COMPUTER SCIENCE (DIV III)

Chair: Associate Professor BRENT HEERINGA

Professors: D. BAILEY, A. DANYLUK, S. FREUND, W. LENHART***, T. MURTAGH. Associate Professors: J. ALBRECHT, B. HEERINGA, M. MCGUIRE. Visiting Assistant Professors: W. JANNEN, J. PARK. Visiting STINT Fellow: J. BOYE§

Computers and computation are pervasive in our society. They play enormously important roles in areas as diverse as education, science, business, and the arts. Understanding the nature of computation and exploring the great potential of computers are the goals of the discipline of computer science. A sample of the areas of research investigated by the Williams Department of Computer Science alone illustrates the vast range of topics that are of interest to computer scientists and computing professionals today. This includes: the use of computer-generated graphic images in the arts and as a tool for visualization in the sciences and other areas; the protocols that make transmission of information over the Internet possible; the design of revolutionary new computer languages that simplify the process of constructing complex programs for computers; the development of machine learning algorithms that can extract useful and even novel information from data that is too complex for humans to analyze; algorithms that can solve problems that were previously too hard to solve in a reasonable amount of time, just by giving up a little bit of optimality in the solution; the investigation of machine architectures and specific hardware aimed at making computing fast.

The department recognizes that students' interests in computer science will vary widely. The department attempts to meet these varying interests through: (1) the major; (2) a selection of courses intended for those who are interested primarily in an introduction to computer science; (3) recommended course sequences for the non-major who wants a more extensive introduction to computer science in general or who seeks to develop some specific expertise in computing for application in some other discipline.

MAJOR

The goal of the major is to provide an understanding of algorithmic problem solving as well as the conceptual organization of computers and complex programs running on them. Emphasis is placed on the fundamental principles of computer science, building upon the mathematical and theoretical ideas underlying these principles. The introductory and core courses build a broad and solid base for understanding computer science. The more advanced courses allow students to sample a variety of specialized areas including graphics, artificial intelligence, computer architecture, networks, compiler design, and operating systems. Independent study and honors work provide opportunities for students to study and conduct research on topics of special interest.

The major in Computer Science equips students to pursue a wide variety of career opportunities. It can be used as preparation for a career in computing, for graduate school, or to provide important background and techniques for the student whose future career will extend outside of computer science.

MAJOR REQUIREMENTS

Required Courses in Computer Science

A minimum of 8 courses is required in Computer Science, including the following:

Introductory Courses

Computer Science 134 Introduction to Computer Science *or* Computer Science 135 Diving into the Deluge of Data
Computer Science 136 Data Structures and Advanced Programming

Core Courses

Computer Science 237 Computer Organization
Computer Science 256 Algorithm Design and Analysis
Computer Science 334 Principles of Programming Languages
Computer Science 361 Theory of Computation

Elective Courses

Two or more electives (bringing the total number of Computer Science courses to at least 8) chosen from 300- or 400-level courses in Computer Science. At least one of these must be a course designated as a PROJECT COURSE. Computer Science courses with 9 as the middle digit (reading, research, and thesis courses) will normally not be used to satisfy the elective requirements. Students may petition the department to waive this restriction with good reason.

Required Courses in Mathematics

Mathematics 200 Discrete Mathematics
and any other Mathematics or Statistics course at the 200-level or higher

Students considering pursuing a major in Computer Science are urged to take Computer Science 134 or 135 and to begin satisfying their mathematics requirements early. Note in particular that Discrete Mathematics covers material complementing that in the introductory courses (Computer Science 134/135 and 136) and is a prerequisite for many advanced courses.

Students who take Computer Science 102T, 107, or 109 may use that course as one of the two electives required for the major in Computer Science. Those who count Computer Science 109 toward the major must select an elective different from Computer Science 371 (Computational Graphics) for their project course. Computer Science 102T, 107, 109, 134, and 135 are not open to students who have taken a Computer Science course numbered 136 or higher.

To be eligible for admission to the major, a student must normally have completed Computer Science 136 as well as Discrete Mathematics by the end of the sophomore year. A second Mathematics course at the 200-level or higher must be completed by the end of the junior year. Students are urged to have completed two of the four core courses (Computer Science 237, 256, 334, and 361) by the end of the sophomore year and must normally have completed at least three out of the four core courses by the end of the junior year.

All computer science majors must attend at least twenty Computer Science Colloquia. Juniors and seniors are encouraged to attend at least five during each semester they are present on campus.

With the advance permission of the department, two appropriate mathematics or statistics courses may be substituted for one Computer Science elective. Appropriate mathematics classes are those numbered 300 or above, and appropriate statistics courses are those numbered 200 or above. Other variations in the required courses, adapting the requirements to the special needs and interests of the individual student, may be arranged in consultation with the department.

LABORATORY FACILITIES

The Computer Science Department maintains two departmental computer laboratories for students taking Computer Science courses, as well as a lab that can be configured for teaching specialized topics such as robotics. The workstations in these laboratories also support student and faculty research in computer science.

THE DEGREE WITH HONORS IN COMPUTER SCIENCE

The degree with honors in Computer Science is awarded to students who have demonstrated outstanding intellectual achievement in a program of study extending beyond the requirements of the regular major. The principal considerations in recommending a student for the degree with honors will be: mastery of core material, ability to pursue independent study of computer science, originality in methods of investigation, and creativity in research. Honors study is highly recommended for those students with strong academic records in computer science who wish to attend graduate school, pursue high-level industrial positions in computing, or who would simply like to experience research in computer science.

Prospective honors students are urged to consult with their departmental advisor at the time of registration in the spring of the sophomore or at the beginning of the junior year to arrange a program of study that could lead to the degree with honors. Such a program normally consists of Computer Science 493 and 494 and a WSP of independent research under the guidance of a Computer Science faculty member, culminating in a thesis that is judged acceptable by the department. The program produces a significant piece of written work and often includes a major computer program. All honors candidates are required to give an oral presentation of their research in the Computer Science Colloquium in early spring semester.

