Philosophy

Good design and programming is not learned by generalities, but by seeing how significant programs can be made clean, easy to read, easy to maintain and modify, human-engineered, efficient, and reliable, by the application of good design and programming practices.

Course Contents

- Practical aspects of software design and programming
  - Intro to software lifecycle and OO design
  - Reuse of design patterns and software architectures
  - Developing, documenting, and testing reusable class libraries and frameworks
  - Building application based on reusable components

- Central themes are
  - Good programming principles and practices
  - Design patterns
  - Separation of interface from implementation

Course Work

- Construct components and applications with UNIX tools
  - e.g., make, emacs, dbx, gprof

- Programming language is C++

- 6 related projects that illustrate key design and programming principles via hands-on experience

- Weekly quizzes

- Final exam
Textbooks

- **Required**
  - Design Patterns: Elements of Reusable Object-Oriented Software, Gamma et al., Addison-Wesley, Reading, MA, 1994
  - The C++ Primer (Second Edition) by Stanley Lippman

- **Recommended**
  - Object-Oriented Design with Applications by Grady Booch
  - Object-Oriented Software Construction by Bertrand Meyer
  - The C++ Programming Language by Bjarne Stroustrup
  - The Annotated C++ Reference Manual by Stroustrup and Ellis
  - Effective C++ by Scott Meyers