Computing & Software Systems 502:
Data Structures and Object-Oriented Programming II
Winter 2016
Syllabus

Description

This course is a continuation of CSS 501 and covers advanced data structures including trees, balanced trees, heaps, graphs, and hash tables along with the associated algorithms. Students learn how to analyze a problem and design and implement a solution using object-oriented design and programming with a focus on inheritance and polymorphism. Students also study the basic formal notations of languages cover topics such as context free grammars.

Details

Instructor: Bob Dimpsey, dimpsey@uw.edu
Office Hours: MW 4:45 – 5:45, Truly House
Course Website: http://courses.washington.edu/css502/dimpsey
Lectures: MW 6:00-7:40 pm, UWBB 240

Grader: Hao Wu, whmay.us@gmail.com

Textbooks

Books Utilized in Class

Data Abstraction and Problem Solving with C++: Walls & Mirrors (6th Edition), Frank M. Carrano, Addison-Wesley, 2013.

Available at the campus library here .

Optional

An Active Introduction to Discrete Mathematics and Algorithms, Charles Cusack, David Santos, GNU Free Software, 2014:
http://www.cs.hope.edu/~cusack/Notes/Notes/Books/Active%20Introduction%20to%20Discrete%20Mathematics%20and%20Algorithms/ActiveIntroToDiscreteMathAndAlgorithms.2.5.pdf

Discrete Mathematics and Its Applications (7th Edition), Kenneth H. Rosen, McGraw Hill, 2012.

It is also recommended that you obtain a good C++ book.

Recommended C++ Books

Thinking in C++, Vols. 1&2, Bruce Eckel, Prentice Hall.

C++: How to Program (8th Edition), H.M. Deitel and P.J. Deitel, Addison-Wesley, 2013.

The C++ Programming Language (3rd Edition), Bjarne Stroustrup, Addition-Wesley, 2000.

Effective C++: 50 Specific Ways to Improve Your Programs and Designs (2nd Edition), Scott Meyers, Addison-Wesley.

More Effective C++: 35 New Ways to Improve Your Programs and Designs (2nd Edition), Scott Meyers, Addison-Wesley.

C++: The Core Language, Gregory Satire and Doug Brown, O'Reilly. (Good if you are coming from a C background)

C++ Notes from Professor Zander: http://courses.washington.edu/css342/zander/css332/

Grades

This course is not graded on a curve. That is, there is not a pre-defined percentage of the class to be assigned to different grade ranges (e.g., x% 4.0, y% 3.5, etc). It is feasible (although unlikely) that all students can receive the highest range.

Also, it is not possible to pre-determine what scores on the exams and tests correspond to what grade ranges.

Grades consist of two major components in the percentage defined by the table below and will be determined as objectively as possible. Coding guidelines are an important part of the objective grading of programming assignments. There will not be a subjective component based on class participation.

Results of exams, quizzes, and assignments are averaged together corresponding to the ratios below. They are then grouped and mapped to traditional to 'A', 'B', 'C', etc. for the quarter. Decimal grades are then computed using the equivalences in the time schedule, linearly interpolating between letter-grade boundaries.

There will be two exams in the quarter, a mid-term and final. There will also be one or more shorter quizzes which will reinforce current material being covered. Assignments will consist mainly of programming problems – there will be five major programs written. The last will also have a design phase grade associated with it.

At any time I will be happy to consult with you on how you are doing in the class relative to grade ranges.

Course Work	**Percentage of Grade**
Exams: Midterm, Final	50%
Programming Assignments	50%
Total	100%

Assignment Submissions

Assignments are to be submitted electronically to Canvas. A link is available on the course Web Site.

Due time for electronic submission is 6 pm PST on specified dates. Late submissions will not be accepted unless there has been previous agreement due to extenuating circumstances

Class Attendance

Class attendance is not mandatory and there is not a grade based upon attendance or participation. However, I strongly encourage you to come to class as there is generally a direct correlation between attendance and performance (aka, grade). Also, quizzes and exams will be given during class. Finally, you will be held responsible for all material covered in class, regardless of its presence (or lack thereof) in the textbook or Web Site.

Programming Assignments

All programming assignments will be submitted via UW canvas. Late assignments will not be accepted.

The program is graded upon the follow aspects:

- Style / Syntax
  - Following Coding Guidelines
  - Clarity
- Factoring / Design
  - Modularity
  - Interface Design
- Correctness – determined through test cases run against code
- Efficiency

You can use any IDE for coding and testing your laboratory and programming assignments. However, I will be using Microsoft's Visual Studio and recommend that you do as well. The Microsoft DreamSparks program provides a copy of VS free to all students. You can find links on the course Web Site. I am currently using version 2015.

Independent of what version you use to develop your programs, **there are only two options available for what environment will be used to grade**. Either the program will be compiled with g++ and executed on a Linux operating system OR on Visual Studio 2015. You can choose which of these two options you would like the grader to utilize.

If your program does not compile or does not run in the chosen these graded environments it will be deemed incorrect. This is true even if it works with others OS's or compilers. This means that if you use a different development environment you should port and test your code to the prescribed environment before submitting it. The Linux lab is available for this purpose if you do not have a prescribed local development environment.

Programs are turned in as soft copies to UW canvas. Each soft copy must be a zipped file—only zip will be accepted. The zip file contains source code (in ASCII Text) and results of the program. Please check each homework specification about what you to write in your report. For grading correctness I will unzip each file, compile the code and execute a number of test cases.

Special needs

If you require academic accommodations, please contact Disability Support Services at (425) 352-5307, TDD (425) 352-5303 or email at dss@uwb.edu. More information is available at the DRS web Site: http://www.uwb.edu/studentservices/drs .

Tentative Class Schedule

The following is a tentative list of subjects, assignments, and tests that we will cover through the quarter. This will undoubtedly change as we make our way and adjust through the quarter.

Date	Topics / In-Class Tests	Reading	Assignment
1/4	Intro, Core C++ Tenets, Guidelines, tool Chain, Complexity Review	Chapt. 10	Lab1 Assigned
1/6	Trees: Intro, BST	Chapt. 15-16	
1/11	Trees: 2-3, B, AVL, Red-Black	Chapt. 19	
1/13	Trees: finish		
1/18	Martin Luther King Jr. Day, No Class		Lab2 Assigned
1/20	Priority Queues	Chapt 13.3, 14 (partial)	
1/25	Heaps, HeapSort, Huffman	Chapt 17	
1/27	Dictionaries/Hashing	Chapt.18	
2/1	OO1: Inheritance, Polymorpishm, virtual functions	Interlude 2, 4 C++ book, notes	Lab3 Assigned
2/3	OO2 cont'd: Iterators, exceptions, STL, ptrs. to functions, casting	Interlude 3, 6, C++ book, notes	
2/8	Midterm Review		
2/10	**Midterm Exam**		Lab4 Assigned
2/15	President's Day, No Class		
2/17	Graphs: Intro, BFS, DFS	Chapt 20.1, 20.2	
2/22	Graphs: Dijkstra	Chapt 20.3, 20.4	
2/24	Graphs		Final Project Assigned
2/29	Languages Intro, Reg Expressions	Chapt. 5.1, 5.2	
3/2	Finite State Automata		
3/7	context free grammars		
3/9	Turing Machines		
3/16	**Final Exam**		