Students considering honors work should obtain permission from the department before registering in the fall of the senior year. Formal admission to candidacy occurs at the beginning of the spring semester of the senior year and is based on promising performance in the fall semester and winter study units of honors work. Recommendations for the degree with honors will be made for outstanding performance in the three honors courses. Highest honors will be recommended for students who have displayed exceptional ability, achievement, or originality.

INTRODUCTORY COURSES

The department offers a choice of five introductory courses; Computer Science 102 The Socio-Techno Web, 107 Creating Games, 109 The Art and Science of Computer Graphics, Computer Science 134 Introduction to Computer Science, and Computer Science 135 Diving into the Deluge of Data.

Computer Science 134 and 135 provide an introduction to computer science with a focus on developing computer programming skills. These skills are essential to most upper-level courses in the department. As a result, Computer Science 134 and 135 together with Computer Science 136, are required as a prerequisite to most advanced courses in the department. Those students intending to take several Computer Science courses are urged to take 134 or 135 early.

Those students interested in learning more about exciting new ideas in computer science, but not necessarily interested in developing extensive programming skills, should consider Computer Science 102 The Socio-Techno Web, 107 Creating Games, or 109 The Art and Science of Computer Graphics.

Students with significant programming experience should consider electing Computer Science 136 (see “Advanced Placement” below). Students are always welcome to contact a member of the department for guidance in selecting a first course.

STUDY ABROAD

Study abroad can be a wonderful experience. Students who hope to take computer science courses while abroad should discuss their plans in advance with the chair of the department. Students who plan to study away but do not expect to take courses toward the major should work with the department to create a plan to ensure that they will be able to complete the major. While study abroad is generally not an impediment to completing the major, students should be aware that certain computer science courses must be taken in a particular sequence and that not all courses are offered every semester (or every year). Students who wish to discuss their plans are invited to meet with any of the faculty in Computer Science. You can find general study away guidelines for Computer Science here.

ADVANCED PLACEMENT

Students with an extensive background in computer science are urged to take the Advanced Placement Examination in Computer Science. A score of 4 or better on the exam is normally required for advanced placement in Computer Science 136.

Students who wish to be placed in Computer Science 136 but who have not taken the Advanced Placement Examination should consult with the department. Such students should have had a good course in computer science using a structured language such as Java.

PLANS OF STUDY FOR NON-MAJORS

The faculty in Computer Science believes that students can substantially enrich their academic experience by completing a coherent plan of study in one or more disciplines outside of their majors. With this in mind, we have attempted to provide students majoring in other departments with options in our department’s curriculum ranging from two-course sequences to collections of courses equivalent to what would constitute a minor at institutions that recognize such a concentration. Students interested in designing such a plan of study are invited to discuss their plans in detail with a member of the faculty. To assist students making such plans, we include some suggestions below.

Students seeking to develop an extensive knowledge of computer science without majoring in the department are encouraged to use the major requirements as a guide. In particular, the four core courses required of majors are intended to provide a broad knowledge of topics underlying all of computer science. Students seeking a concentration in Computer Science are urged to complete at least two of these courses followed by one of our upper-level electives. Such a program would typically require the completion of a total of five Computer Science courses and one course in discrete mathematics.

There are several sequences of courses appropriate for those primarily interested in developing skills in programming for use in other areas. For general programming, Computer Science 134 or 135 followed by 136 and 256 will provide students with a strong background in algorithm and data structure design together with an understanding of issues of correctness and efficiency. Students of the Bioinformatics program are encouraged to take Computer Science 134 or 135 at a minimum, and should also consider Computer Science 136 and 256. The sequence of courses Computer Science 109 and 134 or 135 would provide sufficient competence in computer graphics for many students interested in applying such knowledge either in the arts or sciences. For students requiring more expertise in the techniques of computer graphics, Computer Science 136 and 371 could be added to form a four-course sequence.

There are, of course, many other alternatives. We encourage interested students to consult with the department chair or other members of the department's faculty.

GENERAL REMARKS

Divisional Requirements

All Computer Science courses may be used to satisfy the Division III distribution requirement.

Alternate Year Courses

Computer Science 102T, 107, 109, 205, 315, 336T, 337T, 339, 354, 356T, 371, 372, 373, 374T, 432, and 434T are each normally offered every other year. All other Computer Science courses are normally offered every year.

Course Numbering

The increase from 100, through 200 and 300, to 400 indicates in most instances an increasing level of maturity in the subject that is expected of students. Within a series, numeric order does not indicate the relative level of difficulty of courses. Rather, the middle digit of the course number (particularly in upper-level courses) generally indicates the area of computer science covered by the course.

Course Descriptions

Brief descriptions of the courses in Computer Science can be found below. More detailed information on the offerings in the department is available at http://www.cs.williams.edu/.

Courses Open on a Pass-Fail Basis

Students taking a Computer Science course on a pass-fail basis must meet all the requirements set for students taking the course on a graded basis.

With the permission of the department, any course offered by the department may be taken pass-fail (with the exception of tutorials), though courses graded with the pass-fail option may not be used to satisfy any of the major or honors requirements. However, with the permission of the department, courses taken in the department beyond those requirements may be taken on a pass-fail basis.

