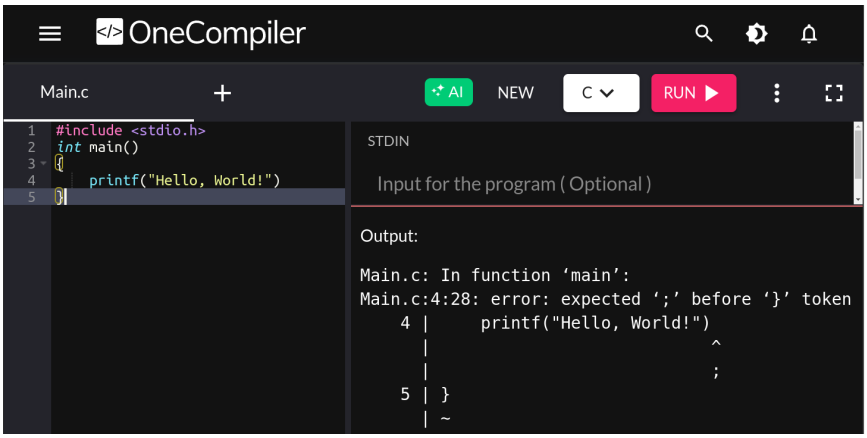


Figure 6: OneCompiler online C compiler with error message

Figure 7: Durer, St Anthony reading (1519)

How will you be evaluated?

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

Rules:

- If you have to miss a deadline, come talk to me first.
- Unexcused late submissions will be penalized (max. 50%).
- Optional final exam will be sourced by term test questions.
- Programming assignments are (usually) home assignments.
- Tests are online, open book, multiple choice quizzes for home.

Which platforms are we going to use?

All platforms are available online in the [Internet] cloud:

Figure 8: Durer, The Saints of Austria (1515)

- **Canvas** (learning management system):
  1. Wiki-page with important links
  2. Mobile app for your phone
  3. Calendar for all assignments
  4. Zoom recordings of all classroom meetings
  5. Gradebook (up to date)
  6. Tests
  7. Announcements
- **GitHub** repo: github.com/birkenkrahe/cc-25
  1. Lectures, assignments, solutions (Org-mode)
  2. PDFs for lectures, assignments, solutions
  3. Images
- **Google Drive** (automatically tied to your Lyon email):
  1. Whiteboard screenshots
  2. Large files for sharing
  3. Automatically saved Google Colab notebooks - Example

- **Google Chat Space** (you should be a member already):
  1. Motivational stuff
  2. Videos, books, articles
  3. Discussion with threads (use them!) - Example
- **OneCompiler** (register with Lyon email to keep your stuff)
  1. Assignments (download from there, upload to Canvas)
  2. Code along (during lectures and practice sessions)
  3. Code on your own (for example during open-book tests)
- **Caveat:** Cloud applications only work well when you've got a fast, reliable network and if your computer is up to it (Linux is best). An alternative on Windows is to activate **WSL** (Windows Subshell for Linux), or install **Cygwin**, or use an IDE like **Code::Blocks**. On MacOS, you've (almost) got Unix already but you need to install stuff.

Bonus: Linux + Emacs + Org-mode (for extra credit)

- If you're interested in learning and using Linux and Emacs with Org-mode: I will offer a special session on Friday, 17 January at 4 pm in room 104 to help you install & understand how to:
  1. Get WSL (Windows Subsystem Linux) on Windows 10 or Windows 11.
  2. Get the Emacs editor with my `.emacs` configuration.
  3. Write and run C programs inside the Emacs editor.
- I will record this session for later reference with Zoom, and if you cannot attend, of if you have questions, you can come to my office hours (book at tinyurl.com/sp25-booking).
- If you either attend the session and/or do it yourself using the Zoom recording (verified), you will get 10 bonus points.
- Why would you do this? Because using Emacs you can code along in the sessions a lot more effortlessly, and because Emacs is the hacker's editor par excellence & teaches you a lot more than raw editing.
- If you're not sure who you are or what you want, attend anyway to see if this is for you or not - but **let me know if you're coming**.

What did you just learn?

