Class Structure

Description

This is a foundation course on learning modern C++ language (C++ 11 and beyond).

This course covers object-oriented programming using modern C++. Because most automation, embedded applications, gaming, and many large data processing applications are written in C++, it is essential that software developers understand and master it. Hardware engineers are increasingly using C++ and OOP in system verification tasks, and as of the current decade, there has been a noticeable shift from C to C++ on microcontroller systems.

This course offers an excellent foundation in developing optimized modern C++ applications. The participants will learn to write faster and modular code, and the debugging techniques, for real-world applications. There will be assignments and exercises to accompany the lectures.

Learning Outcomes
At the conclusion of the course, you should be able to:

- Modern C++ (post C++11)
- Apply object oriented concepts to software development problems using modern C++
- Understand and use the basic programming constructs of C++
- Manipulate various C++ data types, such as arrays, strings, and pointers
- Write C++ code using principles of object-oriented programming
- Understand design patterns in C++ using singleton pattern
- Manage memory appropriately, including proper allocation/deallocation procedures using unique and smart pointers
- Best practices (dos and don'ts)
- Utilize best practices from C++ GSL or Google Abseil libraries

1. Introduction

- Setup development environment
  - Basics of Docker and container management
- Review class structure
- Operating System Fundamentals: Linux, Mac, POSIX, RTOS
- Modern approach to "manpages"
  - tldr
  - chatgpt
- SSH and password-less login
- System Monitoring: `top` , `df`

- Homework: Operating Systems and POSIX

2. Basic Syntax & Functions

- Basics
  - Types, strings
  - Control structures
- Declare and define functions
  - Call by value, reference, pointer
  - Overloading
  - Default parameters

3. Object Oriented Basics

- Classes and objects
- Constructors and destructors
- Inheritance

4. Operators, STL library & Templates

- Operators (=, [], *, -> etc.)
- Various containers
- Iterators
- Streams
- Chrono
- Templates and the need for header only code

5. Midterm

- Review Questions
- Feedback
- Exam

6. Object Oriented Advanced

- Abstract classes
- Virtual and pure virtual functions
- Virtual destructors
- Polymorphism

7. Smart Pointers

- Unique pointer
- Shared pointer
- Weak pointer

8. More Advanced Topics

- Concurrency
  - Threads
  - Mutexes
  - Condition Variables
- Deadlock Empire

9. Powerful Libraries

- Libraries
  - Boost
  - Guideline support library (GSL)
  - Abseil
- Design patterns
  - Singleton

10. Final Examination

- Review Questions
- Collect Feedback
- Exam

Revision #6

Created 3 weeks ago by Preet Kang

Updated 5 days ago by Preet Kang