CSCI 102T The Socio-Techno Web (Q)

This course introduces many fundamental concepts in computer science by examining the social aspects of computing. As more and more people use the technologies and services available via the Internet, online environments like Facebook, Amazon, Google, Twitter, and blogs are flourishing. However, several of the problems related to security, privacy, and trust that exist in the real world transfer and become amplified in the virtual world created by the ubiquity and pervasiveness of the Internet. In this course, we will investigate how the social, technological, and natural worlds are connected, and how the study of networks sheds light on these connections. Topics include the structure of the Social Web and networks in general; issues such as virtual identity, personal and group privacy, trust evaluation and propagation, and online security; and the technology, economics, and politics of Web information and online communities. No background in computer science or programming is required or expected.
Class Format: tutorial
Requirements/Evaluation: evaluation will be based on tutorial discussions, presentations, problem sets and labs, a midterm exam, and a final project or paper
Extra Info: may not be taken on a pass/fail basis; not available for the fifth course option
Enrollment Preferences: first-year students and sophomores who have not previously taken a computer science course
Enrollment Limit: 10
Expected Class Size: 10
Distributional Requirements:
Division 3
Quantitative/Formal Reasoning
Other Attributes:
SCST Related Courses

Not Offered Academic Year 2017
TUT Instructor: Jeannie Albrecht

CSCI 107(S) Creating Games (Q)

Crosslistings: CSCI 107/ARTS 107
Primary Crosslisting
The game is unique as the only broadly-successful interactive art form. Games communicate the experience of embodying a role by manipulating the player's own decisions, abstraction, and discrete planning. Those three elements are the essence of computation, which makes computer science theory integral to game design. Video games also co-opt programming and computer graphics as new

tools for the modern artist. As a result, games are collaborative interdisciplinary constructs that use computation as a medium for creative expression. Students analyze and extend contemporary video and board games using the methodology of science and the language of the arts. They explore how computational concepts like recursion, state, and complexity apply to interactive experiences. They then synthesize new game elements using mathematics, programming and both digital and traditional art tools. Emphasis is on the theory of design in modern European board games. Topics covered include experiment design, gameplay balance, minimax, color theory, pathfinding, game theory, composition, and computability.
Class Format: lecture and studio
Requirements/Evaluation: evaluation will be based on participation, studio work, and quizzes
Extra Info: may not be taken on a pass/fail basis
Prerequisites: none; no programming or game experience is assumed
Enrollment Preferences: first-year students
Enrollment Limit: 19
Expected Class Size: 19
Dept. Notes: not open to students who completed a Computer Science course numbered 136 or above; does not count toward the Art Major
Materials/Lab Fee: lab fee of $25 will be added to the student's term bill
Distribution Notes: meets Division 3 requirement if registration is under CSCI; meets Division 1 requirement if registration is under ARTS
Distributional Requirements:
Division 3
Quantitative/Formal Reasoning
Other Attributes:
FMST Core Courses

Spring 2017
LEC Section: 01 TR 08:30 AM 09:45 AM Instructor: Morgan McGuire

LAB Section: 02 R 01:00 PM 04:00 PM Instructor: Morgan McGuire

CSCI 109(F) The Art and Science of Computer Graphics (Q)
This course provides an opportunity to develop an understanding of the theoretical and practical concepts underlying 2- and 3-dimensional computer graphics. The course will emphasize hands-on studio/laboratory experience, with student work focused around completing a series of projects. Students will experiment with modeling, color, lighting, perspective, and simple animation. As the course progresses, computer programming will be used to control the complexity of the models and their interactions. Lectures, augmented by guided viewings of state-of-the-art computer generated and enhanced images and animations, will be used to deepen understanding of the studio experience.
Class Format: lecture/laboratory
Requirements/Evaluation: evaluation will be based on progress in project work and two examinations
Prerequisites: this course is not open to students who have successfully completed a CSCI course numbered 136 or above
Enrollment Preferences: first-year students and sophomores who have not previously taken a computer science course
Enrollment Limit: 36
Expected Class Size: 36
Distributional Requirements:
Division 3
Quantitative/Formal Reasoning
Other Attributes:
FMST Related Courses

Fall 2016
LEC Section: 01 TR 08:30 AM 09:45 AM Instructor: Duane Bailey

LAB Section: 02 R 01:00 PM 02:25 PM Instructor: Duane Bailey

LAB Section: 03 R 02:35 PM 04:00 PM Instructor: Duane Bailey

CSCI 134(F,S) Introduction to Computer Science (Q)
This course introduces fundamental ideas in computer science and builds skills in the design, implementation, and testing of computer programs. Students implement algorithms in the Java programming language with a strong focus on constructing correct, understandable, and efficient programs. Students explore the material through specific application areas. Topics covered include object-oriented programming, control structures, arrays, recursion, and event-driven programming. This course is appropriate for all students who want to create software and have little or no prior computing experience. More details are available on the department website, http://www.cs.williams.edu.
Class Format: lecture/laboratory
Requirements/Evaluation: evaluation will be based on weekly assignments, final programming projects, and examinations
Prerequisites: none, except for the standard prerequisites for a (Q) course; previous programming experience is not required
Enrollment Preferences: first year students and sophomores
Enrollment Limit: 30
Expected Class Size: 30

Dept. Notes: students with prior experience with object-oriented programming should discuss appropriate course placement with members of the department
Distributional Requirements:
Division 3
Quantitative/Formal Reasoning
Other Attributes:
BGNP Recommended Courses
COGS Interdepartmental Electives

Fall 2016
LEC Section: 01   MWF 09:00 AM 09:50 AM   Instructors: Stephen Freund, Andrea Danyluk

LEC Section: 02   MWF 10:00 AM 10:50 AM   Instructors: Andrea Danyluk, Stephen Freund

LAB Section: 03   M 07:00 PM 10:00 PM   Instructors: Stephen Freund, Andrea Danyluk

LAB Section: 04   T 01:00 PM 04:00 PM   Instructors: Andrea Danyluk, Stephen Freund

LAB Section: 05   T 08:30 AM 11:20 AM   Instructors: Andrea Danyluk, Stephen Freund

