ECE4893/8893 - Advanced Programming Techniques

Instructor

Dr. George F. Riley
Office: Klaus 3360
Office hours: Tue/Thu 10-12noon, other times by email
Email: riley@ece.gatech.edu

Course Summary

The "Advanced Programming Techniques" course will cover a number of advanced topics in programming methods, data management, distributed computing, and advanced algorithms used in typical engineering applications. All class projects and in--class examples will use the C++ programming language. It is designed to be a 4000 level course cross listed with a 6000/8000 level course, taken by both advanced undergraduate and beginning graduate students. The undergraduate and graduate versions will meet in the same room at the same time, and graduate students will be expected to complete two or three additional assignments as compared to the undergraduate students. The format of the class is two 1-hour classroom lectures per week, where the new topic is introduced, and suggestions of how to go about implementing the topic in C++ is discussed. The students will have unsupervised lab to work on the programming projects and complete the assignments. The topics are diverse, and each could merit its own course. Instead, this course will cover each topic from a conceptual standpoint, and discuss in some detail a small number of specific instances of the programming techniques used to implement programs using that topic. One programming assignment for each topic will be provided to give students practical experience in each topic, and to improve the students overall programming skill via substantial practice in coding and debugging.

Tentative Topics

The list of topics to be discussed is tentative, but likely to include:

- Distributed programming with MPI (2 or3 lectures)
- Parallel programming with pthreads (3 lectures)
- Introduction to graphics programming using OpenGL (3 lectures)
- Object--Oriented code templates (2 lectures)
- Event--based Programming (2 lectures)
- Using web services (3 lectures)
- Using non--blocking system I/O (2 lectures)
- Discrete Event Simulation (2 lectures)

- Introduction to database programming using MYSQL (2 lectures)

Teaching Philosophy

Teaching is interactive! Students are strongly encouraged to participate in class and offer opinions on the issues being discussed. I encourage (and expect) you to participate actively in the learning process. In particular, I welcome your comments and questions as we cover material in class. One-way lectures quickly becoming boring, both for you and for me. Also, I have found that students often learn more from other students comments than from the instructor! By asking lots of questions, your understanding of the material will be deepened significantly, and the course will be much more fun! From time to time there will be readings for a class session; these will be posted on the class web page below in a downloadable format. Students are expected to download and read the assigned readings before class.

Policy for Completing Out-of-Class Assignments

We will have programming assignments every week or two weeks, excepting weeks when there is an examination. The policy of completing these assignments is clear and simple. **All students must personally and with their own two hands design, implement (type in) and debug their programs.** Two or more students " Working Together " and turning in one program (or copies of the program) is **not acceptable.** However, students are very much encouraged to seek help when it is needed. You can get help from anyone, including the instructor, teaching assistants, and fellow students. You can ask for help with debugging, help with how to formulate a solution, and help with the syntax of the C/C++ program. However, to be clear, each student must personally type in, compile and debug their own program. Cutting and pasting from other solutions is **not acceptable.**

Computing Resources

We have a newly installed linux-based computing cluster known as the Jinx cluster. This platform is the recommended platform for completing all of the class projects, although you are free to use your personal laptops or desktops, as long as they have the appropriate tools. The instructor and/or TA will grade the assignments using the jinx cluster.

Textbook

There is no textbook for this class. We will use research papers and handouts as required for our reading and discussion.

Getting Help

Students are encouraged to get help from either their fellow students or the instructor. However, when getting help from students be sure to adhere to the policy for completing out-of-class work as above.

- Teaching Assistant: TBD tbd3@mail.gatech.edu
- Office Hours TBD
- TA Office hours across from Klaus 3360

Grading

Programming Projects	50%
Midterm Exam	20%
Final Exam	20%
Class Participation	10%
Total	**100%**

Grading (Distance Learning Students)

Programming Projects	50%
Midterm Exam	25%
Final Exam	25%
Total	**100%**

Syllabus

Day	Month	Date	Description	Handout	Due Date
Tue	Aug	20	MPI Tutorial	https://computing.llnl.gov/tutorials/mpi/	
			Simple Blocking MPI Program	testMPI.cc	
			Simple Non-Blocking MPI Program	testMPI2.cc	
			MPI Programs(pdf)	mpi-examples.pdf	
				AccessingJinx.pdf	
Thu	Aug	22	MPI Continued		

