CSC 300 Data Structures
SDSM&T Course Syllabus for Fall 2008

Course: CSC 300 Data Structures (4 credits)
Prerequisites: CSC 250 Computer Science II and CSC 251 Finite Structures
Room: McLaury 313
Time: MWRF 10:00-10:50AM

Instructor: Dr. Weiss
Office: McLaury 311
Phone: 394-6145
Email: john.weiss@sdsmt.edu
Office Hours: MWF noon-12:50PM and Th 11:00-11:50AM, or by appointment

SDSM&T Course Catalog Description: CSC 300 Data Structures
(4-0) 4 credits. Prerequisite: CSC 250 and CSC 251. A systematic study of data structures and the accompanying algorithms used in computing problems; structure and use of storage; methods of representing data; techniques for implementing data structures; linear lists; stacks; queues; trees and tree traversal; linked lists; and other structures.

Prerequisites
CSC 300 is an intermediate-level computer science course. Students taking this class should have completed at least one year of college-level computer programming coursework, and should be able to write substantial computer programs in a high-level programming language such as C++. More specifically, this course assumes a programming background that includes coverage of recursion, structs and classes, arrays, pointers, dynamic memory management, and linked lists in C++. Students should also have a solid mathematics background, including college algebra and discrete mathematics.

Textbook: Data Structures and Algorithm Analysis in C++, 3rd ed.
(Mark Weiss, Addison-Wesley, 2006)

Course Requirements
1) programming assignments (4) 30%
2) problem sets (3) 10%
3) exams (2) 30%
4) final exam* 30%

* In order to pass this course, you must receive a passing grade on the final exam.

Topics
1) Introduction to algorithm analysis (1 week)
2) Simple data structures (linked lists, stacks, queues, binary trees) (2 weeks)
3) Advanced data structures (height-balanced trees, heaps, graphs) (5 weeks)
4) Related algorithms (searching and sorting, hashing, graph algorithms) (5 weeks)
5) Advanced topics (greedy methods, dynamic programming, NP-completeness) (2 weeks)
6) Advanced C++ topics (templates, inheritance) (1 week)
Objectives
CSC 300 is the third course in a four-semester sequence designed to teach students the fundamentals of problem solving on the computer. This sequence provides students with the skills required for computer programming, algorithm development, algorithm analysis, and software development, as well as proficiency in a high-level programming language (C++).

The primary objective of this course is to introduce students to a wide variety of fundamental data structures and associated algorithms. The course material provides a foundation for all upper-level computer science courses.

Outcomes
Upon completion of this course, students will obtain the following outcomes:
• intermediate-level problem solving and algorithm development skills on the computer
• ability to analyze algorithms using big-Oh notation
• understanding of fundamental data structures such as lists, trees, heaps, and graphs
• understanding of fundamental algorithms such as searching, sorting, and hashing
• increased fluency in the high-level programming language C++
• ability to write programs using both procedural and object-oriented paradigms
• ability to use the Linux operating system for software development
• greater understanding of the software development process
• experience working in teams

Software
CSC 300 is a programming-intensive course, so be prepared to spend many hours struggling with the computer this semester. We will be using the GNU C++ compiler on the Linux operating system for programming assignments. Linux and the GNU C++ compiler are available on the computers in the departmental Linux Lab in McLaury 215; each student enrolled in CSC 300 will be given an account to use on these systems. You may install this free software on your home PC as well. Further instructions and handouts on using GNU C++ and Linux will be given in class as the semester progresses.

Grading
Programming assignments are weighed heavily in the determination of your final grade. Unless your programs run correctly, it is unlikely that you will pass this course, so be sure to leave yourself enough time to complete the programming assignments. There is no “late policy” in this class; all assignments must be turned in by the due date, otherwise they will not be accepted. To receive full credit, programs must not only be correct, but must adhere to good programming style guidelines (standardized formatting, meaningful identifiers, modular code, good documentation, etc.). Programs will be tested using the GNU C++ compiler on Linux (not the Microsoft Visual Studio C++ compiler on Windows). Program grading is further discussed in the Programming Guidelines document on the course Website.

Academic Integrity
Although you may exchange ideas with your classmates, you must complete these assignments by yourself (or with members of your team, in the case of group projects). In particular, it is forbidden under any circumstances whatsoever to exchange source code with your classmates. COPYING CODE IS A SERIOUS INFRINGEMENT UPON THE SDSM&T ACADEMIC INTEGRITY POLICY, AND WILL BE TREATED ACCORDINGLY. Academic integrity is further discussed in the Academic Integrity policy statement on the course Website.
Attendance
Attendance is required for all courses at SDSM&T. You are responsible for the lecture material as well as any assigned readings. Lecture material may diverge significantly from the assigned readings, so good attendance is particularly important in this class. Attendance is further discussed in the Classroom Conduct policy statement on the course Website.

Course Website
http://www.mcs.sdsmt.edu/csc300

Electronic Devices Policy
Please turn off cell phones and pagers before class starts. Notebook computers may be used to take notes, but not for answering email, browsing the Web, or other non-course related activities. No use of any other electronic or computer media is allowed during class time.

The following statements must appear on all SDSM&T course syllabi:

ADA Statement
Students with special needs or requiring special accommodations should contact the instructor (John Weiss, 394-6145) and/or the campus ADA coordinator (Jolie McCoy, 394-1924) at the earliest opportunity.

Freedom in Learning Statement
Under Board of Regents and University policy, student academic performance may be evaluated solely on an academic basis, not on opinions or conduct in matters unrelated to academic standards. Students should be free to take reasoned exception to the data or views offered in any course of study and to reserve judgment about matters of opinion, but they are responsible for learning the content of any course of study for which they are enrolled. Students who believe that an academic evaluation reflects prejudiced or capricious consideration of student opinions or conduct unrelated to academic standards should contact the dean of the college which offers the class to initiate a review of the evaluation.