Spring 2017
LEC Section: 01   MWF 09:00 AM 09:50 AM   Instructor: Thomas Murtagh

LEC Section: 02   MWF 10:00 AM 10:50 AM   Instructor: Thomas Murtagh

LAB Section: 03   T 01:00 PM 04:00 PM   Instructor: Thomas Murtagh

LAB Section: 04   M 07:00 PM 10:00 PM   Instructor: Thomas Murtagh

LAB Section: 05   M 01:00 PM 04:00 PM   Instructor: Thomas Murtagh

CSCI 135(F,S) Diving into the Deluge of Data (Q)
We are surrounded by information: weather forecasts, twitter feeds, restaurant reviews, stock market tickers, music recommendations, among others. This course introduces fundamental computational concepts for representing and manipulating data. Using the programming language Python, this course explores effective ways to organize and transform information in order to solve problems. Students will learn to design algorithms to search, sort, and manipulate data in application areas like text and image processing, social networks, scientific computing, databases, and the World Wide Web. Programming topics covered include object-oriented and functional programming, control structures, types, recursion, arrays, lists, streams, and dictionaries. This course is appropriate for all students who want to create software and learn computational techniques for manipulating and analyzing data. More details are available on the department website, http://www.cs.williams.edu
Class Format: lecture/laboratory
Requirements/Evaluation: evaluation will be based on weekly assignments, programming projects, and examinations
Prerequisites: some experience programming in any computer language; not open to students who have successfully completed CSCI 134 or above
Enrollment Preferences: first year and sophomore students
Enrollment Limit: 30
Expected Class Size: 30
Dept. Notes: students with substantial prior programming experience should discuss appropriate course placement with members of the department; this course may be taken in place of CSCI 134 & fulfills a CSCI 134 prerequisite for other courses
Distributional Requirements:
Division 3
Quantitative/Formal Reasoning

Fall 2016
LEC Section: 01   MWF 09:00 AM 09:50 AM   Instructor: Brent Heeringa

LEC Section: 02   MWF 11:00 AM 11:50 AM   Instructor: Bill Jannen

LAB Section: 03   M 01:00 PM 02:25 PM   Instructor: Brent Heeringa

LAB Section: 04   M 02:35 PM 04:00 PM   Instructor: Brent Heeringa

LAB Section: 05   M 01:00 PM 02:25 PM   Instructor: Bill Jannen

LAB Section: 06   M 02:35 PM 04:00 PM   Instructor: Bill Jannen

Spring 2017
LEC Section: 01   Cancelled   Instructor: Jon Park

LEC Section: 02   MWF 11:00 AM 11:50 AM   Instructor: Duane Bailey

LAB Section: 03   Cancelled   Instructor: Jon Park

LAB Section: 04   M 02:35 PM 04:00 PM   Instructor: Jon Park

LAB Section: 05   T 08:30 AM 09:50 AM   Instructor: Duane Bailey

LAB Section: 06   Cancelled   Instructor: Duane Bailey

CSCI 136(F,S) Data Structures and Advanced Programming (Q)
This course builds on the programming skills acquired in Computer Science 134. It couples work on program design, analysis, and verification with an introduction to the study of data structures. Data structures capture common ways in which to store and manipulate data, and they are important in the construction of sophisticated computer programs. Students are introduced to some of the most important and frequently used data structures: lists, stacks, queues, trees, hash tables, graphs, and files. Students will be expected to write several programs, ranging from very short programs to more elaborate systems. Emphasis will be placed on the development of clear, modular programs that are easy to read, debug, verify, analyze, and modify.
Class Format: lecture/laboratory
Requirements/Evaluation: evaluation will be based on programming assignments, homework and/or examinations
Prerequisites: CSCI 134 or equivalent; MATH 200 is recommended, but not required
Enrollment Preferences: first-year and sophomore students
Enrollment Limit: 36-F 36-S
Expected Class Size: 36-F 36-S
Dept. Notes: Single section in the Fall has enrollment limit of 36. Each Spring section (two) will have a limit of 24.
Distributional Requirements:
Division 3
Quantitative/Formal Reasoning
Other Attributes:
BGNP Recommended Courses

Fall 2016
LEC Section: 01   MWF 10:00 AM 10:50 AM   Instructor: William Lenhart

LEC Section: 02   MWF 09:00 AM 09:50 AM   Instructor: William Lenhart

LAB Section: 03   W 12:00 PM 02:00 PM   Instructor: William Lenhart

LAB Section: 04   W 02:00 PM 04:00 PM   Instructor: William Lenhart

Spring 2017
LEC Section: 01   MWF 10:00 AM 10:50 AM   Instructor: Bill Jannen

LEC Section: 02   MWF 09:00 AM 09:50 AM   Instructor: Morgan McGuire

LAB Section: 03   R 01:00 PM 04:00 PM   Instructor: Bill Jannen

LAB Section: 04   W 01:00 PM 04:00 PM   Instructor: Morgan McGuire

LAB Section: 05 W 01:00 PM 04:00 PM Instructor: Jon Park

LAB Section: 06 W 07:00 PM 10:00 PM Instructor: Jon Park

CSCI 205 Cinematography in the Digital Age (D)
Crosslistings: ENGL 203/CSCI 205/ARTH 205
Secondary Crosslisting
In this course we study the language of modern cinema as shaped by two forces. The first is the aesthetics of cinematography, as contributed by many cultures. The second is digital film production, which has proved both empowering and constraining. The modern filmmaker succeeds only through understanding both forces.
The structure of the course is similar to a writing workshop. We begin with close reading of isolated scenes from influential films, which we compare and critique in writing and discussion. We augment this with cinematic and image processing theory, solidified through experiments in Photoshop and Premiere that reveal how digital technology shapes a director's choices. We then create our own short scenes using these tools and consumer video recorders. We refine our film fragments in the context of group critique.
Topics covered include: framing and composition, pace, storyboarding, blocking, lighting, transitions, perspective, sensors, quantization, compression, visual effects, Internet streaming, and color spaces. Studied films include those by Georges Méliès, Stanley Kubrick, Joris Ivens, Barbara Kopple, Martin Scorsese, Sarah Polley, Orson Welles, David Lynch, Fritz Lang, Michael Haneke, Hayao Miyazaki, Spike Lee, Sophia Coppola, and Ken Burns. This course explores diversity through comparative study of how different cultures variously render similar themes, and through a larger investigation of film's ability to make audiences identify with potentially alien points of view.
Class Format: studio

Requirements/Evaluation: video production activity, computational exercises in Photoshop, script and storyboarding exercises, participation in discussions, and essays
Extra Info: may not be taken on a pass/fail basis; not available for the fifth course option
Prerequisites: a 100 level English course, or a score of 5 on the AP Exam in English Literature or a 6 or 7 on the International Baccalaureate; or permission of instructor
Enrollment Preferences: sophomores; Computer Science and English majors
Enrollment Limit: 36
Expected Class Size: 30
Distribution Notes: meets Division 1 requirement if registration is under ENGL or ARTH; meets Division 3 requirement if registration is under CSCI
Distributional Requirements:
Division 1
Exploring Diversity
Other Attributes:
FMST Core Courses

