

Guide to Courses
Department of Computer Science
Effective Fall 2007

409:022 *Internship in Computer Science*

Description: Cooperative education and internship assignments, on- or off-campus, provide experience related to each student's academic interests. The cooperative education opportunities are open to undergraduate computer science students who have completed 22C:021 and Calculus I, 24 hours of undergraduate work, and who have a GPA of at least 2.50. Graduate students must have 15 hours of graduate coursework and a GPA of 3.0. For more information on potential assignments, visit the Pomerantz Career Center, Room 100.

To obtain special permission, contact the Career Center

Approved GE: None

Prerequisites: 22C:016, 22C:021, and 22M:025 or 22M:031

22C:001 *Computer Literacy*

Description: This course provides a balanced introduction to computer technology and its applications, for the student wishing a rapid overview. Topics include computer literacy; computer system organization; history of computing; the impact of computer technology on society; issues of privacy, ethics, and security; networks and the Internet; artificial intelligence and robotics. The course also provides a variety of useful projects using applications, including communications, word processing, desktop publishing, spreadsheets, graphics, databases, and the World Wide Web. This course is not part of the computer science major or minor curriculum. The course is taught by TAs in the daytime sections; by faculty in the evening sections.

Not open for credit to students who have completed a higher-numbered 22C course or the course 6K:70.

Approved GE: None

22C:002 *First-Year Seminar*

Description: Small discussion class taught by a faculty member; topics chosen by instructor; may include outside activities (e.g., films, lectures, performances, readings, visits to research facilities). Open only to first- and second-semester students.

Approved GE: None

22C:005 Introduction to Computer Science

Description: This is a one-semester introductory course in computer science. It is appropriate for non-majors, minors, and for majors who want a broad overview of the discipline. Topics include computing fundamentals such as digital data representation, how computers work, simple algorithms and their efficiency, networks and databases. Hands-on experience with Web page development using state-of-the-art software such as Flash (most Internet ads are Flash movie clips). The course concludes with an overview of artificial intelligence and the scope and limitations of information technology. Requirements include a midterm and a final exam, which covers only the second half of the course. A pool of exam sample questions is provided by the instructor a week before each exam. Lab projects are a central and exciting part of the course. Labs are always explained and demonstrated by the instructor during the lectures. The labs include three Web page development projects that use text, graphics, sound, animation, and special effects. The course is taught by a faculty member; lab sections are led by TAs.

Not open for credit to students who have completed 22C:021 or a higher numbered 22C course.

Approved GE: Quantitative or Formal Reasoning

Prerequisites: Satisfaction of the admission requirement in mathematics

22C:016 Computer Science I: Fundamentals

Description: This is the introduction to programming course in the computer science major and minor curriculum. Prior programming experience is not required, although some students have had some previous exposure to programming. It emphasizes object oriented programming style and methodology using the Java programming language. The lecture is taught three times a week. The 50-minute discussion periods, led by a TA, are used to discuss programming exercises, and to answer questions in a small classroom atmosphere. Concepts are presented in the context of a complete collection of working examples and exercises. Both applets and applications are studied. Programming projects are the heart of the course, and are used to implement key programming notions, including control structures, visibility, objects and classes, composition and inheritance. Lectures are taught by a faculty member; discussion sections are led by TAs.

Approved GE: Quantitative or Formal Reasoning

Prerequisites: 22M:005 and MPT II score of 20 or higher or MPT III score of 10 or higher

22C:019 Discrete Structures

Description: This is an introduction to mathematical methods used in studying problems of computer science. The topics covered include: mathematical logic; proof techniques, especially mathematical induction; set theory, functions, and relations; procedures, recursion, and operation counts; recurrence relations; analysis of algorithms; counting methods, permutations and combinations; graphs and trees. Calculus I is recommended, but not required as a prerequisite. The course is required for all computer science majors and minors. Two one-hour exams and a final are given, plus weekly quizzes and weekly problem sets from the text. The course is taught by a faculty member. Calculus I is recommended. NOTE: This course may not be taken AFTER either 22C:021 or 22C:022. It must be taken before or concurrently with either course.

