COMP6771
Advanced C++ Programming

1.1 Course Outline

Teaching Staff

Lecturer in Charge
Imran Razzak

Course Admin
Simon Haddad

Potential Guest Lecturers
Optiver
Nathaniel Shead

Tutors
Jack Peng
Jayden Leung
Junyuan Zhou
Peiyu Tang
Rory Golledge
Simon Haddad
Sunny Chen

Course Objectives

You will develop:

1. Skills in writing software using modern C++.
2. Skills in writing unit tests to create robust software.
3. Skills in using libraries to develop software.
4. Skills in using tools to build software.
5. Knowledge & understanding about imperative, generic, and object-oriented programming.

What is C++?

- General-purpose programming language evolved from C.
- Created by Bjarne Stroustrup in 1979.
- Backwards-compatible with C.
- Designed to run natively, directly on hardware.
- Only language lower than C++ is assembly.
- Supports development of **zero-overhead** & **opt-in overhead abstractions.**
- Multi-paradigm: imperative, generic, object-oriented, functional.

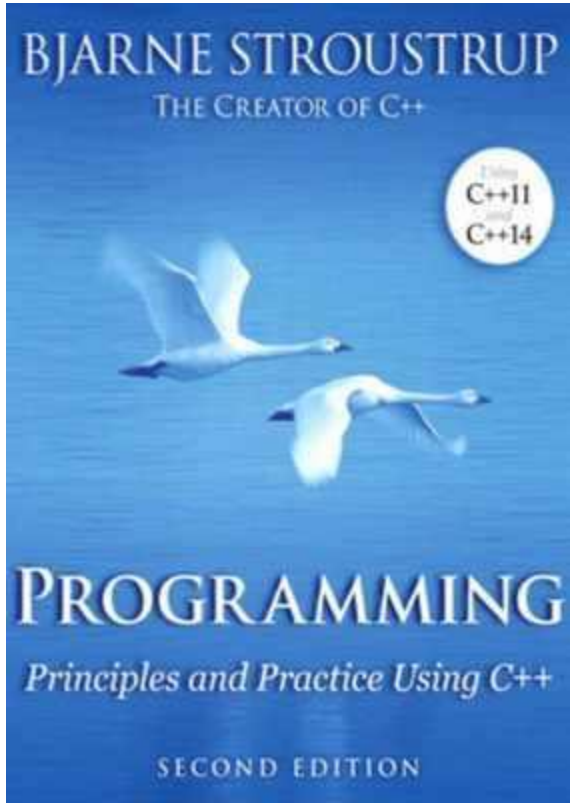
What isn't C++?

- **C++ is not C!**
- Easy to think you can build your C++ understanding on top of your C understanding.
- Valid C code is often (but not always!) valid C++ code, but good C is very different from good C++.
- C and C++ have a large common intersection but are distinct languages. For example, in teaching best practices, we will not be using:
  - `malloc()/free()`
  - C-style arrays
  - C-style strings
- And will sometimes discourage the use of raw pointers (`int*`, `char*`, etc.)

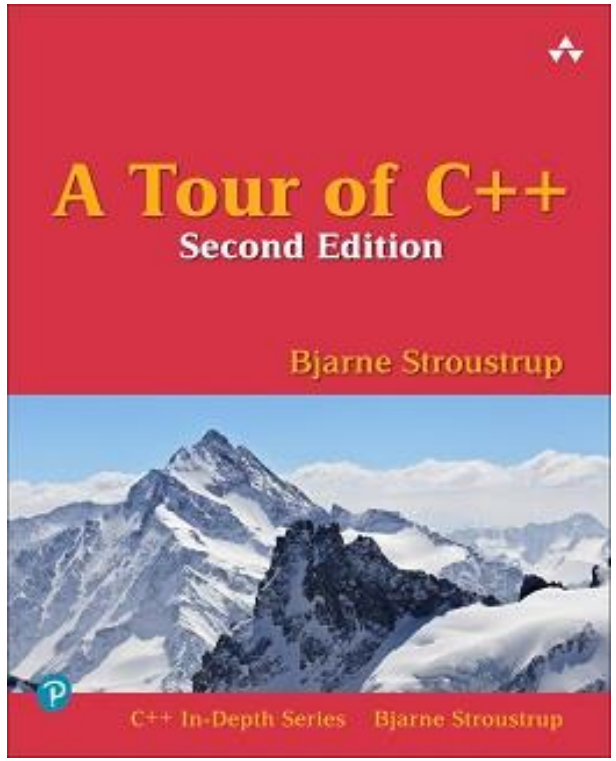
What is C++ Good For?

- Operating systems
- Low latency software (e.g., high frequency trading)
- Direct control of hardware
- Game Development
- Implementations of *other* programming languages (e.g., NodeJS)
- Just about anything you can think of!

Learning Resources



Comprehensive intro to C++
(>100 pages of exercises!)



Will help you in a pinch (e.g. before exams and interviews)
Readable in 4 hours.

[cppreference]

- Good for looking up APIs and recalling language rules
- **DO NOT USE CPLUSPLUS.COM**

Where to Get Help

- Your question/answer hierarchy:
- Edstem forum.
- Your tutor (see Timetable page for links)
- Lecturer (cs6771@cse.unsw.edu.au)
- Imran (imran.razzak@unsw.edu.au)

Only questions that are non-sensitive will be answered on the forum.

Course Schedule

See the course outline for the full schedule.

Weekly teaching includes:

- 4 hours of lectures
- 2 hours of labs
- Incremental development on assignments

We may provide additional material and webinars to assist in your learning. While these will be recommended, they will not be required.

Assessment Schedule

Assessment	Weighting	Due Date
Weekly Exercises	10%	Weeks 1 – 5, 7 – 10 (Sunday of that week, 8pm)
Assignment 1	15%	Late Week 3 (see assignment spec)
Assignment 2	20%	Early Week 7 (see assignment spec)
Assignment 3	25%	Early Week 10 (see assignment spec)
Exam	30%	Exam Period

Exam Assessment

Final exam **may** be scaled.

Final exam:

- No hurdle

Assignments:

- have an emphasis on correctness
- rely on version control (assumed knowledge)
- have a late penalty outlined in the specification

Plagiarism will not be tolerated.

- Immediate zero for assignment.

Course Tools

Course Website on WebCMS3

Tutorials & Assignments distributed through Gitlab

Developer Environment:

- Build C++ using CMake
- C++ Compiler (must have C++20 support): GCC 10+, Clang 12+
- Recommended Editors: VS Code, CLion
- Documentation: cppreference
- Git

Feedback (stop recording)