CSE Curriculum Committee

Agenda Summary
Dec. 7, 2010

Full agenda is on the web site: http://www.aem.umn.edu/~shield/csecc/

1. Approval of Sept. 21, 2010 meeting Minutes – see web site.
2. Meeting Schedule for Spring 2011 (watch web site for updates)
    - 2:30 on Tuesday 2011-2-1
    - 2:30 on Tuesday 2011-4-19
3. Items for Information only (already approved in ECAS):
    a. See web site
4. Items for Approval without Objection (already approved in ECAS):
    a. BBE 4404 – Bio-based Composites Engineering: Rename to Biopolymers & Biocomposites Engineering
    b. CHEM 3501/3502 (Intro to Therm, Kin, Stat Mech/Intro to Quantum, Spectroscopy): Deactivated (replaced by next two courses)
    c. CHEM 5501 – Intro to Thermo, Kin, Stat Mec: Renumbered to 4501
    d. CHEM 5502 – Intro to Quantum, Spectroscopy: Renumbered to 4502
    e. GEO 4402 – Biogeochemical Cycles in the Ocean: prerequisite change:
        i. New: [Chem 1021, Chem 1022] or #
        ii. Old: [Chem 3501, 1 yr of calculus] or #
    f. GEO 5980 Seminar: Current Topics in Geology and Geophysics: New Course
5. Action Items (new course syllabi are below or separate handouts):
    a. New Course: BBE 4012 (see syllabus below)
    b. New Course: CSCI 1910H (see syllabus below)
    c. New Course EE 4163 (see handout for syllabus)
6. New Business
7. Adjourn

New Courses Syllabi

BBE 4012 Transport in Biological Processes I

Credits: 4

Prerequisites: (MATH 2243 or 2373), (MATH 2263 or 2374), BBE 3001, PHYS 1302W

Course Format: 3 Lectures and 1 Lab per week.
Course Grading: Homework (20%), Two hour exam (20% each), Finals (35%), Class Participation (5%).

Textbook(s) Any one of the three
1. Fundamentals of Fluid Mechanics by Munson, Young, Okiishi, and Huebsch Sixth Ed. John Wiley and Sons.
2. Engineering Fluid Mechanics by Crowe, Elger, Williams, and Roberson. Ninth Ed. John Wiley and Sons.
3. Fluid Mechanics, Fundamentals and Applications by Yunus Cengel and John M. Cimbala. Second Ed. McGraw Hill.

Course Objective: Develop an understanding of fluid dynamics in biological engineering and related fields. Use of control volume analysis to develop basic equations and to solve problems is emphasized. Understand the concept of Newtonian and non-Newtonian viscosity. Understand and use differential equations to determine pressure and velocity in various flow systems. Determine losses in flow systems. Learn to use dimensional analysis to design physical or numerical experiments and to apply dynamic similarity.

Course Description: An introductory course in fluid mechanics. Fluid statics and kinematics. Differential and finite control volume analysis with continuity, momentum, and energy equations. Bernoulli and Euler Equation. Dimensional analysis. Potential flow. Non-Newtonian Fluids.

Week 1 Introduction, Classification of Fluid Flows, System and Control Volumes, Units and Dimensions, Properties of Fluids
Week 2 Properties of Fluids, Pressure, Introduction to Fluid Statics, hydrostatic forces on submerged plane surfaces, hydrostatic forces on submerged curved surfaces.
Week 3 Buoyancy and Stability, Fluids in rigid body motion, Fluid kinematics, Lagrangian and Eulerian descriptions, Flow pattern and visualization (streamlines, streamtubes, pathlines etc.), Vorticity and Rotationality
Week 4 Reynolds Transport Theorem, Conservation of Mass, Mechanical energy and Efficiency, Bernouli's Equation, General Energy Equation, Analysis of Steady Flow.
Week 5 Exam # 1, Introduction to Control Volume analysis, Linear Momentum Equation, Review of Rotational Motion and Angular Momentum.
Week 6 Angular Momentum Equation, Constitutive Equations to describe Newtonian and non-Newtonian Fluids.

Week 7 Laminar and Turbulent Flow, Entry Region, Losses in Pipe flow, Piping Network and Pump selection.
Week 8 Navier Stokes Equation, Solution to Momentum Equation for fluids having different constitutive equation.
Week 8 Solution to Momentum Equation for fluids having different constitutive equation, Creeping Flow, Irrotational Flow, Boundary Layer Approximation.
Week 9 Dimensional Homegeneity, Dimensional Analysis and Similarity, Buckingham ï□° theorem, Modeling and Similarity
Week 10 Classification of Open Channel Flow, Froude Number and Wave Speed, Uniform flow in Channels, Hydraulic Cross Section.
Week 11 Exam # 2, Gradually Varied Flow, Rapidly Varied Flow, Hydraulic Jump
Week 12 Turbomachinary, Pumps, Pump Scaling Laws
Week 13 Turbines and Turbine Scaling Laws
Week 14 Compressible Flow, Mach #, Isentropic and Nonisentropic Flow.