			Simple Barrier and Gather MPI Program	testMPI3.cc	
			Non-Blocking receive example	testMPI4.cc	
			MPI Programs (revised pdf)	mpi-examples.pdf	
			2D Fourier Transform Assignment	fft2d.pdf	
Tue	Aug	27	MPI and FFT continued		
Thu	Aug	29	PThreads Example	PthreadsExample.pdf	
			ThreadedCount.cc	ThreadedCount.cc	
			simpleThread.cc	simpleThread.cc	
			PThreads tutorial	https://computing.llnl.gov/tutorials/pthreads/	
Tue	Sep	3	pthreads continued		
Thu	Sep	5	Implementing a Barrier	Barriers-handout.pdf	
			Leslie Lamport's Bakery Algorithm	Bakery-handout.pdf	
Tue	Sep	10	2D Threaded DFT Assignment	dft2d-PThreads.pdf	
Thu	Sep	12	Discussion of mutexes (Bakery Algorithm) and barriers		
Tue	Sep	17	Templates	TemplateIntroduction-handout.pdf	
			templateintroduction.cc	templateintroduction.cc	
Thu	Sep	19	Templated LinkedList	TemplateLinkedList-handout.pdf	
			templatedlinkedlist.cc	templatelinkedlist.cc	
Tue	Sep	24	Mid-Term Exam		
Thu	Sep	26	Templated Vector Assignment	TemplatedVector.pdf	
Tue	Sep	31	Templated Vector (continued)		
Thu	Oct	3	Templated Vector (continued)		
Tue	Oct	8	Smart Pointers	PtrExample.pdf	
Thu	Oct	10	Smart Pointers (continued)		
Tue	Oct	15	Fall break NO CLASS		
Thu	Oct	17	OpenGL Documentation	http://www.opengl.org/sdk/docs/man2/	
			circle.cc	circle.cc	
			circle-complete.cc	circle-complete.cc	
			Makefile-OpenGL-Jinx	Makefile-OpenGL-Jinx	
Tue	Oct	22	OpenGL Continued		
			OpenGL Jet Image assignment	Jet.pdf	
Thu	Oct	24	Discussion of OpenGL Jet assignment		

			Icosahedron Solution	icos.cc	
			Using the GNU Debugger	gdb-refcard.pdf	
Tue	Oct	29	Multi-precision arithmetic	gmp-man-5.0.2.pdf	
			Original RSA article from Scientific American 1977	Rsapaper.pdf	
			RSA Encryption algorithm	RSA-handout.pdf	
			Diffie-Hellman algorithm	diffie-hellman_key_exchange.pdf	
			Factorial Implementation	factorial.cc	
			Compute Pi Implementation	gmp-chudnovsky.c	
Thu	Oct	31	Non-Blocking Input Output with select	ChatExample.pdf	
			Chat client	chat.cc	
			Chat server	chatserv.cc	
			RSA Encryption assignment	RSA.pdf	
Tue	Nov	5	Model/View/Controller Design Pattern	MVC-handout.pdf	
			TicTacToe Class Declarations	ttt.h	
			TicTacToe Class Implementations	ttt.cc	
			TicTacToe main program	TicTacToe.cc	
Thu	Nov	7	Discussion of Mandelbrot Set project	MBSet.pdf	
			Interesting Mandelbrot Set Video	http://vimeo.com/12185093	
Tue	Nov	12	Introduction to Cuda	CudaOverview.pdf	
			CUDA Mandelbrot Set Assignment	MBSet.pdf	
Thu	Nov	14	Using Makefiles		
			Makefile1	Makefile1	
			Makefile2	Makefile2	
			Makefile3	Makefile3	
			Makefile4	Makefile4	
			Makefile5	Makefile5	
Tue	Nov	19	NO CLASS Instructor out of town		
Thu	Nov	21	The STL Sorted Containers	MapSet-handout.pdf	
			map-set.cc	map-set.cc	
Tue	Nov	26	Discrete Event Simulation	des-simple1.cc	
Thu	Nov	28	NO CLASS Thanksgiving holiday		

Tue	Dec	3	Discrete Event Simulation continued	des-simple3.cc	
Thu	Dec	5	Interprocess Communication with Shared memory	ShmFork-handout.pdf	
Thu	Dec	12	FINAL EXAM	11:30am to 2:20pm	

Contact Information:

riley@ece.gatech.edu
School of Electrical and Computer Computing
Georgia Institute of Technology
Atlanta, GA 30332-0250

Last Modified: 03:16PM, 04 December 2013.

ECE New Course Request

COURSE OBJECTIVES AND OUTCOMES

Core undergraduate courses (required courses or those that satisfy specific degree requirements such as probability/statistics or senior lab elective) MUST have course educational objectives and outcomes defined when submitted for permanent catalog listing. Courses proposed for satisfying certain degree requirements (e.g., senior lab electives) may be required to include certain objectives and/or outcomes. ***The following page provides detailed instructions and examples.***

20. Course Educational Objectives (Maximum of 6 objectives; maximum of 150 character each)

In brackets at the end of statement, identify the Student Outcome(s) to which that objective is contributing.

As part of this course, students …

1. Become familiar with various methods for concurent and distributed programming methods
2. Program complex engineering applications in the C or C++ porgramming language
3. Become familiar with three-dimensional graphics library interfaces
4. Implement advanced encryption techniques using multi-precision math libraries
5. Both create and use the popular "Smart Pointers" approach for memory management in C++ programs.
6. Program client-server applications using non-blocking system I/O calls

21. Course Educational Outcomes (Maximum of 15; maximum of 150 character each)

Upon successful completion of this course, students should be able to …

1. Determine when to use distributed computing methods or parallel computing methods to solve complex entineering applications
2. Create high-quality visual 3-D images of complex objects using the OpenGL graphica interface
3. Implement several multi-precision public key and private key encryption methods using the GNU multi-precision math library.
4. Create programs without memory leaks using the "Smart Pointers" approach
5. Implement client /server applications using the sockets API and using the nonblocking approach for handling multiple clients simultaneously.
6. Manage large programming tasks using Makefiles

7.

8.

9.

10.

11.

12.

13.

14.

15.