Approved GE: None

Prerequisites: Grade of C- or higher in 22C:016

22C:021 Computer Science II: Data Structures

Description: The second course required for computer science majors and minors emphasizes the design, implementation, and analysis of common data structures and algorithms. The goal is to teach how data structures provide the necessary data abstraction for the development of large software systems and their central role in software engineering. Data structures covered include sets, linked lists, stacks, queues, hash tables, trees, heaps, and graphs. Students are introduced to algorithms for searching, sorting, and data structure manipulation and learn the techniques to analyze program efficiency. Programming using recursion and dynamic data structures are covered. The programming language is Java or C++. The course involves a midterm and final exam and multiple programming and homework assignments. Lectures are taught by a faculty member; discussion sections are led by TAs.

Approved GE: None

Prerequisites: Grades of C- or higher in 22C:016

Pre- or

Corequisite: 22C:019

22C:022 Object-Oriented Software Development

Description: The course continues the presentation of skills and knowledge necessary for effective software development, building upon the basic competence students gained in 22C:016. The course focuses on developing professional-level programming competence using the object-oriented paradigm and associated concepts of classes, objects, methods, inheritance, and polymorphism. Students are taught the basics of software engineering and use that knowledge in a group project to develop a large

Guide to Courses

Department of Computer Science

Page 4

software system. The goal of this course is to educate students on how to develop large software using existing application programming interfaces (APIs) for graphics, networking, databases, multithreading, and user interfaces. The language of instruction is Java. The group project constitutes a major portion of the evaluation, but the course also involves a midterm and final exam and several programming projects. The course is required for all computer science majors and minors and may be taken concurrently with 22C:021. The lecture is taught by a faculty member; discussion sections are led by TAs.

Prerequisites: Grade of C- or higher in 22C:016

Pre- or

Corequisites: 22C:019

22C:031 Algorithms

Description: Topics include algorithm design techniques (divide and conquer, dynamic programming, greedy) and analysis techniques (big-O notation, recurrence); sorting (merge sort, heapsort, and quicksort); basic graph algorithms (depth-first and breadth-first search, minimum spanning trees, shortest paths); introduction to NP-completeness and approximation algorithms. There are several programming projects. The course is taught by a faculty member.

Prerequisites: Grades of C- or higher in 22C:021, and 22M:025 or 22M:031

22C:060 Computer Organization

Description: This course has two main goals: (1) to teach students how a digital computer works and (2) to introduce students to assembly language programming. The hardware component of the course begins by introducing the basic switching components of all digital circuits. It next shows how to analyze circuits and also how to build circuits that conform to specified computational properties. It introduces many standard circuits used by all computers, such as logic and shift circuits, arithmetic circuits, and memory circuits. The course then presents the design of a current general-purpose digital computer; class discussions cover the interface between the computer and its external devices. It presents alternatives in computer design, contrasting complex-instruction-set computers (CISCs) with reduced-instruction-set computers (RISCs). The software component of the course introduces assembly language programming. It describes instructions of classes including control-flow, load and store, operate (arithmetic, logical, and shift and rotate), stack, type conversion, privileged, and I/O. The course also describes addressing modes and their utility. The software component concludes by describing software to control hardware devices. Requirements include 8-10 homework assignments and two or three exams. The course is taught by a faculty member.

Prerequisites: Grades of C- or higher in 22C:021

22C:072 Elementary Numerical Analysis

Description: This course covers root-finding for polynomials and general algebraic equations; numerical solution of systems of linear equations; interpolation theory; numerical integration; computer arithmetic; numerical solution of ordinary differential equations; detailed error analysis of several techniques; and illustrative programming projects. The course is taught by a faculty member. Same as 22M:072

Prerequisites: Grade of C- or higher in 22M:022 or 22M:026 or 22M:032

22C:096 Topics in Computer Science

Description: Complements material in other courses. Recent examples include Advanced Java Programming and Introduction to XML. An instructor number and approval are required for registration in this course. Contact the instructor (or designated individual) for the instructor number, which you enter as the section number when you register. At the same time you should make the required semester hours, time, and place arrangements. It may be taken four times. Depending on content, this course may or may not count towards the computer science minor.