Not Offered Academic Year 2017
STU Instructor: Shawn Rosenheim

CSCI 237(F,S) Computer Organization (Q)
This course studies the basic instruction set architecture and organization of a modern computer. Over the semester the student learns the fundamentals of translating higher level languages into assembly language, and the interpretation of machine languages by hardware. At the same time, a model of computer hardware organization is developed from the gate level upward. Final projects focus on the design of a complex control system.
Class Format: lecture/laboratory
Requirements/Evaluation: evaluation will be based primarily on weekly labs, a final project, and one or more exams
Extra Info: may not be taken on a pass/fail basis; not available for the fifth course option
Prerequisites: CSCI 134, or both experience in programming and permission of instructor
Enrollment Preferences: current or expected Computer Science majors
Enrollment Limit: 30
Expected Class Size: 30
Distributional Requirements:
Division 3
Quantitative/Formal Reasoning

Fall 2016
LEC Section: 01 MWF 11:00 AM 11:50 AM Instructor: Jeannie Albrecht

LAB Section: 02 T 01:00 PM 02:25 PM Instructor: Jeannie Albrecht

LAB Section: 03 T 02:35 PM 04:00 PM Instructor: Jeannie Albrecht

Spring 2017
LEC Section: 01 MWF 12:00 PM 12:50 PM Instructor: Duane Bailey

LAB Section: 02 T 01:00 PM 02:25 PM Instructor: Duane Bailey

LAB Section: 03 T 02:35 PM 04:00 PM Instructor: Duane Bailey

CSCI 256(S) Algorithm Design and Analysis (Q)
This course investigates methods for designing efficient and reliable algorithms. By carefully analyzing the structure of a problem within a mathematical framework, it is often possible to dramatically decrease the computational resources needed to find a solution. In addition, analysis provides a method for verifying the correctness of an algorithm and accurately estimating its running time and space requirements. We will study several algorithm design strategies that build on data structures and programming techniques introduced in Computer Science 136. These include induction, divide-and-conquer, dynamic programming, and greedy algorithms. Additional topics of study include algorithms on graphs and strategies for handling potentially intractable problems.
Class Format: lecture
Requirements/Evaluation: evaluation will be based on problem sets and programming assignments, and midterm and final examinations
Prerequisites: CSCI 136 and MATH 200
Enrollment Preferences: current or expected Computer Science majors
Enrollment Limit: 40
Expected Class Size: 40
Distributional Requirements:
Division 3
Quantitative/Formal Reasoning
Other Attributes:
BGNP Recommended Courses

Spring 2017
LEC Section: 01   MWF 10:00 AM 10:50 AM   Instructor: Brent Heeringa

CSCI 315T Computational Biology (Q)
Crosslistings: PHYS 315/CSCI 315
Secondary Crosslisting
This course will provide an overview of Computational Biology, the application of computational, mathematical, statistical, and physical problem-solving techniques to interpret the rapidly expanding amount of biological data. Topics covered will include database searching, DNA sequence alignment, clustering, RNA structure prediction, protein structural alignment, methods of analyzing gene expression, networks, and genome assembly using techniques such as string matching, dynamic programming, hidden Markov models, and expectation-maximization.
Class Format: lab three hours per week plus weekly tutorial meeting
Requirements/Evaluation: evaluation will be based on weekly programming assignments, problem sets, a few quizzes and a final project
Extra Info: may not be taken on a pass/fail basis, not available for the fifth course option
Prerequisites: programming experience (e.g., CSCI 136), mathematics (PHYS/MATH 210 or MATH 150), and physical science (PHYS 142 or 151, or CHEM 151 or 153 or 155), or permission of instructor
Enrollment Preferences: based on seniority
Enrollment Limit: 10
Expected Class Size: 8
Distributional Requirements:
Division 3
Quantitative/Formal Reasoning
Other Attributes:
BGNP Recommended Courses

Not Offered Academic Year 2017
TUT   Instructor: Daniel Aalberts

CSCI 319(F) Integrative Bioinformatics, Genomics, and Proteomics Lab (Q)
Crosslistings: BIOL 319/MATH 319/CHEM 319/PHYS 319/CSCI 319
Secondary Crosslisting
What can computational biology teach us about cancer? In this capstone experience for the Genomics, Proteomics, and Bioinformatics program, computational analysis and wet-lab investigations will inform each other, as students majoring in biology, chemistry, computer science, mathematics/statistics, and physics contribute their own expertise to explore how ever-growing gene and protein data-sets can provide key insights into human disease. In this course, we will take advantage of one well-studied system, the highly conserved Ras-related family of proteins, which play a central role in numerous fundamental processes within the cell. The course will integrate bioinformatics and molecular biology, using database searching, alignments and pattern matching, phylogenetics, and recombinant DNA techniques to reconstruct the evolution of gene families by focusing on the gene duplication events and gene rearrangements that have occurred over the course of eukaryotic speciation. By utilizing high through-put approaches to investigate genes involved in the MAPK signal transduction pathway in human colon cancer cell lines, students will uncover regulatory mechanisms that are aberrantly altered by siRNA knockdown of putative regulatory components. This functional genomic strategy will be coupled with independent projects using phosphorylation-state specific antisera to test our hypotheses. Proteomic analysis will introduce the students to de novo structural prediction and threading algorithms, as well as data-mining approaches and Bayesian modeling of protein network dynamics in single cells. Flow cytometry and mass spectrometry will be used to study networks of interacting proteins in colon tumor cells.
Class Format: two afternoons of lab, with one hour of lecture, per week
Requirements/Evaluation: lab participation, several short homework assignments, one lab report, a programming project, and a grant proposal
Prerequisites: BIOL 202; students who have not taken BIOL 202 but have taken BIOL 101 and CSCI 315 or PHYS 315, may enroll with permission of instructor. No prior computer programming experience is required.
Enrollment Preferences: seniors, then juniors, then sophomores
Enrollment Limit: 12
Expected Class Size: 12
Dept. Notes: does not satisfy the distribution requirement in the Biology major
Distributional Requirements:
Division 3
Quantitative/Formal Reasoning
Other Attributes:
BGNP Core Courses
BIMO Interdepartmental Electives

