CSC 222 – Data Structures and Algorithms II

Course Description: (3h) A continuation of the study, analysis, and implementation of abstract data structures. The complexity of algorithms is studied more rigorously than in Computer Science 221 and complexity classes are introduced.

Prerequisites: Computer Science 221 and Mathematics 111

Professor: Dr. William Turkett
Manchester 240
758-4427
turketwh@wfu.edu

Office Hours: Any time the door is open on Monday through Thursday (I expect to be on campus most days from 9:00am to around 7:00pm). Guaranteed times are 3:00-5:00pm on TR. A calendar will be posted on my door with my updated schedule each week.

Meeting Time For Class: 2:00-2:50pm MWF, Manchester 244

Webpage: http://www.cs.wfu.edu/~turketwh/CS222/Fall2006/index.html - Please check the website frequently for updates concerning the class.


Grading:

- Three Tests (Mid September, Late October, Final) – 40%
- Programming Assignments – 25%
- Homeworks – 20%
- Reading Quizzes – 15%

Expected Grading Scale:

- A 92-100 A- 90-91
- B+ 87-89 B 82-86 B- 80-81
- C+ 77-79 C 72-76 C- 70-71
- D+ 67-69 D 62-66 D- 60-61
- F 0-59
**Attendance:**
Regular attendance in class is expected.

**Tests and Final Exam:**
There will be two tests during the semester to judge the student's progress in the course. These tests may include material from the readings, lectures, problem sets, and programming exercises. The final exam will be cumulative over the material for the entire semester, but with the major focus on the material from the last third of the semester. All tests and exams will be closed book. Make up tests will be allowed only if the absence is excused by the University.

**Reading Quizzes:**
The intended design of this course is such that the assigned reading (of usually < 10 pages per class meeting) will be complemented and extended by the lecture. Short reading quizzes will be posted on Blackboard. The purpose of these quizzes is to encourage reading of the assignments, to highlight key parts of the reading, and to allow you to provide me with feedback before the lecture. These should be submitted by midnight of the day before the lecture. I anticipate one reading quiz for every two days of lecture.

**Homework Assignments (“Problem Sets”):**
There will typically be a problem set assigned each week. For most weeks, problem sets will be due on Wednesdays and should be turned in by the start of the class on the due date. No late homeworks will be accepted.

**Programming Assignments:**
There will be 2-3 programming assignments over the course of the semester, primarily starting after the first test. These should be submitted via Blackboard by their assigned due date and time. No late programs will be accepted.

**Academic Integrity:**
All tests, programs, and homework should be done independently by each student. Copying of partial or complete work will not be tolerated and will be referred to the University Judicial System. Do not throw away or recycle any notes until the end of the semester. Should a question of authorship arise you will be expected to produce hand-written and printed documents that trace the development of your work. Algorithmic and electronic means of detecting violations may be used by the instructor on submitted assignments.

**Learning Assistance:**
If you have a disability that may require an accommodation for taking this course, please contact the Learning Assistance Center (758-5929) within the first two weeks of the semester.
University Closure:
In the event that the University closes due to pandemic or other disaster, you will be requested to read the book sections as outlined in the CSC 222 Lecture Plan document (to be passed out soon!). Powerpoint slides will be distributed providing appropriate lecture material to correspond with the readings. If time allows, corresponding videos highlighting the lecture (such as drawing out an algorithm’s steps) will also be provided. Reading quiz questions will be folded into the problem sets. The original programming exercises will be assigned at appropriate intervals. These assignments will be distributed via email or Blackboard if the Internet is available or via postal mail if it’s not. You should send electronic versions of your answers to the problem sets and programming exercises to William Turkett (turketwh(at)wfu(dot)edu, if the Internet is available, or 278 Glen Eagles Drive, Winston-Salem, NC 27104 if it is not). You will be provided with tests and finals through the means described above. These should be taken closed book, without access to papers, persons, or resources other than a calculator. The return dates for the examinations will be specified in the mailing. I will also provide you with a personal telephone number so that you can ask questions directly.
My Intended Goals For This Course:

- To provide the student with a foundational understanding of algorithmic complexity analysis.
- To provide an in-depth examination of the fundamental classes of algorithms to improve the student’s ability to determine the applicability of algorithms to different problems.
- To examine significant data structures not previously introduced in the student’s course work and algorithms associated with those data structures, as well as to reinforce the student’s understanding of data structures that have been presented before.
- To provide significant, practically-motivated programming problems that demonstrate the student’s understanding of the issues covered in goals 1-3 above.
- To improve the student’s skills in common programming environments.

Topics Covered (Tentative):

- **Analysis of Algorithms:**
  - Asymptotic Complexity ($O, \Omega, \Theta$)
  - Complexity Classes (P and NP, NP-Complete)
  - Approximation Algorithms *(if time allows)*
- **Review/Catch Up Of Material From CSC 221**
- **Major Algorithm Design Strategies:**
  - Types of Algorithms *(as time allows)*
    - Greedy
    - X and Conquer, where X => {Divide | Transform | Decrease}
    - Backtracking
    - Branch and Bound
    - Dynamic Programming
    - Iterative Improvement
  - Application of Algorithms To:
    - Advanced Data Structures
    - Important, Practical Problems
- **Proficiency With:**
  - Proofs (Correctness, Complexity, Induction, Idea of Reductions)
  - Programming Environments (C++, Editors, Debuggers, Make systems)