Prerequisites: Consent of instructor

22C:099 Honors in Computer Science

Description: This course allows for individualized work on honors projects. Open only to computer science majors in the honors program. Membership in the honors program is open to any undergraduate who maintains a UI grade-point average of 3.20 or higher. It may be taken four times.

Prerequisites: Consent of instructor

22C:104 Introduction to Informatics

Description: An overview of computing principles and fundamental aspects of computer science. Intended for graduate students planning to pursue computational aspects of their own disciplines. Topics covered include history of computing, basic computer architecture and operating system concepts, fundamentals of relational databases, algorithmic idioms, computational complexity, and introductory programming concepts in Perl. Not open to computer science students.

Prerequisites:

22C:109 Programming Languages and Tools

Description: This course offers rotating sections covering different programming languages (C, C++, C#, Java, COBOL) and tools (Visual Studio). The topics covered in a section vary by programming language. The goal of this course is to expose students to the latest technology and allow students to learn a new programming language once they understand the fundamentals of programming. Students are assumed to have previous programming experience at the level of 22C:016. The course does not count for credit towards a CS major or minor. The course is taught by an instructor or a TA.

Approved GE: None

Prerequisites: Some exposure to programming (e.g., 22C:005, 22C:016 or 22C:104), or consent of instructor

***22C:109: Programming Languages and Tools:
SCA Programming with COBOL***

Description: This course teaches the COBOL programming language and its use in business applications, records, files, and mass storage devices; programming techniques for table handling, sorting, generation of reports from files, and the maintenance of sequential and random-access files. The course is taught by an instructor or a TA.

***22C:109: Programming Languages and Tools:
SCB Programming With C***

Description: This course covers the use of major portions of the C programming language: variables, expressions, and statements; program modularization through functions, macros, and blocks; control structures; representation of numeric and textual data using scalar and structured data types; and operating system interfaces to files and other services. Programming methodology topics such as using program design and development tools and managing multi-file programs are examined. The course is taught by TAs.

***22C:109: Programming Languages and Tools:
SCC Programming with C++***

Description: Topics include basic constructs in C++; class specification; multiple inheritance, operator and function name overloading; virtual functions and templates; basic concepts of data abstraction and object-oriented programming in C++. This course is not intended for computer science majors who have C++ experience. The course is taught by TAs.

22C:109: Programming Languages and Tools:

SCC Programming with C++

Description: Topics include basic constructs in C++; class specification; multiple inheritance, operator and function name overloading; virtual functions and templates; basic concepts of data abstraction and object-oriented programming in C++. This course is not intended for computer science majors who have C++ experience. The course is taught by TAs.

22C:109: Programming Languages and Tools:

SCD Programming with Java

Description: This course covers the philosophy of Java, the Java language, objects and classes, applications in Java, packages and interfaces; exceptions, threads, the abstract window toolkit, applets, and programming for the web. This course is intended for non-CS majors who already know how to program in a language other than Java, and who want to learn to program in Java. Not open for credit to students who have taken 22C:016. The course is taught by a faculty member or a TA.

22C:109: Programming Languages and Tools:

SCE Programming in C#

Description: This course covers XXX

The course is taught by a faculty member or a TA.

22C:111 Programming Language Concepts

Description: This course examines advanced topics in programming languages, for example, syntax specification and informal semantic models; program control structures, including recursion, co-routines, backtracking, and concurrency; and data abstraction and structuring methods. The course introduces programming paradigms, such as functional and logic programming, that contrast with more conventional languages. Examples and projects may rely on several languages such as C, C++, Ada, and Java, with Prolog for the logic programming part and ML or Haskell for the functional programming part. Requirements include one or two in-class exams, final exam, and several computer projects. This is a required course for majors in computer science. The course is taught by a faculty member.