CSci 1901H - Honors Structure of Computer Programming

Credits: 4

Prereq - &(MATH 1371 or MATH 1571H) and honors student and permission of University Honors Program;

Equiv: CSCI 1901

Text: Abelson and Sussman, Structure and Interpretation of Computer Programs, 2nd edition McGraw Hill Book Company, 1996. The textbook is also available for free online.

Instructor contact & student workload:
3 hours of lecture per week.
2 hours of lab per week.
About 6 hours outside of these per week.

Grading:
10% In-class Exercises
10% Midterm Exam 1
15% Midterm Exam 2
20% Final
5% Quizzes
20% Labs
10% Homeworks
10% Final Project
Using the above weights, a total of 90% and up will earn you some level of A, 80% and up at least some level of B, 70% and up at least some level of C, 60% and up at least a D.

Class Webpage: All future handouts, assignments, announcements, and any additional material will be available through the 1901H class web page in the directory http://www.cselabs.umn.edu/classes/

Learning Goals: CSci 1901/1901H is a required course for computer science and computer engineering students that students should take in their freshman year. It is one of the courses required for admission to the computer science major, and is the first class in the sequence for majors.

1901/1901H offers an introduction to the fundamental principles of programming and to different programming paradigms, with emphasis on the design of abstract data types and recursive algorithms.

Computer science and engineering students need to acquire the reasoning and abstraction skills needed for designing algorithms and programs. This course teaches how to think as a computer scientist, by teaching the process of building abstractions to hide implementation details, of decomposing problems into simpler problems, and of controlling the intellectual complexity of designing large software systems.

1901/1901H does not assume any previous programming knowledge; however, it does have a co-requisite of Calculus I. This means you should either have completed Calculus I successfully, or should be taking Calculus I. Some material from Calculus I, such as differentiating polynomials, may be used in 1901/1901H; moreover, the mathematical and logical reasoning skills used in Calculus I also play a heavy role in this class.

Upon completing this course you should be able to

1. explain what a computational process is and be able to express computational processes using procedures in the programming language or languages used in this course.
2. explain what an abstract data type is, how to create new abstract data types, and how to express them in Scheme or in other programming languages used in the course.
3. use recursion as a problem solving method.
4. explain how the process of evaluation is performed from the perspective of the computer and how procedural objects are created and manipulated.
5. explain and use basic principles of program design, such as abstractions to hide implementation details, and problem decomposition to control the intellectual complexity of the problem.
6. explain how to use different programming styles (functional, object-oriented, data-driven).
7. use good programming style in the programs you write.
8. design appropriate data structures and algorithms to solve a given problem, using recursion whenever appropriate.
9. design, implement, and test a computer program that solves a given problem, and have some basic understanding of the computational complexity of your program.

Honors Class: This is the honors version of CSci 1901. It will therefore cover the usual 1901 material such as recursion, abstractions, modularity, and data structures. However, as the honors version it will also allow looking at some of the usual 1901 topics in more detail, and including

some additional material not in the usual 1901 class. Examples of possible additional topics include, but are not limited to: robotics, social aspects of computer science, additional programming paradigms, and programming language design and implementation.

Labs: You are required to use the MIT software package, which is available on the ITLabs computers. You may install MIT on your own computer but the TA will not be able to assist you with managing your personal system. It is your responsibility to make sure your programs work on the ITlabs machines on which the TA will grade your programs. Labs will be graded on correctness, completeness, and style. Correctness and completeness refer to how well the program works. Style includes good design, readability (indentations, descriptive names for variables and procedures, and appropriate use of blank spaces), and useful comments. You are expected to comment (include input, output, assumptions, etc. and will be discussed in class/discussion) each of your function. Testing your program for correctness is very important and you are expected to submit your own test cases along with the results. You should include test cases that verify where your code works as well as cases that show where it fails.
Lab programs you write must follow the specific instructions on each lab description.

Scholastic Conduct: Although you are free to discuss assignments with others, the work you submit for grading must represent your own efforts only. Exams are closed book and note and are to be completed using only your own knowledge of the course material. The experience gained from the labs will be very helpful for the exams and future labs may build on previous ones, so put in the effort needed to fully understand the solutions. Cheating on quizzes or exams is a serious offense, and will be dealt with as such. Additionally, labs are done in groups of two, but collaboration with others outside your group of two is prohibited. Homework assignments are individual efforts. You may discuss (in a general way) labs and homework problems with others, but you may not collaborate by writing code or specific solutions with others (with the exception of your lab partner in the case of labs). Copying other's answers, or letting another person copy your answers (either intentionally or as a result of negligence) is a serious situation and can result in failing the course. For further information on academic misconduct please see http://www-users.cs.umn.edu/ barry/intro/acad-conduct.html. If you have any questions about what is and is not allowable in this class, please ask the course instructor.

Incompletes: Incompletes will be given only in very rare instances when an unforeseeable event causes a student who has completed all the coursework to date to be unable to complete a small portion of the work (typically the final assignment or exam). Incompletes will not be awarded for foreseeable events including a heavy course load or poorer-than-expected performance. Verifiable documentation must be provided for the incomplete to be granted, and arrangements for the incomplete should be made as soon as such an event is apparent.