COMPUTER SCIENCE 50100
Computing for Science and Engineering
FALL 2011
COURSE SYLLABUS
August 23

Course description: Computational concepts, tools, and skills for computational science and engineering: scripting for numerical computing, scripting for file processing, high performance computing, and software development. Project may be required. Credit in this course may not be used toward a graduate degree in Computer Science. Prerequisites: C S 15900 or 15800, MA 26200.

Role of course: This is a core course in the CS&E Program. The purpose of the course is to expose students to computational concepts, tools, and skills useful for research in computational science and engineering, beyond what is learned in a first programming course (and basic mathematics courses). A related aim is to prepare students for other courses in CSE core by teaching CS material mostly unavailable in existing graduate level courses. The following CSE core courses are probably accessible after taking this course: CS 51501, CS 52500, CS 53000, CS 54100. The following are probably accessible before taking this course: CS 51400, CS 51500, CS 52000.

Prerequisites:

- mathematical knowledge and maturity of an MS student in the physical sciences. (It is not assumed that the student is in the physical sciences, only that he(/she) has this level of mathematics.) In particular, some familiarity with matrix algebra and elementary probability is expected.
- programming experience in C, C++, Java, or Fortran or extensive scripting experience; also, commensurate computer skills.

Sister course: The course CS 59000-CLS, Computing for Life Sciences, is designed for students from Biological Sciences, Agriculture, and Pharmacy. It meets 11:30 am–12:20 pm MWF in Lawson Computer Science Bldg 1106. The instructor is Professor Daisuke Kihara.

Staff:

Instructor	**Teaching Assistant**
Robert Skeel	Bala Gnanasekaran
Lawson 1205	Lawson B116E (#2)
(765)494-9025	
fax (765)494-0739	
`skeel at cs dot`	`bgnanase at`
Office hours:	
Mondays 1:00-1:30	
Tuesdays	∗1:30–2:30
Wednesdays 1:30–2:00	11:00–noon, ∗noon–1:00

∗ only those weeks that an assignment is due

Web page: The class web page is `http://www.cs.purdue.edu/homes/skeel/cs501.html` . This will be neglected in favor of Blackboard Vista, which will also be used for grade records and turning in homework.

Textbook: There is no suitably comprehensive textbook. Notes will be distributed having links to web resources. A good reference is Software Carpentry. Some useful books:

- Python in a Nutshell, 2nd Edition
  by Alex Martelli, O'Reilly, July 2006,
- Learning Python, 4th Edition
  by Mark Lutz, and David Ascher, O'Reilly, September 2009,
- Python Scripting for Computational Science, 3rd Edition, 2nd Printing
  by Hans Petter Langtangen, Springer, 2008,
- The C Programming Language, 2nd Edition
  by Brian W. Kernighan and Dennis M. Ritchie, Prentice-Hall, 1988, ISBN 0-13-110362-8,
- Introduction to High Performance Scientific Computing
  by Victor Eijkhout, 2010. Downloads are free, and the printed copy is very affordable.
- An Introduction to Parallel Computing: Design and Analysis of Algorithms, 2nd Edition
  by Ananth Grama, George Karypis, Vipin Kumar, and Anshul Gupta Addison-Wesley, 2003, ISBN: 0-201-64865-2.

MIT Open CourseWare: Here are the most relevant ones in computer science:

- elements of software construction, except for the use of Java,
- introduction to algorithms,
- computer system engineering,
- practical programming in C,
- performance engineering of software systems,
- a gentle introduction to programming using Python, 2011,
- a gentle introduction to programming using Python, 2008,
- multicore programming primer,
- applied parallel computing.

Examination schedule

Thursday, **Oct 6** 1st and 2nd exam, 3:00-4:15 pm,

Thursday, **Nov 10** 3rd and 4th exam, 3:00-4:15 pm,

Dec 12–17 5th, 6th, and final exam, 2 hours

The lowest score of exams 1–6 will be dropped.