Fall 2016
LEC Section: 01   W 12:25 PM 01:00 PM   Instructor: Lois Banta

LAB Section: 02   WR 01:00 PM 04:00 PM   Instructor: Lois Banta

LAB Section: 03   Cancelled

CSCI 326 Software Methods (Q)

Sophisticated software systems play a prominent role in many aspects of our lives, and while programming can be a very creative and exciting process, building a reliable software system of any size is no easy feat. Moreover, the ultimate outcome of any programming endeavor is likely to be incomplete, unreliable, and unmaintainable unless principled methods for software construction are followed. This course explores those methods.
Specific topics include: software processes; specifying requirements and verifying correctness; software architectures; concurrent, parallel, and scalable systems design; testing and debugging; and performance evaluation.
Class Format: lecture/lab
Requirements/Evaluation: homework, programming assignments, group work, presentations, exams
Prerequisites: CSCI 136
Enrollment Preferences: current or expected Computer Science majors; those who have not yet taken a project course
Enrollment Limit: 18
Expected Class Size: 18
Distributional Requirements:
Division 3
Quantitative/Formal Reasoning

Not Offered Academic Year 2017
LEC

CSCI 334(S) Principles of Programming Languages (Q)
This course examines the concepts and structures governing the design and implementation of programming languages. It presents an introduction to the concepts behind compilers and run-time representations of programming languages; features of programming languages supporting abstraction and polymorphism; and the procedural, functional, object-oriented, and concurrent programming paradigms. Programs will be required in languages illustrating each of these paradigms.
Class Format: lecture
Requirements/Evaluation: evaluation will be based on weekly problem sets and programming assignments, a midterm examination and a final examination
Prerequisites: CSCI 136
Enrollment Preferences: current or expected Computer Science majors
Enrollment Limit: 40
Expected Class Size: 40
Distributional Requirements:
Division 3
Quantitative/Formal Reasoning

Spring 2017
LEC Section: 01   TR 09:55 AM 11:10 AM   Instructor: Stephen Freund

CSCI 336T Computer Networks (Q)
This course explores the design and implementation of computer networks. Topics include wired and wireless networks; techniques for efficient and reliable encoding and transmission of data; addressing schemes and routing mechanisms; resource allocation for bandwidth sharing; and security issues. An important unifying themes is the distributed nature of all network problems. We will examine the ways in which these issues are addressed by current protocols such as TCP/IP and 802.11 WIFI.
Class Format: lecture
Requirements/Evaluation: evaluation will be based on problem sets, programming assignments, and midterm and final examinations
Extra Info: may not be taken on a pass/fail basis; not available for the fifth course option
Prerequisites: CSCI 136 and 237
Enrollment Preferences: current or expected Computer Science majors
Enrollment Limit: 24
Expected Class Size: 24
Dept. Notes: project course
Distributional Requirements:
Division 3
Quantitative/Formal Reasoning

Not Offered Academic Year 2017
TUT   Instructor: Thomas Murtagh

CSCI 337T Digital Design and Modern Architecture (Q)
This tutorial course considers topics in the low-level design of modern architectures. Course meetings will review problems of designing effective architectures including instruction-level parallelism, branch-prediction, caching strategies, and advanced ALU design. Readings will be taken from recent technical literature. Labs will focus on the development of custom CMOS circuits to implement projects from gates to bit-sliced ALU's. Final group projects will develop custom logic demonstrating concepts learned in course meetings.
Class Format: tutorial
Requirements/Evaluation: evaluation will be based on microprocessor design projects, participation in tutorial meetings, and examinations
Extra Info: may not be taken on a pass/fail basis; not available for the fifth course option
Prerequisites: CSCI 237

Enrollment Preferences: current or expected Computer Science majors
Enrollment Limit: 10
Expected Class Size: 10
Distributional Requirements:
Division 3
Quantitative/Formal Reasoning

Not Offered Academic Year 2017
TUT Instructor: Duane Bailey

CSCI 339 Distributed Systems (Q)
This course studies the key design principles of distributed systems, which are collections of independent networked computers that function as single coherent systems. Covered topics include communication protocols, processes and threads, naming, synchronization, consistency and replication, fault tolerance, and security. Students also examine some specific real-world distributed systems case studies, including Google and Amazon. Class discussion is based on readings from the textbook and research papers. The goals of this course are to understand how large-scale computational systems are built, and to provide students with the tools necessary to evaluate new technologies after the course ends.
Class Format: lecture/laboratory
Requirements/Evaluation: evaluation will be based on homework assignments, programming projects, and exams
Extra Info: may not be taken on a pass/fail basis; not available for the fifth course option
Prerequisites: CSCI 136 or equivalent programming experience, and CSCI 237, or permission of instructor
Enrollment Preferences: current or expected Computer Science majors
Enrollment Limit: 30
Expected Class Size: 30
Dept. Notes: project course
Distributional Requirements:
Division 3
Quantitative/Formal Reasoning

Not Offered Academic Year 2017
LEC Instructor: Jeannie Albrecht

CSCI 356T(F) Advanced Algorithms (Q)
This course explores advanced concepts in algorithm design, algorithm analysis and data structures. Areas of focus will include algorithmic complexity, randomized and approximation algorithms, geometric algorithms, and advanced data structures. Topics will include combinatorial algorithms for packing, and covering problems, algorithms for proximity and visibility problems , linear programming algorithms, approximation schemes, hardness of approximation, search, and hashing.
Class Format: tutorial
Requirements/Evaluation: evaluation is based on weekly problem sets, several small programming projects, weekly paper summaries, and a small, final project
Extra Info: may not be taken on a pass/fail basis; not available for the fifth course option
Prerequisites: CSCI 256; CSCI 361 is recommended but not required
Enrollment Preferences: current or expected Computer Science majors
Enrollment Limit: 10
Expected Class Size: 10
Distributional Requirements:
Division 3
Quantitative/Formal Reasoning

