

CANADA COLLEGE



Software Engineering Program

Office of Admission

1118 Sainte Catherine West #405

Montreal, Quebec, H3B 1H5, Canada

Tel: 514-868-6262, 514-707-3456, 514-935-3106, 1-886-868-6262

Fax: 514-868-6262

www.collegecanada.com

apply@collegecanada.com

Software Engineering

Program Description

Diploma Program

There is a tremendous need for technical experts with the ability to create innovative computer systems and for highly trained professionals to manage these systems. The Computer Science program is offered for individuals who are interested in applying, designing, and implementing computer systems. Graduates of the Software Engineering program are prepared to seek a wide variety of technical positions, including systems programmer, systems analyst, software engineer, database administrator, webmaster, and networking engineer, or admission to universities in a related field of studies. Students are provided with a sound theoretical and practical background coupled with the skills to understand, develop and use theories. The specific goal of the program is to graduate highly-trained computer professionals who have a foundation in algorithm development and software engineering.

The Computer Science curriculum provides all graduates with a foundation in programming, algorithm development, computer architecture, operating systems, and networks through a set of core courses. The curriculum also allows specialization through the choice of one of two different options: Software Engineering or Management Information Systems. Courses ranging from introductory programming courses to courses in Computer Architecture, Software Design and Construction, Client/Server Programming, Human Computer Interaction, Web Application Development, Systems Programming, and Data Communication are available.

The Computer Science program is committed to the belief that curriculum must reflect both theory and actual professional experience. Full-time professors have held high-level positions in industry, and most adjunct faculty are currently employed as computer professionals.

How is the program structured?

Typically, students progress through the program with their entering classmates. This gives students the opportunity to build personal and business support networks and lasting friendships. The program encompasses an integrated curriculum of courses totaling 60 credit hours. The schedule depends on the country that CANADA COLLEGE branch is offering this program. In order to receive their diploma, students have to pass 20 courses in total, either in a 24 months (2 semesters per year) option or in 18 months.

Diploma in Computer Science course sequence

For course sequencing information, please get the information form your local office.

What background is required to apply?

Entrance into the Diploma program in Computer Science requires a high-school diploma or a collegial diploma from your country of residence, regionally-accredited school. It is recommended that the diploma be completed within the last five years. However, the selection committee will review this requirement on a case-by-case basis. Applicants should also be familiar with Windows Operating System, and have typing skill of at least 30 words per minute.

Academic Honesty

Students enrolled in Canada College of Education courses are expected to observe the same code of academic honesty required of other CANADA COLLEGE students. Violation of this code can result in academic penalties, such as receiving a failing grade in the course and other disciplinary actions. Academic dishonesty includes, but is not limited to, cheating on examinations and plagiarism, the latter meaning offering the language or ideas of someone else as one's own. Plagiarism may range from failure to credit isolated formulas, sentences, paragraphs, or ideas to entire articles copied from books, periodicals, speeches, or the writings of other students.

Situations involving academic misconduct for credit courses will be reviewed by a CANADA COLLEGE Educational Committee on Academic Conduct. This committee is composed of advisory board members for the certificate/diploma program in which the student is enrolled. If evidence of academic misconduct is established, students will be given a failing grade for the course and any refund of tuition fees will be denied

Program Options

The computer science major requires students to complete core courses, which include structured programming and the principles and design of information systems. After completing the core courses, students have their choice of three emphases based on personal interests and abilities. These emphases are 1) hardware systems, 2) systems administration and 3) information systems.

Admission Requirement

- Be at least 18 years of age.

- Have a high-school diploma with acceptable GPA and passing mark of at least %80 in last year mathematics, Physics and English.

- Submit a cheque of \$50 application fee. This fee is required and should be attached to the application for admission. It is nonrefundable. This fee will be waived for those taking a first time single class and for those customized training courses paid for by the company sponsoring the class.

Application Filing Periods

Each campus accepts applications until capacities are reached. Many campuses limit admission in an enrollment category because of overall enrollment limits. If applying after the initial filing period, consult the campus admissions office for current information. In general:

- Applications for the fall semester are accepted beginning Oct. 1. The initial filing period lasts until Nov. 30.
- Applications for the spring semester are accepted beginning Aug. 1. The initial filing period lasts until Aug. 31.
- Applications for the summer semester are accepted beginning Feb. 1. The initial filing period lasts until Feb. 28.

Note: *The above dates might slightly be varied in different countries.*

Applications postmarked or received during the initial filing period will be given equal consideration within established enrollment categories and quotas. There is no advantage in filing before the initial filing period. Applications received before the initial filing period may be returned, causing a delay in processing.