Prerequisites: Grades of C- or higher in 22C:019, 22C:021 and 22C:022

22C:112 Operating Systems

Description: This course provides an introduction to the design and implementation of modern operating systems. Students will learn how the operating system manages devices including I/O systems, secondary storage, and peripherals. Memory management is introduced including file systems, virtual memory, segmentation, paging, and addressing. Process scheduling and interprocess communication is covered including related issues of interrupts, deadlock, synchronization, and locks. Additional topics include security, library loading/linking, and privacy.

Prerequisites: Grade of C- or higher in 22C:060

22C:113 Introduction to Systems Software

Description: This course provides an introduction to the design and implementation of system software. Program-execution environments are described in terms of operating systems functions: command interpretation, process management, memory management, message management, file management, I/O device management, process synchronization, protection, and security. Also covered are efficient data structures and implementations of system components, programming support software (assemblers, compilers, linkers, loaders), and development of a conceptual framework for system-software design based on theoretical concepts illustrated by projects and assignments, including the design and implementation of selected system software components. Two midterm exams plus a final examination, two small projects, two large projects, a term report, and about eight homework assignments. This is a required course for majors in computer science. The course is taught by a faculty member.

Prerequisites: Grades of C- or higher in 22C:060

22C:118 Introduction to Networks and Their Applications

Description: This course introduces students to computer networks and the application techniques for using networks. The course surveys the basic concepts of messaging and media transfer. After an introduction to general concepts, the subject matter turns to important software applications that rely on networks: parallel simulation, network file sharing, multiplayer games, and distributed databases. The course is taught by a faculty member.

Prerequisites: Grade of C- or higher in 22C:060

22C:131 *Limits of Computation*

Description: Topics covered include Turing machines, undecidability, complexity classes, reductions, NP-completeness, NP-complete problems, randomized algorithms and randomized complexity classes, approximation algorithms and related complexity classes, dealing with NP-completeness. The course is taught by faculty

Prerequisites: Grade of C- or higher in 22C:031 or equivalent

22C:135 *Theory of Computation*

Description: Topics include finite automata; regular sets, expressions; context-free and context-sensitive grammars and their properties; push-down automata; standard, universal, and linear-bounded Turing machines; relationships between formal languages and automata; and undecidability and its consequences. Only very well prepared undergraduates should consider this course. Two in-class exams plus final and weekly homework is required. The course is taught by a faculty member.

Prerequisites: Grade of C- or higher in 22C:031 or equivalent

22C:137 *Theory of Graphs*

Description: Topics include connectivity properties such as Euler and Hamilton cycle problems; graph colorings, matchings; characterization of families of graphs such as trees, planar graphs, networks; and graph algorithms and their applications. The course is taught by a faculty member.

Same as 22M:152

Prerequisites: Grade of C- or higher in 22M:050 or equivalent

22C:141 *Knowledge Discovery*

Description: Topics include knowledge discovery process, including data reduction, cleansing, transformation; advanced modeling techniques from classification, prediction, clustering, association; evaluation and integration. The course is taught by a faculty member.

Same as 6K:275

Prerequisites: Consent of instructor

22C:142 Knowledge Discovery and Data Mining

Description: This course presents the concepts, algorithms, techniques, and systems in Knowledge Discovery and Data mining (KDD), including 1) data preprocessing, 2) data warehouse and OLAP systems, 3) frequent pattern and association analysis, 4) classification and prediction, and 5) clustering analysis. The course will serve mainly senior-level computer science undergraduate students and first-year graduate students interested in the field. The course may also attract students from other disciplines who need to implement and/or use data mining systems to analyze large amounts of data.

Prerequisites: 22C:031 or 22C:104, and some probability/statistics

22C:144 Database Systems

Description: This course provides an introduction to database systems including database querying, design, and programming. The course consists of three major components. The first component explains databases from a user perspective including how to query using SQL, relational algebra, and other query languages. The second component involves designing relational databases using Entity-Relationship (ER) diagrams and other modeling languages. The last part of the course involves database programming with current technologies such as JDBC, XML, PHP, and JSP. Students completing the course will have experience with current database technologies, and the ability to use and develop databases and associated applications. The course is taught by a faculty member.