Fall 2016
TUT Section: T1 Cancelled

CSCI 361(F) Theory of Computation (Q)
Crosslistings: CSCI 361/MATH 361
Primary Crosslisting
This course introduces a formal framework for investigating both the computability and complexity of problems. We study several models of computation including finite automata, regular languages, context-free grammars, and Turing machines. These models provide a mathematical basis for the study of computability theory—the examination of what problems can be solved and what problems cannot be solved—and the study of complexity theory—the examination of how efficiently problems can be solved. Topics include the halting problem and the P versus NP problem.
Class Format: lecture
Requirements/Evaluation: evaluation will be based on problem sets, a midterm examination, and a final examination
Prerequisites: CSCI 256 or both a 300-level MATH course and permission of instructor
Enrollment Preferences: current or expected Computer Science majors
Enrollment Limit: 40
Expected Class Size: 40
Distributional Requirements:
Division 3
Quantitative/Formal Reasoning

Other Attributes:
COGS Interdepartmental Electives

Fall 2016
LEC Section: 01   MWF 12:00 PM 12:50 PM   Instructor: Thomas Murtagh

CSCI 371(F) Computational Graphics (Q)
PhotoShop, medical MRIs, video games, and movie special effects all programmatically create and manipulate digital images. This course teaches the fundamental techniques behind these applications. We begin by building a mathematical model of the interaction of light with surfaces, lenses, and an imager. We then study the data structures and processor architectures that allow us to efficiently evaluate that physical model. Students will complete a series of programming assignments for both photorealistic image creation and real-time 3D rendering using C++, OpenGL, and GLSL. These assignments cumulate in a multi-week final project. Topics covered in the course include: projective geometry, ray tracing, bidirectional surface scattering functions, binary space partition trees, matting and compositing, shadow maps, cache management, and parallel processing on GPUs.
Class Format: lecture, with optics laboratory exercises
Requirements/Evaluation: evaluation based on assignments, projects, and exams
Extra Info: may not be taken on a pass/fail basis; not available for the fifth course option
Prerequisites: CSCI 136 and CSCI 237 or permission of instructor
Enrollment Preferences: current or expected Computer Science majors
Enrollment Limit: 24
Expected Class Size: 24
Dept. Notes: project course
Distributional Requirements:
Division 3
Quantitative/Formal Reasoning
Other Attributes:
FMST Core Courses

Fall 2016
LEC Section: 01   TR 09:55 AM 11:10 AM   Instructor: Morgan McGuire

LAB Section: 02   R 01:00 PM 04:00 PM   Instructor: Morgan McGuire

CSCI 372T Visual Media Revolution (Q)
We live at the beginning of the second revolution in visual media. Two centuries ago, the camera and the Jacquard loom introduced machines for creating art. By automating the artist's hand, they also forced questions of how objective technique gives rise to subjective meaning and where the border lies between mechanical and human contributions. Those progenitors eventually led to digital film, computer games, and digital content creation for architecture and industrial design.
Today, accessible and pervasive computation provokes a second revolution. Augmented reality, 3D scanning, 3D printing, virtual reality, and computational photography are exploding into mainstream experience. Where previous digital media refined analog practice through evolution, these are forms that could not exist without computation. As the world seeks the promise of new visual forms, we find that fundamentals of earlier media remain valid and take them as our guide. This tutorial investigates the technology of emerging computational media and explores their impact on the relationship between process and aesthetics.
Class Format: tutorial
Requirements/Evaluation: oral presentations and short papers
Extra Info: may not be taken on a pass/fail basis; not available for the fifth course option
Prerequisites: CSCI 256
Enrollment Preferences: current or expected Computer Science majors
Enrollment Limit: 10
Expected Class Size: 10
Distributional Requirements:
Division 3
Quantitative/Formal Reasoning
Other Attributes:
FMST Core Courses

Not Offered Academic Year 2017
TUT   Instructor: Morgan McGuire

CSCI 373(F,S) Artificial Intelligence (Q)
Artificial Intelligence (AI) has become part of everyday life, but what is it, and how does it work? This course introduces theories and computational techniques that serve as a foundation for the study of artificial intelligence. Potential topics include the following: Problem solving by search, Logic, Planning, Constraint satisfaction problems, Uncertainty and probabilistic reasoning, Bayesian networks, and Automated Learning.
Class Format: lecture/laboratory
Requirements/Evaluation: several programming projects in the first half of the semester and a larger project spanning most of the second half of the semester; reading responses and discussion; midterm examination
Extra Info: may not be taken on a pass/fail basis; not available for the fifth course option
Prerequisites: CSCI 136 and (CSCI 256 or permission of intstructor)

Enrollment Preferences: current or expected Computer Science majors
Enrollment Limit: 24
Expected Class Size: 24
Dept. Notes: project course
Distributional Requirements:
Division 3
Quantitative/Formal Reasoning
Other Attributes:
COGS Interdepartmental Electives

Fall 2016
LEC Section: 01   MWF 09:00 AM 09:50 AM   Instructor: Jon Park

Spring 2017
LEC Section: 01   MWF 09:00 AM 09:50 AM   Instructor: Andrea Danyluk

CSCI 374T(F) Machine Learning (Q)
This tutorial examines the design, implementation, and analysis of machine learning algorithms. Machine Learning is a branch of Artificial Intelligence that aims to develop algorithms that will improve a system's performance. Improvement might involve acquiring new factual knowledge from data, learning to perform a new task, or learning to perform an old task more efficiently or effectively. This tutorial will cover examples of supervised learning algorithms (including decision tree learning, support vector machines, and neural networks), unsupervised learning algorithms (including k-means and expectation maximization), and possibly reinforcement learning algorithms (such as Q learning and temporal difference learning). It will also introduce methods for the evaluation of learning algorithms, as well as topics in computational learning theory.
Class Format: tutorial
Requirements/Evaluation: evaluation will be based on presentations, problem sets, programming exercises, empirical analyses of algorithms, critical analysis of current literature
Extra Info: may not be taken on a pass/fail basis; not available for the fifth course option
Prerequisites: CSCI 136 and CSCI 256 or permission of instructor
Enrollment Preferences: Computer Science majors
Enrollment Limit: 10
Expected Class Size: 10
Distributional Requirements:
Division 3
Quantitative/Formal Reasoning
Other Attributes:
COGS Interdepartmental Electives