Application Acknowledgment. You may expect to receive an acknowledgment of your application. The notice may also include a request that you submit additional records necessary for the campus to evaluate your qualifications. You may be assured of admission if the evaluation of your qualifications indicates that you meet CANADA COLLEGE admission requirements and campus. An offer of admission is not transferable to another term or to another campus.

Language of Instruction

CANADA COLLEGE's language of instruction is French and or English. CANADA COLLEGE's branches overseas are operating with either English or French. This is depending on the country's language of instruction. However in most countries, we offer our programs in English.

TOEFL or CET Requirement. All applicants whose native language is not English and who have not attended schools at the secondary level or above for at least three years full-time where English is the principal language of

instruction must present a score of 500 or above on the Test of English as a Foreign Language. Applicants taking the Computer-Based Test of English as a Foreign Language must present a score of 173 or above. Students, who do not have the TOEFL, have to sit for **Canadian English Test (CET)**, developed by CANADA COLLEGE and offered in different countries that CANADA COLLEGE currently has its branches offering different programs of studies. This exam is offered with the consent from the Canadian Embassy. Notice that in countries such as Iran, TOEFL has been banned since 1979 and students seeking admission into a foreign university has to travel to the neighboring countries such as Turkey, UAE or Syria to write this exam.

If you have successfully completed CANADA COLLEGE's ESL program and have received %80 or better need not to write **TOEFL** or **CET**.

Note: In their first year, students are obliged to take and pass English Language for Academic Purposes offered by our ESL department.

Grading System:

Letter Grade	Percentage	Grade Point
A+	97-100	4.5
A	91-96	4.0
A-	85-90	3.7
B+	75-84	3.5
B	70-74	3.0
C+	65-69	2.5
C	60-64	2.0
D	50-59	1.0
F	<50	0.0

Course ID	Course Title	Duration	Credit(s)
CS 101	Introduction to Computing Evaluation: Lab works: 5 * %10 = %50 Midterm: 1 * %15 = %15 Final: %35	45	3
ESL 260	English Language for Academic Purposes Evaluation: Lab works: 4 * %8 = %32 Midterm: 1 * %18 = %18 Final: %50	45	3
CS 110	Programming Laboratory Evaluation: Lab works: 4 * %25 = %100 Midterm: NA Final: NA	43	2
CS 125	Introduction to Computer Science Evaluation: Lab works: 4 * %10 = %40 Midterm: 1 * %15 = %15 Final: %45	45	3
CS 173	Discrete Mathematical Structures Evaluation: Lab works: NA Midterm: 1 * %25 = %25 Final: %75	28	1

CS 225	Data Structures and Software Principles Evaluation: Lab works: 3 * %10 = %30 Midterm: 1 * %20 = %20 Final: %50	45	3
CS 231	Computer Architecture I Evaluation: Lab works: 5 * %10 = %50 Midterm: 1 * %15 = %15 Final: %35	45	3
CS 232	Computer Architecture II Evaluation: Lab works: 5 * %8 = %40 Midterm: 1 * %15 = %15 Final: %45	45	3
CS 257	Numerical Methods Evaluation: Assignments: 5 * %5 = %25 Midterm: 1 * %20 = %20 Final: %55	45	3
CS 311	Database Systems Evaluation: Assignments: 4 * %5 = %20 Midterm: 1 * %20 = %20 Final: %60	45	3
CS 314	Multimedia Systems Evaluation: Assignments: 4 * %8 = %32 Midterm: 1 * %18 = %18 Final: %50	45	3

CS 335	Computer-Assisted Instruction Evaluation: Assignments: 4 * %10 = %40 Midterm: NA Final: %60	45	3
CS 318	Computer Graphics Evaluation: Assignments: 4 * %5 = %20 Midterm: 1 * %20 = %20 Final: %60	45	3
CS 319	Advanced Topics in Computer Graphics Evaluation: Assignments: 4 * %5 = %20 Midterm: 1 * %20 = %20 Final: %60	45	3
CS 321	Programming Languages and Compilers Evaluation: Assignments: 4 * %5 = %20 Midterm: 1 * %30 = %30 Final: %50	45	3
CS 322	Programming Language Design Evaluation: Assignments: 4 * %5 = %20 Midterm: 1 * %20 = %20 Final: %60	45	3
CS 323	Operating Systems Design Evaluation: Assignments: 3 * %5 = %15 Midterm: 1 * %20 = %20 Final: %65	45	3

