

CS 6463 Fundamentals of High Performance Optimization, Fall 2007

Instructor: Dr. Whaley (whaley@cs.utsa.edu)
Office: SB 4.01.12
Office Hours: MW 3-4PM
Office Phone: 458-5545
Class homepage: www.cs.utsa.edu/~whaley/teach/cs6463FHP0/
Class hours: MW 5:30-6:45PM
Classroom: HSS 3.04.26

Textbook none.

Prereq CS 5513 or 4753 Computer Architecture & CS 2413 Systems Programming or equivalent. We will be programing in a Unix environment, using ANSI C, assembly, and make.

Grading There are two grading schemes available for this course. In the default scheme, the majority of grade comes from actual programming assignments, whose purpose is to extend your understanding of the topics under discussion. If oral/written quizzes show inadequate student preparation, or if homework is of low quality or exceedingly uniform, I will be forced to rely more heavily on testing to ensure that students are understanding the material, as shown in the test-based grading scheme. If this occurs, I will announce the change to the class, and the written testing will begin in the following week.

Default grading scheme:

- 85% of grade from projects, including final project:
 - 0-2 presentations per student
 - Rest determined by extensive programming projects
- 10% oral/written quiz
- 5% class participation
- No written final (final project instead)

Test-based grading scheme:

- 60% prog projects, including final project
- 35% written exams, including final exam
- 5% class participation

Students are expected to be able to fully explain the workings of their own programs, and may be called upon to do so. If they cannot, no credit will be given for that assignment.

Attendance You are responsible for all material presented in class. Exams and due dates will be scheduled in advance. A grade of zero will be recorded for missed exams unless prior arrangements are made (only allowed in extraordinary circumstances). Assignments turned in after the due date, but before the beginning of the next scheduled class will be penalized 10%. Assignments will not be accepted that are more than one class period late (resulting in a grade of zero).

Cheating Students are encouraged to discuss programs in a general way to gain greater insight. Copying another's code, writing code for someone else, or allowing another to copy your code are cheating, and can result in a grade of zero for all parties.

Therefore, take precautions so that your old printouts, unattended screen, etc. are not available to other students. In general, no student should ever show another student in the class even a section of his/her code. If you are in doubt whether an activity is permitted collaboration or cheating, ask the instructor.

- Decorum** Students are expected to refrain from side conversation or other distracting behavior in class. Students should arrive on time for class; if late, come in quietly with a minimum of disturbance. All cell phones/pagers/PDAs/etc. should be turned OFF before the beginning of class, and not be consulted in any way during class. During testing, any such consultation may result in a grade of zero. Violations of this policy will minimally result in expulsion from the classroom, and repeated violation will result in expulsion from the course.
- Email** Questions about lectures, homework and course organization may be sent to the instructor. I cannot guarantee an immediate response, but will address the issue through direct response, general announcement, or a suggestion to visit during office hours. Last minute questions (i.e. sent the night before an assignment is due) may not be answered before class begins, so tackling problems early is encouraged.
- Regrading** If you believe I have made an error in grading your exam or assignment, you may submit the graded work along with a written request for reconsideration. You must explain in writing clearly and succinctly the reasons your grade should be changed. In fairness to other students, I cannot vary the grading criteria on an individual basis, though suggestions may be taken into consideration for future classes.
- Objectives** This course will cover various areas of computer science whose understanding is required to tune performance-critical kernels in the real world.
- Material** Topics of interest include (but are not limited to): (1) Complexities of getting reliable and context-sensitive timings on actual hardware in the real world, (2) Overviews of several architecture areas critical to optimization, including memory hierarchy, pipelining, superscalar exploitation, etc., (3) Application of traditional optimizations on various architectures including: software pipelining, superscalar scheduling, scalar expansion, blocking/tiling, and various array optimizations (contiguous vs. strided vs. pointer to pointer access, TLB issues, etc.), (4) Basics of assembly programming (particularly x86) – necessary to do extremely low-level optimization, including SSE, (5) Exploiting low-level architectural features, including SSE and prefetch, (6) Basics of IEEE floating point arithmetic and storage
- Disability** If you have a physical, psychological, medical or learning disability that may impact on your ability to carry out assigned course work, I would urge that you contact University Disability Services (DS), Multidisciplinary Studies Building, Room 2.03.18, 210-458-4157 (Voice), 210-458-4981 (TTY), 210-458-4980 (Fax), homepage: <http://www.utsa.edu/disability/>. Please bring a letter to me from the DS indicating your need for academic accommodations within the first week of class. The syllabus and other class materials can be made available in alternative format upon request.