Fall 2016
TUT Section: T1   TBA   Instructor: Andrea Danyluk

CSCI 375(F,S) Natural Language Processing (Q)
Natural language processing is a branch of computer science that studies methods for analyzing and generating written or spoken human language. It is a rapidly developing field that has given rise to many useful applications including search engines, speech recognizers, and automated personal assistants. Potential topics include information retrieval, information extraction, question answering, and language models.
Class Format: lecture
Requirements/Evaluation: exams, problem sets, and programming projects
Extra Info: may not be taken on a pass/fail basis; not available for the fifth course option
Prerequisites: CSCI 136 or Permission from Instructor
Enrollment Limit: 24
Expected Class Size: 24
Dept. Notes: Project Course
Distributional Requirements:
Division 3
Quantitative/Formal Reasoning

Fall 2016
LEC Section: 01   MWF 10:00 AM 10:50 AM   Instructor: Johan Boye

Spring 2017
LEC Section: 01   TR 08:30 AM 09:45 AM   Instructor: Jon Park

CSCI 397(F) Independent Reading: Computer Science
Directed independent reading in Computer Science.
Class Format: independent study
Prerequisites: permission of department
Distributional Requirements:
Division 3

Fall 2016
IND Section: 01   TBA   Instructor: Brent Heeringa

CSCI 398(S) Independent Reading: Computer Science
Directed independent reading in Computer Science.
Class Format: independent study
Prerequisites: permission of department
Distributional Requirements:
Division 3

Spring 2017
IND Section: 01   TBA   Instructor: Brent Heeringa

CSCI 432(S) Operating Systems (Q)
This course explores the design and implementation of computer operating systems. Topics include historical aspects of operating systems development, systems programming, process scheduling, synchronization of concurrent processes, virtual machines, memory management and virtual memory, I/O and file systems, system security, os/architecture interaction, and distributed operating systems.
Class Format: lecture/laboratory
Requirements/Evaluation: evaluation will be based on several implementation projects that will include significant programming, as well as written homework and exams
Extra Info: may not be taken on a pass/fail basis; not available for the fifth course option
Prerequisites: CSCI 237 and either CSCI 256 or 334
Enrollment Preferences: current or expected Computer Science majors
Enrollment Limit: 24
Expected Class Size: 24
Dept. Notes: project course
Distributional Requirements:
Division 3
Quantitative/Formal Reasoning

Spring 2017
LEC Section: 01   MR 01:10 PM 02:25 PM   Instructor: Jeannie Albrecht

LEC Section: 02   TR 11:20 AM 12:35 PM   Instructor: Jeannie Albrecht

CSCI 434T(F) Compiler Design (Q)
This tutorial covers the principles and practices for the design and implementation of compilers and interpreters. Topics include all stages of the compilation and execution process: lexical analysis; parsing; symbol tables; type systems; scope; semantic analysis; intermediate representations; run-time environments and interpreters; code generation; program analysis and optimization; and garbage collection. The course covers both the theoretical and practical implications of these topics. As a project course, students will construct a full compiler for a simple object-oriented language.
Class Format: tutorial
Requirements/Evaluation: evaluation will be based on presentations, problem sets, a substantial implementation project, and two exams
Extra Info: may not be taken on a pass/fail basis; not available for the fifth course option
Prerequisites: CSCI 237 and 361 (concurrent enrollment is acceptable); CSCI 334 is recommended, but not required
Enrollment Preferences: current or expected Computer Science majors
Enrollment Limit: 10
Expected Class Size: 10
Dept. Notes: project course
Distributional Requirements:
Division 3
Quantitative/Formal Reasoning

Fall 2016
TUT Section: T1   Cancelled

LAB Section: T2   Cancelled

CSCI 493(F) Research in Computer Science
This course provides highly-motivated students an opportunity to work independently with faculty on research topics chosen by individual faculty. Students are generally expected to perform a literature review, identify areas of potential contribution, and explore extensions to existing results. The course culminates in a concise, well-written report describing a problem, its background history, any independent results achieved, and directions for future research.
Class Format: independent study
Requirements/Evaluation: evaluation will be based on class participation, presentations, and the final written report
Enrollment Preferences: open to senior Computer Science majors with permission of instructor

Dept. Notes: this course (along with CSCI W31 and CSCI 494) is required for students pursuing honors, but enrollment is not limited to students pursuing honors
Distributional Requirements:
Division 3

Fall 2016
HON Section: 01   TBA   Instructor: Brent Heeringa

CSCI 494(S) Senior Thesis: Computer Science
Class Format: independent study
Requirements/Evaluation: evaluation will be based on class participation, presentations, and the final written report
Prerequisites: CSCI 493
Enrollment Preferences: open to senior Computer Science majors with permission of instructor
Distributional Requirements:
Division 3

Spring 2017
HON Section: 01   TBA   Instructor: Brent Heeringa

CSCI 497(F) Independent Reading: Computer Science
Directed independent reading in Computer Science.
Class Format: independent study
Prerequisites: permission of department
Distributional Requirements:
Division 3

Fall 2016
IND Section: 01   TBA   Instructor: Brent Heeringa

CSCI 498(S) Independent Reading: Computer Science
Directed independent reading in Computer Science.
Class Format: lecture/laboratory
Prerequisites: permission of department
Distributional Requirements:
Division 3

Spring 2017
IND Section: 01   TBA   Instructor: Brent Heeringa