Object-Oriented Principles and Practice / C++

Alice E. Fischer

January 23, 2018
OOPP / C++
  Mechanics

Programming
  Context

Course Goals
  Course Goals
Course Information

All information about this course is posted on the course website:

http://eliza.newhaven.edu/cpp/

- Syllabus
- The main textbook, Exploring C++ (my compiled notes)
- The lecture schedule and weekly lecture notes
- Code samples
- Programming assignments

**Course Requirements:** A term project built by combining about ten weekly programming projects, with documentation (∼50%), midterm (∼20%), and final (∼30%).
Course mechanics

- I expect you to learn and follow the design and coding standards covered in the lecture and the book.
- Turn in your zipped-up assignments by email to my home address.
- It is your job to prove that the code works. Therefore: include adequate test output with every submission.
- If you need help with an assignment, zip it and send it to me with HELP in the subject line. I cannot give you any realistic help unless I can see the entire code base, as it was, when the problem occurred.
- Late assignments will be accepted. However, lateness is self-punishing. If you cannot do the work on time, you will not understand the lectures and you will do badly on the exams.
Your Programming Environment

Please download and use either XCode (Macintosh) or Eclipse (Windows and Linux). Eclipse users need to get the CLang compiler or the Gnu C++ compiler (g++) not the stripped-down mingw compiler. Widows people will need to use cygwin.

Everything you turn in must be written entirely in standard C++11 standard, and I must be able to compile it on my machine without modification. DO NOT include any Windows pathnames or any Microsoft-only header files! (conio.h, windows.h).

The Visual Studio IDE puts inappropriate things into your code and does not give good error comments. Students who use it inevitably have more trouble than others.

CodeBlocks does not supply the minimum level of support and diagnostics that you need. Don't waste your time on it.
Reference

For online reference, use one of these sites.

www.cplusplus.com
http://en.cppreference.com/w/

If you want a printed textbook, I recommend that you buy the book Herbert Schildt, *C++: The Complete Reference*. It serves as a reference manual for C++ syntax, as well as a basic text.

Do not use online sites such as Stack Overflow or Geeks for Geeks. They contain notoriously bad advice and bad code that does not meet the standards of this course.
Kinds of Programming
Three views of programming

People program for different reasons.

Programming can be . . .

1. One-time software: code that solves a computational problem.
2. Personal software that you can reuse.
3. Building a product to be used by others, over a period of several years.
Problem solving

Desired properties of single-use programs for solving problems:

▶ Correct outputs from correct inputs
▶ Succinct expression of algorithm
▶ Simple development cycle

Beginning programming courses focus on solving small problems.
Script programming is quickly written and, often, quickly forgotten.
Software Product Construction

Desired properties of software constructed for widespread use:

- Correct outputs from correct inputs
- Robust in face of bad inputs; reliable
- Economical in resource usage (time and space)
- Understandability and verifiability of code
- Security
- Ease of repurposing
- Ease of deployment
- Maintainability
Programming in the large

This course will focus on constructing large-scale software.

- Interface design and quality is important
- Thousands of lines of code
- Potentially written by many programmers
- And maintained over the years by different programmers
- Over a large span of time
- Deployed on a large number of computers
- With different architectures and operating systems
- Interacting with foreign code and devices
Course Goal: Recognize and Build Quality C++ Programs

Code should . . .

1. Meet the specifications.
2. Be correct. I should have reason to trust the correctness.
3. Be debuggable.
4. Be portable.

Quality C++ programs are written according to strict OO design standards.
Learn what Object-Oriented programming is – and isn’t.

- Learn about classes, objects, type hierarchies, templates, exceptions, the template library, and their implementations.
- Learn how C++ objects are implemented and managed, and how to use move semantics.
- Learn and use the basic principles and patterns of OO design.
- Learn and use the new capabilities of C++11 and C++14.
Course goals, continued.

- Follow instructions; question them if you think they are wrong.
- Create a complex program one module at a time.
- Learn clean OO design paradigms and practices.
- Write code that is concise, readable, and non-repetitive.
- Test, analyze, debug, and present your work in a professional manner.
- Create and use a makefile.
- Use a reference manual.
Tools to Achieve the Goals

All of these tools are needed in this course:

1. Privacy: isolate each class from the others.
2. Types: use the correct type for everything. No more using 1 for a truth value.
3. A good C++11 or C++14 compiler with warnings turned on.
4. Effective debugging technique, with or without a debugger.
5. Carefully designed test suites and well documented results.
6. Design patterns.
Chapter 1 of Exploring C++

- C++ is a very powerful language, which, if used badly can produce projects that are badly designed, badly constructed, and impossible to debug or maintain.
- Your goal is to learn to use the language well, and with good style.
- Please read *and follow* the style guidelines in Section 1.2
- Download the two tools files from the Code Examples page of the website.
- Then read Section 1.3, about the tools library, and use this information to customize your own copy of the tools.