Anticipated course outline

Theme I: scripting to get things done

August introduction to CS&E, Unix, software installation

September file/text processing and mathematical computing in Python

Theme II: programming for performance

October C programming

November parallel computing: MPI, OpenMP

Theme III: developing software

December developing software: OOP, GUIs, software engineering

Policies

a. *Copying* is the *evident* use of outside material in your solution (anything not present in lectures and handouts). Outside material includes everything except material composed by the instructor. In the case of collaboration or use of outside written material, the score for an assignment may be discounted. For example, 3 students turning in identical correct answers each may receive as little as one third credit. The course staff reserve the right to determine the discount, if any. Use of your work by another will be assumed to be done with your consent unless there is evidence to the contrary.

b. *Cheating* is the evident use of outside material in your solution *without including a written acknowledgment of the source.* (Citing sources is simply good scholarship.) This does not preclude some discussion when starting an assignment. However, the onus is on you to either acknowledge the collaboration or ensure that it is sufficiently limited so as not to be evident in your work.[1] *Cheating is penalized in accordance with university rules and the instructor's policy is to impose substantial penalties.*

c. Excuses. Failure to attend an exam or failure to submit an assignment on time is recorded as a zero except when it is unavoidable because of some legitimate emergency (a medical excuse from a physician, or a death in the immediate family). In such a case it is desirable that the instructor be contacted before the exam or before the due date of the assignment.

   The teaching assistant may grant you an extension for an assignment, but he has the prerogative to refuse you.

d. Requests for assignment regrades should be submitted to the TA within one week of it being returned to the class. If you remain dissatisfied after the regrade, you may appeal to the instructor.

e. No extra work will be provided for students wishing to improve their grade. Also, final grades will not be changed after they are posted (except if they were incorrectly computed).

[1] In practice, this is enforced when the course staff believe that the collaboration goes beyond the sharing of ideas. but again the course staff reserve the right to make this judgment.

Assignments There are about six assignment sets, much of it requiring the writing of programs. Tentative due dates are Sep 8, Sep 22, Oct 13, Oct 27, Nov 17, and Dec 1. The plan is to drop the lowest score. There is no project this semester. We will use Python and C. There will be considerable computer use. Ideally, the student already has access to a multicore computer for his/her research and will use this for the assignments. The student will be expected to install the necessary (free) software on his/her desktop. As an alternative, accounts on an ITaP/RCAC multiprocessor are provided. One way or another, we will attempt to accommodate a variety of platforms. In any case, we will ask you what platform (hardware and system software) you are using.

Grading

assignments	35%	A	80–100%	D	40–50%
in-class exams	30–40%	B	65–80%	F	0–40%
final-week exams	25–35%	C	50–65%		

Plus and minus grades will also be given. The grade boundaries may be lowered by as much as 10% in order to achieve a reasonable grade distribution. Also, there may be minor upward or downward adjustments of the grade boundaries.

Plans for potential campus interruptions In the event of a major campus emergency, course requirements, deadlines and grading percentages are subject to changes that may be necessitated by a revised semester calendar or other circumstances. Here are ways to get information about changes in this course: The primary method of communicating with students is Blackboard Vista. The course website might be used a backup, as might email.

ASSIGNMENT # 0 The following actions are required:

1. Send an email to the TA stating precisely what operating system you are using. In the case of Windows, state whether or not you will be using cygwin. If you will use more than one platform, so state.

2. Install items 1–6 below. Consult Section 1.3 of the class notes for assistance and advice.

Installation of Software

The following is required:

1. a C compiler that supports OpenMP, e.g., version 4.2 or later from `www.gnu.org`. Test with the program `ohello.c` in the class notes.

2. MPI. Presumably this is already installed. Test with the program `hello.c` in the class notes.

3. Python 2.5.2 or later but no later than 2.7.

4. NumPy 1.3.0 or later.

5. MPI for Python (mpi4py) 1.0 or later.

6. a plotting program, e.g., Matplotlib.

The following is recommended:

7. IPython.