Object-Oriented Programming in C++
CGS 3406
About Myself

• Primary Research Areas
  – Real-time Systems
    • Operating system design
    • Scheduling
  – Storage

• Use C/C++ (among other languages) in my research
About This Course

• Introductory *programming* course
• Material applicable to many programming languages
• Object oriented programming
This Course

- Remember that this is primarily a course on programming
- However, you will have to learn some C++
  - Basic facilities
  - Learning through doing
  - Practice
Objectives

- Basic understanding of software and hardware
- Solve computing problems using a top-down approach with a well-structured design using procedural programming techniques
- Design, implement, test, and debug a C++ program to solve problems
- Control structures used in procedural programming
- Make use of data types and structures in C++
- Design and implement algorithms to perform common tasks
- Demonstrate competence with the use of functions, reference parameters, arrays, pointers, recursion and I/O
Meeting the Objectives

- Lectures
- Book, notes, online resources, ...
- ~8 Programming Assignments
- 3 Tests
- Quizzes
- More information in the course syllabus
Programming
Programming

• Translating an idea (e.g., algorithm) into a form that can be run on a computer
• Tell the computer what to do
  – Easy?
• Problem solving
• Is programming the same as computer science?
Computer Hardware

- **CPU**
  - Executes machine language

- **Main memory**
  - Stores instructions and data

- **Secondary storage (e.g., SSD, hard disk)**
  - Persistent
Programming Languages

• Machine Language
  – Based on machine's core instruction set
  – Difficult for humans to read (1's and 0's)
  – Example: 1110110101010110001101010
    • Say what?

• Assembly Language
  – Translation of machine instructions to symbols, slightly easier for humans to read
  – Example: ADD $R1, $R2, $R3
  – Well, we know it has something to do with addition!
Programming Languages

• High-level procedural languages
  – Abstraction of concepts into more human-readable terms
  – Closer to "natural language" (i.e. what we speak)
  – Easy to write and design, but must be modified for computer to execute
  – Examples include C, Pascal, Fortran

• Object-oriented languages
  – Abstraction taken farther than procedural languages
  – Objects model real-world objects, not only storing data (attributes), but having inherent behaviors (operations, functions)
  – Easier to design and write good, portable, maintainable code
  – Examples include Smalltalk, C++, Java
C++
Why C++

“There are only two kinds of languages: the ones people complain about and the ones nobody uses.”

— Bjarne Stroustrup, The C++ Programming Language

• Portability
  - supported on nearly all platforms (including embedded systems)

• Supports major programming styles

• Used in many projects
  - Typically systems and embedded

• Need a programming language to learn programming
This Course

• The entire C++ language will not be covered
  – Not necessary to meet objectives
  – This course won't make you an expert in C++
    • C++ has many “interesting” rules/constructs

• General approach
  – Programming concepts, techniques, principles
  – Map them to code
  – Process to perform the mapping
"Hello, World!"

- Program that says Hi
- Very simple program
- Verify
  - Basic understanding of language syntax
  - Build tools
  - Execution platform
#include <iostream>       // pre-processor directive
using namespace std;

int main()             // start of program
{
    cout << "Hello, World!\n"; // standard output stream
    return 0;          // return value to operating system
}
Compile, Link, Execute

Source Code (.cpp, .h) → compiler → Object code (.o)

Executable Program

linker → Libraries
Executable Program

standard output stream
Operating System

- Libraries
- Kernel

Application
Coding

- Quality
  - Design
    - Organization
  - Ease of understanding
  - Self documenting code

- Style

- Debugging
Plagiarism

the act of using another person's words or ideas without giving credit to that person
- Merriam-Webster Dictionary

• Code
  – Do not “share” assignment code/ideas from others
  – Do not claim others' code as your own
• Discussing material is encouraged
  – Especially for exams
  – Help each other out
• List any sources you used
  – If in doubt, list the source
• Plagiarism detection software will likely catch cheating
• Expect to verbally explain assignment code you turn in
How to Succeed

“For the things we have to learn before we can do them, we learn by doing them.”
— Aristotle, The Nicomachean Ethics

- Write code frequently
- Ask questions
  - Office hours
- Assignments
- Tests/Quizzes
- Bring computer and do examples in class
Till Next Class

• Program and execute a “hello world” program
  – Computer
    • Lab
    • Personal
  – Software
    • Visual Studio
    • g++
    • ...