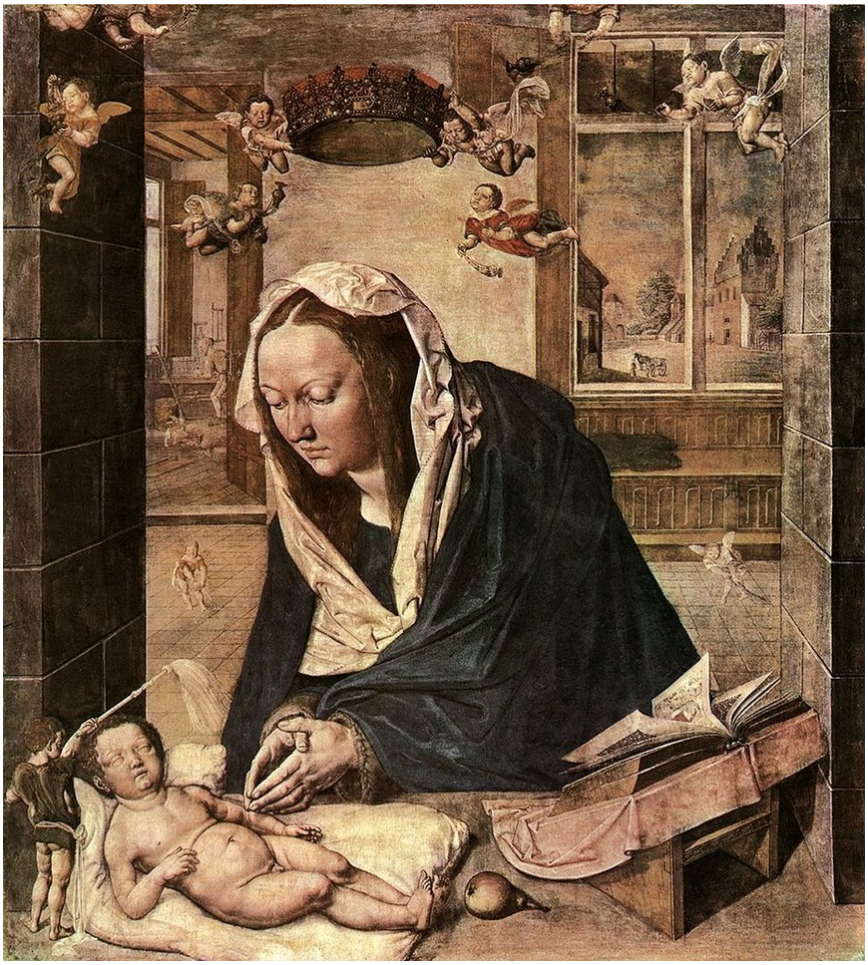


Figure 9: Durer, Madonna mit dem Kind

Review for yourself what you remember, then let's check:

1. That there are many (easy to learn) programming languages.
2. That Lyon CS and DS graduates get cool jobs fast.
3. That you'll have to read, code and run C programs.
4. How you'll be graded for your coursework.
5. Which online platforms we're going to use.

The following quote illustrates my own teaching style, too:

> "I try to say everything at least three times: first, to introduce it; second, to show it in context; and third, to show it in a different context, or to review it." (Robert Chassell)

You find up to date stuff on GitHub in the `agenda.org` file: Check it out at: `https://github.com/birkenkrahe/cc-25/`

1. Weekly updates (housekeeping, assignments, tests)
2. Preview/review of reading assignments and tests
3. Links to practice labs and code along files

First assignment

- The first assignment is to read the first chapter, "The Way of the Program" in the free online textbook Think C (copy: tinyurl.com/think-c-book) - 8 pages of text + glossary + exercises. You don't have to do the exercises!
- The first test of 30 (!) multiple choice questions (due next week) will mostly be based on this chapter.

Glossary

Computer Science is riddled with terminology. This is just from today, and we'll probably add 20-30 new terms every week. Don't be put off by that: Most of these can (and should) be looked up when needed. I'll let you know in no uncertain terms when you need to memorize anything.

Term	Explanation
BASIC	Beginner's All-purpose Symbolic Instruction Code
FORTRAN	FORMula TRANslator
C++	"C with classes"
R	Statistical programming language
Python	A versatile all-purpose programming language
Lisp	The first AI-focused programming language
HTML/CSS	Markup and styling languages for web page design
Shell	A command-line interface for interacting with the OS
SQL/SQLite	Languages for managing/querying databases
Snap!	A block-based visual programming language
nano	A simple, easy-to-use text editor
GCC	GNU Compiler Collection
bash	A Unix shell and command-line interpreter
BPMN	Business Process Model and Notation (process modeling)
Canvas	A learning management system (LMS)
GitHub	A platform for hosting and sharing code
Google Colab	A cloud-based environment to code, compile, run programs
Google Drive	A cloud storage service for sharing stuff
Google Chat	A communication platform for sharing and chatting
Linux	An open-source operating system
Cygwin	A suite of Unix utilities for Windows
OneCompiler	Online editor and command-line application