CS 326	Compiler Construction Evaluation: Assignments: 4 * %5 = %20 Midterm: 1 * %20 = %20 Final: %60	45	3
CS 327	Software Engineering I Evaluation: Assignments: 4 * %5 = %20 Midterm: 1 * %30 = %30 Final: %50	45	3
CS 328	Computer Networks and Distributed Systems Evaluation: Assignments: 4 * %5 = %20 Midterm: 1 * %20 = %20 Final: %60	45	3
CS 329	Software Engineering II Evaluation: Assignments: 4 * %5 = %20 Midterm: 1 * %30 = %30 Final: %50	45	3
CS 330	Communication Networks Evaluation: Assignments: 4 * %5 = %20 Midterm: 1 * %30 = %30 Final: %50	45	3
	22 Courses	971 h	63 Credits

English Language for Academic Purposes (ELAP)

Duration: 45 hours

The English Language for Academic Purposes courses are a sequential study of the grammar and vocabulary necessary for academic work in English. By taking these courses, you will begin building up your ability to recognize correct structures in English. Then you will progressively work on producing more and more complex grammar.

Students preparing for study at any university where English is the medium of instruction will benefit from the English Language for Academic Purposes (ELAP) courses.

All of the exercises have academic contexts. The vocabulary is from the University List, a compilation of the most common sub-technical vocabulary in English across all academic disciplines. The words are used throughout the lessons. There are also exercises to practice using targeted vocabulary correctly.

Course Introduction

The purpose of this class is to improve your grammar and vocabulary in English for academic contexts. If you do well in this class, you will be better prepared to do well in courses where English is the medium of instruction.

Target Goals

By the end of this course, you should be able to recognize the following:

- identify these parts of speech in sentences: nouns, pronouns, verbs, adjectives, descriptive adverbs (including comparative and superlative forms), prepositions, and modal auxiliaries
- identify subjects, verb phrases, objects, object of prepositions, and prepositional phrases
- state the number of clauses in a sentence
- identify and correct incomplete sentences and comma splices
- recognize and name the four basic verb tenses: simple present, simple past, present continuous, and past continuous
- recognize and name four perfect tenses: present perfect, present perfect continuous, past perfect, and past perfect continuous
- recognize common errors in meanings of modals
- recognize noncount and count nouns
- recognize the meanings of level 1 words or phrases from the University List, a compilation of the most common sub-technical vocabulary in academic textbooks

You should also be able to control these structures:

- identify and correct incomplete sentences and comma splices
- recognize and correct incorrect expressions of future time
- recognize and correct impossible verb forms
- recognize and correct incorrect modal constructions
- control
 - existential **there** and false subject **it**
 - fragments, run-ons, and comma splices
 - simple present, present continuous, simple past, and past continuous in a paragraph
 - subject-verb agreement
 - irregular verbs
 - present perfect and present perfect continuous with **for** and **since**
 - causatives
 - gerunds and infinitives as subjects and as objects of verbs and prepositions
 - the list of common noncount nouns
 - the "singular count noun cannot go bare" rule
 - plural **-s**

101 Introduction to Computing

Fundamental principles, concepts, and methods of computing, with emphasis on applications in the physical sciences and engineering. Basic problem solving and programming techniques; fundamental algorithms and data structures; and use of computers in solving engineering and scientific problems.

Course Objectives This is an introductory computer science course intended primarily for physical science and engineering students. It emphasizes fundamental concepts and algorithms of use in engineering and the physical sciences.
General Education Requirement This course meets the campus general education requirement.
Prerequisite Grade 12 high school.
Credits : 3
Format 2 hours of lecture and 1 hour of lab-discussion per week
Semester Usually offered fall and spring. See your local time table.
Course Web Site Ask your local course administrator.
Recent Textbook: Required: <i>Applied C: An Introduction and More;</i> by Fischer, Eggert, & Ross; McGraw-Hill, 2000; 0-07-021748-3 <i>Getting Started with Matlab: A Quick Introduction for Scientists and Engineers;</i> by Pratap; Oxford University Press, 2001; 0-19-515014-7 Recommended: <i>Course Notes Packet;</i> by Gambill; Stipes Publishing, 2003
Laboratory Work Computer programming in the C language on the College workstation computers.

Course Spec Table	
Hours	Topics
3	Introduction to computing concepts.
4	Basics of computer programming. Constants, variables, expressions, precedence of operators, assignment, basic input and output, built-in functions.
3	Structured programming ideas: sequence, selection, iteration.
3	Selection statements: conditional expressions; if-then-else statements; case statements.
3 ½	Iteration statements: counting loops, while loops.
2 ½	ASCII data files; file pointers; end-of-file.
4	Modular programming: Functions with and without return values; actual and formal parameters.
4	Arrays, matrices, and vectors. One- and two-dimensional arrays; simple sort algorithms; matrix addition, subtraction, multiplication, transposition.
3	Characters and character strings.
4	Pointers and addressing; dynamic memory allocation.
13	Numerical methods. Linear interpolation; linear regression; pseudo-random numbers; roots of functions; solution of simultaneous linear equations by Gaussian elimination; numerical integration. (This section is dispersed appropriately throughout the semester to illustrate the above techniques.)