Prerequisites: Grades of C- or higher in 22C:021 and 22C:022 or equivalent

22C:145 Artificial Intelligence

Description: Topics include problem-solving methods, state-space representations, heuristic search, problem-reduction techniques, machine inference, and game playing; knowledge representations; overviews of expert systems, language processing systems; and machine perceptions. Course requirements include weekly assignments, a midterm exam, a final and a term project. The course is taught by a faculty member.

Prerequisites: Grade of C- or higher in 22C:031 or equivalent

22C:146 Introduction to Computational Linguistics

Description: This course presents an overview of computational linguistics with a more in-depth discussion of illustrative topics such as finite state morphology, syntactic parsing with probabilistic context-free grammars, and word sense disambiguation using machine learning techniques. The focus is on written text processing, although there is some discussion of speech processing. Students also consider the similarities between computer language processing and human language processing.

Guide to Courses

Department of Computer Science

Page 11

A special emphasis is placed on empirical methods: corpora (how can one collect a corpus relevant to the task at hand and how should it be annotated?), evaluation (how can one test how good the system/theory is?), and analysis (how can one understand why a system/theory works or does not?). Numerous team projects are assigned and an attempt is made to balance project teams with respect to prior computer and linguistics experience. The goal is that such team projects foster peer learning. The course is taught by a faculty member. Same as 103:140.

Prerequisite: Consent of instructor

22C:151 Computer Graphics

Description: This course focuses on 3D interactive and non-interactive graphics. Includes coordinate systems, transformations, shading and lighting models, hidden-surface removal, ray tracing, radiosity, and rasterization. Course requirements usually include one exam and several programming or written homework assignments. The course is taught by a faculty member.

Prerequisites: Grades of C- or higher in 22C:031 and 22M:027 or equivalents

22C:160 High Performance Computer Architecture

Description: This course studies processor architectures: Von Neumann machine, evolutions in instruction set design, RISC and CISC, implementation of instruction set, microprogramming; storage systems; cache, main and secondary memory, virtual memory; I/O organizations; CPU design: instruction and arithmetic pipelines; high performance computers; array and vector processors, shared-memory and distributed-memory multiprocessors; and case studies from historic and current architectures. Two midterms and a final exam, and eight homework assignments. The course is taught by a faculty member.

Same as 055:132

Prerequisites: Grade of C- or higher in 22C:112 or 22C:113, or equivalent

22C:162 Advanced Operating Systems

Description: This course examines operating system support for sequential, concurrent, and distributed programming; interprocess communication and synchronization constructs: semaphores, sockets, monitors and remote procedure calls; and management and protection of memory and communication resources. Class grade is based on one midterm, a final exam, and weekly homework. The course is taught by a faculty member.

Prerequisites: Grades of C- or higher in 22C:031, 22C:111, and 22C:112 or 22C:113; or equivalent

22C:165 Parallel Programming

Description: This course deals with parallel programs from the perspectives of concept, design, implementation, performance evaluation, and the concept of process. It also covers parallel algorithms, language and architectural support, and development and running of parallel programs on available parallel machines. Two midterm exams are given and five two-week projects are assigned. One final project is used as the final exam.

Prerequisites: Grade of C- or higher in 22C:112 or 22C:113 or equivalent

22C:166 Distributed Systems and Algorithms

Description: This course is an introduction to distributed computing with emphasis on distributed algorithms. It covers techniques used in advanced distributed system services including election algorithms, distributed graph algorithms, fault tolerance, distributed agreement, global snapshots, object replication, logical time, self-stabilization, and multicast operations. The course is taught by a faculty member.

Prerequisites: Grade of C- or higher in 22C:162 or 22C:168 or equivalent

22C:168 Computer Communications

Description: Networks, ISO model, network topology, communication of digital data, data link control; errors, error control; point-to-point networks, broadcast networks, local area networks; transport services; wireless networking; internetworking; user services. Open only to seniors and graduate students in computer science or electrical and computer engineering. The course is taught by a faculty member.

