CSC 495 Class Information and Syllabus

Instructor: Stephen R. Tate (Steve)
Lectures: Tues/Thurs 2:00–3:15, Petty 007
Office: Petty 166
Office Hours: Wed/Fri 9:30 – 11:30, or by appointment
Phone: 336-256-1033
E-mail: `srtate@uncg.edu`

Prerequisites: CSC 330 or permission of instructor. Students who are currently enrolled in CSC 330 and are interested in taking this class are welcome to seek permission, assuming they have had excellent performance in CSC 130 and CSC 230, and have programming experience outside of just those two classes.

Brief Description: This class will cover computational problem-solving techniques. Part of the course will involve going through *Programming Pearls* by Jon Bentley, a classic and one of the best "problem solving with programming" books ever written. The rest of the course will revolve around solving programming problems similar to the challenges found in programming contests, identifying common structure and problem solving techniques. The problems that we consider will have two characteristics: They are hard to solve (if not impossible) without writing a program, and a programming solution exists that is relatively succinct (typically under 100 lines). This is not about system building or writing production software — it's about solving an immediate problem that you need an answer to.

Please note the emphasis on problem solving. This course is *not* about the mechanical aspects of programming, such as syntax or basic logic. You should already be fluent in a high-level programming language (preferably Java or C++). If you must refer to programming language references when you are writing programs, you will probably not be able to complete the work in this class.

Student Learning Outcomes: Upon successful completion of this course, students will have demonstrated an ability to

- write programs to solve advanced computational problems;
- analyze several alternative solutions to determine the best approach;
- justify that a program correctly solves a given problem.

An additional outcome, although difficult to measure and assess, is that students should build a better intuitive sense of problem solving through programming.

Class Web Page: `http://www.uncg.edu/cmp/faculty/srtate/495/`

Required Book: Jon Bentley. *Programming Pearls (second edition)*. ACM Press, New York, 2000.

Teaching Methods and Assignments: This class will meet twice per week, for 75 minutes each meeting. As this class if focused on using programming to solve problems, there will be a *lot* of programming. We will write programs in class, and you will have programs to write for homework every week. Class materials will all be in a "C-like" procedural language — the *Programming Pearls* book uses C-like pseudocode as well as, in places, real code in either C or C++. In-class programming will be done in either Java or C++, depending on the problem and the wishes of the class.

Students may program in any approved language that they are comfortable with, where "approved languages" include at least C, C++, Java, and Python. Student programs will be submitted using a web-based submission system, which will run some automated tests and provide some immediate feedback to the user. This system needs to be set up and tested for each approved language, so if you want to use something other than C, C++, Java, or Python, talk to the instructor as early as possible to see if it is possible. Programs will be graded based on both correctness and elegance (which includes style and algorithmic decisions).

In a typical week, there will be four programs assigned: two required programs that everyone is expected to do, and two challenge programs that are more difficult problems to solve and will earn "Ninja Points" (see below). There may be occasions where one of the required programs will be replaced by a written exercise (such as writing up a justification that a program correctly solves a problem).

Ninja Points: The required programs are the minimal expectations for this class, and students can earn a passing (but not good) grade by doing just those programs. There will be opportunities every week to go beyond the minimal expectations, by solving more challenging problems, doing extra activities, or performing tasks particularly well. These earn one of two types of "ninja points." *Base* ninja points are points that everyone can earn — for example, each challenge program can be done by everyone, and each solution provides a certain number of base ninja points. *Extra* ninja points are competitive — for example, there might be a competition for the fastest solution to a problem, with the winner or winners receiving extra ninja points. Part of your final grade depends on how many ninja points you earn during the semester: these are scaled so that if your total is at least half of the possible base ninja points, you will receive the full 20% of the final grade allocated to ninja points. A "scoreboard" of ninja point totals will be available on blackboard so that you can track your progress, and at the end of the semester, the student with the most ninja points will win a "Programming Ninja" T-shirt.

Final Challenge: At the end of the semester, in place of a final exam there will be a programming challenge. More details will provided on this later in the semester.

Evaluation and Grading: Each assignment will be labeled with the number of points that it will count, relative to other assignments. Scores will be combined to produce a final average according to the following weighting scheme:

Required Programs	65%
Ninja Points (scaled)	20%
Final Challenge	15%

Reading: As described above, each week will start with a discussion of one "column" from the *Programming Pearls* book. The table below gives the due date for each reading — you should read each column before this date, and make notes about questions you have and lessons you have learned. Read through the questions at the end of the chapter, and think about solutions (hints and solutions are available at the back of the book). Class discussion will not repeat the material from the reading, but will build upon it, so it is vital that you do the required reading before class!

Thursday		
	Jan 12	Column 1: Cracking the Oyster
	Jan 17	Column 2: Aha! Algorithms
	Jan 24	Column 3: Data Structures Programs
	Jan 31	Column 4: Writing Correct Programs
	Feb 7	Column 5: A Small Matter of
	Feb 14	Column 6: Perspective on Performance *and* Column 7: The Back of the Envelope
	Feb 21	Column 8: Algorithm Design Techniques
	Feb 28	Column 9: Code Tuning
	Mar 13	Column 10: Squeezing Space
	Mar 20	Column 11: Sorting
	Mar 27	Column 12: A Sample Problem
	Apr 3	Column 13: Searching
	Apr 10	Column 14: Heaps
	Apr 17	Column 15: Strings of Pearls

Academic Integrity: Students are expected to be familiar with and abide by the UNCG Academic Integrity Policy, which is online at `http://academicintegrity.uncg.edu/`

Assignments in this class are for individual work, unless explicitly stated otherwise. General concepts and material covered in the class may be discussed with other students or in study groups, but specific assignments should not be discussed and any submitted work should be entirely your own. Any incidents of academic dishonesty will be handled strictly, resulting in either a zero on the assignment or an F in the class, depending on the severity of the incident, and incidents will be reported to the appropriate UNCG office.

Attendance Policy: Attendance will not be taken in class, and is voluntary; however, all students are responsible for everything done or said in class (this can include changes in assignments, due dates, etc.). The sole source of information for much of the problem-solving techniques that are needed to solve the programming problems will be in-class discussions, and doing the required work without regularly attending class meetings would be very difficult. The university allows for a limited number of excused absences for religious observances. Students who plan to take such an absence should notify the instructor at least two weeks in advance so that accommodations can be made (also see the late work policy below). It is the student's responsibility to obtain notes from another student if they miss class.

Laptop/Cellphone Policy: Students are encouraged to bring laptops to class, if possible, and write programs along with the class. However, laptops should not be used for checking e-mail, chatting, or other entertainment during class time. This class has a strict "no non-class related use" rule for laptops — if you are found violating this policy, then your in-class laptop privileges will be taken away. Cellphones are a distraction for everyone, and should be turned off during class. If there is a special situation where you need to have your phone on for a particular day, please let the instructor know the situation before class.

Late Policy and Makeup Exams: Assignments are due at the beginning of class on the due date, and late submissions will not be accepted. Keep up with your work! Students with planned absences, whether for university events, religious observance, or other reason, are expected to make arrangements with the instructor to turn in work before the due date.

ADA Statement: UNCG seeks to comply fully with the Americans with Disabilities Act (ADA). Students requesting accommodations based on a disability must be registered with the Office of Disability Services located in 215 Elliott University Center: (336) 334–5440.