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

Instructor: Dr. Zong
Office: McLaury 203C
Phone: 605-394-6034
Email: Ziliang.Zong@sdsmt.edu
Office Hours: Monday: 11:00 am – 1:00 p.m.
Wednesday: 11:00 am – 1:00 p.m.

Course Description: CSC 300 Data Structures (4 credits).
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; algorithm analysis; linked lists; stacks; queues; trees; graphs; heaps; sorting algorithms and hash functions.

Prerequisites: CSC 250 with a minimum grade of “C” and CSC 251.
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. 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, 2007)

Course Evaluation:
Grades for this course will be based on:
  1) Quizzes 5%
  2) Programming Assignments (4) 45%
  3) Exam 1 10%
  4) Exam 2 10%
  5) Exam 3 10%
  6) Final Exam 20%

and will be assigned as follows:

<table>
<thead>
<tr>
<th>Grade</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Score</td>
<td>90 — 100</td>
<td>80 — 89</td>
<td>70 — 79</td>
<td>60 — 69</td>
<td>&lt; 60</td>
</tr>
</tbody>
</table>
Topics
1) Introduction to algorithm analysis (1 week)
2) Simple data structures (linked lists, stacks, queues, binary trees) (2 weeks)
3) Advanced data structures (trees, heaps, graphs) (5 weeks)
4) Related algorithms (searching and sorting, hashing, graph algorithms) (5 weeks)
5) Advanced topics and applications (think in parallel, parallel programming, multi-core) (2 weeks)

More details about the topics will be posted at the course schedule webpage as the semester progresses: http://www.mcs.sdsmt.edu/zzong/csc300/schedule.html

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. To address the new challenges brought by the multi-core, advanced topics related to multi-core programming will also be introduced in this course. The course material will provide 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 the complexity of algorithms using big-O notation
• understanding of fundamental data structures such as lists, stacks, queues, 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 use both Windows and Linux operating system for software development
• ability to think in parallel for multi-core oriented programming
• greater understanding of the software development process
• experience working in teams
• communicate, in writing, how to analyze and/or solve a problem

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 or the Microsoft Visual Studio C++ compiler on Windows. Program grading policies will be further discussed in class when assignments are distributed. Please note that the new department policy requires that students must have a “C” or better grade in order to take higher level courses.
Software
CSC 300 is a programming-intensive course, so be prepared to spend many hours struggling with the computer this semester. You can use the GNU C++ compiler on the Linux operating system or the Microsoft Visual Studio C++ compiler on Windows for programming assignments. You should already have the Microsoft Visual Studio C++ installed in your Tablet and the Linux GNU C++ compiler is 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.

Attendance
If you do not attend class, it gives the instructor no opportunity to teach you the course material. Good attendance is particularly important in this class. Attendance is required for all class sessions. If you must miss a session for some important reasons, please email me your explanations at least two days before the session is scheduled and make sure that your absence has been approved, otherwise 1 point (up to 5) will be deducted from your final grade if you miss a class with a pop-up quiz.

Make-up policy
Make-up examinations will be given only if you contact me one week before the examination and provide a legitimate reason. There will be NO make-up exams without a verified excuse.

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 strictly forbidden under any circumstances whatsoever to exchange source code with your classmates and students who have previously taken this course. 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.

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.

ADA Statement
Students with special needs or requiring special accommodations should contact the instructor (Dr. Ziliang Zong, 394-6034) 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.