Same as 055:134

Prerequisites: 22S:039 or 22S:120, and familiarity with C and Unix; or consent of instructor

22C:169 Computer Security

Description: Topics include mechanism versus policy distinction; authentication mechanisms, access control mechanisms, and security domains; perimeter security, internal security and defense in depth; communication security, cryptographic protocols, key management and key distribution; threat models, security assessment methods, and clarification of secure systems. The course is taught by a faculty member.

Prerequisites: Grade of C- or higher in 22C:060 or equivalent

22C:170 Numerical Analysis Nonlinear Equation Approximation Theory

Description: Topics include the treatment of the general idea of error, rootfinding methods for nonlinear equations, interpolation theory, approximation of functions, and numerical integration. Along with 22M:171, the two courses form a general introduction to numerical analysis, and may be taken in either order. The course is taught, in small sections, with approximately 25 students per class. Grades are based on a midterm and a final exam; homework problems count for a significant portion of the grade. The course is recommended for well-prepared undergraduates. Same as 22M:170

Prerequisites: 22M:027 and 22M:028; or 22M:037, or 22M:056; and computer programming knowledge; or consent of instructor

22C:171 Numerical Analysis: Differential Equations and Linear Algebra

Description: This course covers numerical methods for initial-value problems for ordinary differential equations; direct and iterative methods for linear systems of equations; and eigenvalue problems for matrices. Same as 22M:171

Prerequisites: 22M:027 and 22M:028, or 22M:037, or 22M:056; 22M:100; and computer programming knowledge, or consent of instructor

22C:174 Optimization Techniques

Description: Topics include the basic theory of optimization, use of numerical algorithms in the solution of optimization problems; linear and non-linear programming, sensitivity analysis, convexity, optimal control theory, dynamic programming, and calculus of variations. Grading is based on homework and exams. The course is taught by faculty. Same as 22M:174

Prerequisites: 22M:100 or equivalent

22C:177 High Performance and Parallel Computing

Description: Diverse aspects of high performance and scientific computing. The core of the class is on parallel scientific computing methods, such as parallel algorithms for dense and sparse matrix solvers and implementation of these algorithms using standard libraries such as MPI. Theoretical issues from parallel algorithms are also covered. In some semesters we may include current issues in parallel computing, such as grid computing. Same as 22M:178

Prerequisites: A linear algebra course or a numerical analysis course, and a programming language.

22C:180 Fundamentals of Software Engineering

Description: This course is an introduction to software engineering process. Students learn about software development process models and examine life cycle phases: planning, problem analysis, requirements definition, specification, design, implementation, testing, maintenance and project management. Students complete a semester-long group project. Approximately seven homework assignments and two exams are given. The course is taught by a faculty member.

Same as 055:180

Prerequisites: Senior undergraduate or graduate standing in electrical and computer engineering or computer science

22C:181 Formal Methods in Software Engineering

Description: This course introduces formal models, methods, and their application in various phases of software engineering process. The purpose of formal methods is to enable the construction of large, highly reliable software. Their foundation is the precise specification of the run-time properties that a software system is expected to satisfy. The course presents a collection of techniques for formal software development including operational, algebraic, model-based, property-based specification methods; for the verification of consistency and completeness of specifications; for the verification of software properties. Course work includes exercises in specification construction and verification, both on paper and using formal method-based tools. The course is taught by a faculty member.

Same as 055:181

Prerequisites: Grade of C- or higher in 22C:180 or consent of instructor

22C:182 Software Engineering Languages and Tools

Description: The class focuses on models and methods, their application in all phases of software engineering process; operational, algebraic, model-based, property-based specification methods; verification of consistency, completeness of specifications; verification of software properties; exercises in specification construction, and verification using method-based tools.

Same as 055:182

Prerequisites: Grade of C- or higher in 22C:180 or C++ programming experience, and consent of instructor

22C:183 Software Engineering Project

Description: This course includes the use of object-oriented concepts and object-based models in software system analysis and design; Booch, OMT, and Booch-Rumbaugh unified method and notation; Jacobson's use cases; use of design patterns; software architectures; case studies; team project for a real software project; object-oriented process and project management. The course is taught by a faculty member.

Same as 055:183

Prerequisites: 22C:181 and 22C:182; or consent of instructor

22C:185 Programming Language Foundations

Description: This is an introductory course on the formal foundations of programming languages. Its overall goals are to expose students to established techniques for providing precise and implementation independent definitions of programming languages, to help student develop deeper insights into key concepts and paradigms in programming, and to investigate methods for constructing correct programs and rigorously proving their properties. The formal foundations are presented using a variety of models, including attribute grammars, operational, axiomatic, denotational, and algebraic techniques. They involve the generation of proofs of program equivalence, correctness, and termination. The course is taught by a faculty member.

Prerequisites: Grades of C- or higher in 22C:031 and 22C:111 or equivalents

22C:186 Introduction to Compiler Construction

Description: Topics explored are concept, design, and implementation; simple one-pass compiler; lexical analysis (token specification and recognition, automatic scanner generation); syntax analysis (context-free grammars, top-down, bottom-up, and operator precedence parsing, LL- and LR-parser techniques, treating ambiguous grammars, error recovery); intermediate code generation (postfix notation, three-address code, syntax trees); and code optimization (local, global, loop); and a large programming project.

Prerequisites: Grades of C- or higher in 22C:031, 22C:111, and 22C:112 or 22C:113, or equivalents

22C:189 Software Engineering Project Management

Description: Resource requirement estimation, planning, management; risk analysis; scheduling, tracking, control; personnel supervision, training, evaluation; process determination and management, including change control, configuration management; technical project leadership, assessment; participation in management of projects and teams in 22C:183.

Prerequisites: Grade of C- or higher in 22C:182 and 22C:183, and consent of instructor

22C:191 Research for Thesis

Description: Open only to MS degree candidates in computer science. May be taken two times.

Prerequisites: Consent of advisor required

22C:196 Topics in Computer Science

Description: This course complements material in other courses. Recent offerings have included Advanced Java Programming, E-Voting: Computers in Elections, and Computational Combinatorics. May be taken four times.

Prerequisites: Consent of instructor

22C:197 Readings in Computer Science

Description: Material in this course is not covered in other courses. Students interested in studying subjects not covered in other courses may negotiate with a faculty member for individualized instruction. May be taken six times.

Prerequisite: Consent of instructor

22C:198 Individual Programming Projects

Description: Students interested in developing a unique programming project may negotiate with a faculty member for individualized instruction. May be taken three times.

Prerequisites: Consent of instructor

22C:231 Design and Analysis of Algorithms

Description: Review of design and analysis techniques; advanced data structures; graph algorithms (network flows, matching, min-cut); NP-completeness, randomization and approximation algorithms; special topics (string matching, computational geometry or number theoretic algorithms). There will be three exams and two projects. The course is taught by a faculty member.

Prerequisites: Grades of C- or higher in 22C:031 or 22C:131 or equivalent

22C:242 Data Mining and Machine Learning

Description: This course will present selective topics in data mining and machine learning including machine learning theory, SVM, and kernel methods. This course will also explore the state-of-the-art in data mining and machine learning by discussing research papers and performing research projects. Students will practice research skills, learn to formulate and present ideas, and write a research paper. The course is taught by a faculty member.

Prerequisites: Grades of C- or higher in 22C:021 and 22C:022 or equivalents

22C:244 Database System Implementation

Description: This course expands on 22C:144 to cover advanced database implementation and design topics including file organizations, storage management, database system architectures, query optimization, transaction management, recovery, and concurrency control. Additional topics including distributed databases, mobile databases, and integration may also be covered. A major component of the course is a database implementation project using current database languages and systems. Grades are determined by a midterm, final, several homework assignments, and the project. The course is taught by a faculty member.

Prerequisites: 22C:144 or equivalent

22C:245 Advanced Artificial Intelligence

Description: May include theorem proving, concept formation, AI programming languages and concepts, machine understanding, robot models, philosophies of machine intelligence.

Prerequisites: 22C:145 or equivalent

22C:251 Advanced Computer Graphics

Description: Topics such as global illumination and rendering; volume rendering; animation; curves and surfaces, advanced modeling and mapping techniques; graphics hardware; real-time graphics for virtual environments.

Prerequisites: 22C:151 or consent of instructor

22C:253 Algorithms in Discrete Optimization

Description: This course starts where 22C:231 leaves off, with NP-completeness and approximation algorithms. Given that most natural computational problems seem to be NP-complete, devising efficient approximation algorithms has become the main focus of algorithms research in the past decade. In this course we will study problems, techniques, and results that occupy an important place in the landscape of approximability.

Problems such as Set Cover, Steiner Trees, TSP, Knapsack, Job Scheduling, Facility Location, Multicommodity Flow, k-Median, and MAXSAT from the point of view of approximability are considered. The most commonly used techniques are studied, such as the greedy method, local search, LP-rounding, primal-dual methods, semidefinite programming, and low distortion embeddings of metric spaces. Using these and other techniques, polynomial time approximation schemes (PTAS), constant-factor approximation, and log-factor approximations are devised for the above problems.

Along with positive algorithmic results, the PCP theorem and results relating to the hardness of approximation of some of these problems are also studied. Time permitting, the course ends with a study of approximate counting and the use of Markov Chain Monte Carlo (MCMC) methods in devising efficient approximate counting algorithms.

Prerequisites: 22C:231 or consent of instructor

22C:290 Readings for Research

Description: Individualized instruction for PhD candidates in computer science, covering subjects not covered in other courses.

Prerequisites: Consent of instructor

22C:294 Seminar on Systems and Networks

Description: Advanced topics in systems and networks; for example, fault-tolerate distributed systems. This course includes classification and modeling of failures; dealing with crash failures, Byzantine failures, and omission failures; distributed consensus; group communication; replicated data management; transient failures and self-stabilizing systems; clock synchronization; and distributed simulation.

Prerequisites: Consent of instructor

22C:295 Seminar on Artificial Intelligence

Description: This seminar presents advanced topics in artificial intelligence and contributes to prepare the students to undertake research in the field. Each offering of the seminar will typically focus on some sub area of AI, depending on the research interests of the instructor and students. During the semester, the instructor and students will read and discuss research papers, textbook chapters and/or research monographs on the selected topic. The students may be required to give oral presentations of such materials and/or written summaries. The seminar in general has no exams, but there may be individual or group projects implementing some of the approaches studied during the seminar.

Prerequisites: Consent of instructor

22C:296 Seminar on Computer Science

Description: Various topics in computer science, not covered in other courses. Recent examples: Seminar on Human Computer Interaction, Parallel Algorithms, and Topics in Database Management Systems.

Prerequisites: Consent of instructor

22C:298 Seminar on Programming Languages

Description: This seminar examines recent developments in programming languages, broadly construed. This may include object-oriented, functional, logic, constraint, or specification languages. Each offering will focus on a paradigm, and may pursue language and program design, syntax and semantics, program specification and verification, or implementation issues.

Prerequisites: Consent of instructor

22C:299 Research for Dissertation

Description: Open only to PhD candidates in computer science.

Prerequisites: Consent of instructor

Guide to Courses

Department of Computer Science

Page 20

22C:391 Research Seminar: Algorithms

Description: Presentation and discussion of research papers in algorithm design and analysis. Repeatable. Open only to PhD candidates.

Prerequisites: 22C:153 or consent of instructor

22C:394 Research Seminar: Distributed Systems

Description: Discussion of distributed systems theory. Repeatable.

Prerequisites: 22C:194 or 22C:294 or consent of instructor

22C:398 Research Seminar: Programming Languages

Description: Discussion of distributed systems theory. Repeatable.

Prerequisites: Consent of instructor

22C:399 Research Seminar: Colloquium Series

Description: Presentations by internal and external speakers. Repeatable.

Prerequisites: None