110 Programming Laboratory

Practical laboratory course in the methods used and skills required for writing and maintaining well-structured software. Extensive practice with a programming language is provided. Different sections use different programming languages. An existing knowledge of fundamental computing principles is assumed.

Course Objectives This course provides instruction in a computer language to students who have previously mastered the fundamentals of computer programming in any computer language.
General Education Requirement No
Prerequisite It is recommended that students enrolling in the CP (C++) section of this course have prior C programming experience in the C section of this course.
Credit 2 hour. Students may not receive credit for studying any given language more than once. (That is, they may take a given section only once.) Students may register for more than one section in the same semester. Students may not receive credit both for CS 101 and also for the "C" section of CS 110. Students may not receive credit both for CS 125 and also for the Java section of CS 110.
Format 3 laboratory hours per week
Semester Usually offered fall and spring.
Course Web Site Ask your local course administrator.
Recent Textbook Required (Java Section): <i>Beginning Java 2 - JDK 1.3 Version;</i> by Horton; Wrox Press; 1861003668 updated to reflect Spring 2003 selections
Laboratory Work Computer programming on the College workstation computers.

Course Spec Table

Hours	Topics
5	Language constructs: variables, assignments, loops, decision structures, input/output, files, subprograms/procedures, numeric and nonnumeric data.
10	Design and construction of software: top down and bottom up design, decomposition, structuring, design for reuse, documentation, study of examples, writing software as a team, using software from others.
28	Programming assignments: a variety of progressively more complex assignments.

125 Introduction to Computer Science

First course for computer science majors and other students with a deep interest in computing. The course introduces students to basic concepts in computing and fundamental techniques for solving computational problems.

Course Objectives
This is the introductory computer programming course intended primarily for those with a major or minor in computer science, computer engineering, or electrical engineering. It uses the Java language, and prepares the student for more advanced work in the 200-level computer science courses. This course is appropriate both for students with little or no previous computer programming experience, and for those who have previously taken an introductory programming course.
General Education Requirement
No
Prerequisite
Three years of high school mathematics or Math.
Credit
3 hours
Format
3 hours of lecture and 2 hour of lab-discussion per week
Semester
Usually offered fall, spring and summer.
Course Web Site
Ask your local course administrator.
Recent Textbook
Recommended: <i>An Introduction to Computer Science Using Java;</i> by Kamin, Mickunas, & Reingold; McGraw-Hill; 0-07-232305-1 <i>Beginning Java 2 - JDK 1.3 Version;</i> by Horton; Wrox Press; 1861003668 **updated to reflect Spring 2003 selections
Laboratory Work
Computer programming in Java on Windows NT workstations computers.

Course Spec Table	
Hours	Topics
2	Introduction to hardware and software (processor and memory and machine language through high-level languages).
7	Fundamental data storage and manipulation (types and variables, statements and expressions, and arrays).
7	Flow of control (Boolean expressions, conditional statements, and loops).
5	Classes (classes and objects, instance variables and instance methods, and encapsulation).
7	Inheritance (subclasses, dynamic binding, abstract classes, and interfaces).
	Introduction to recursion.
6	Searching and sorting algorithms (linear and binary search, selection sort, insertion sort, and quick sort - introduced via recursive versions).
5	Linked lists (linked memory, recursive list methods, and merge sort on linked lists).
1	Algorithm analysis.

173 Discrete Mathematical Structures

Studies discrete mathematical structures frequently encountered in the study of computer science. Topics will include sets, propositions, boolean algebra, induction, recursion, relations, functions and graphs.

Course Objectives This course provides the mathematical foundation for the study of the theory of computation for those who completing a major or minor in computer science.
General Education Requirement No
Prerequisite None
Credit 1 hours.
Format 2 hours of lecture per week
Semester Usually offered fall, spring and summer.
Course Web Site Ask your local course administrator.
Recent Textbook Required: <i>Discrete Mathematics and Its Applications</i> , 5th edition; by Rosen; McGraw-Hill; 0-07-242434-6 **updated to reflect Spring 2003 selections
Laboratory Work

Course Spec Table	
Hours	Topics
3	Sets
4	Propositions
4	Two-valued boolean algebra
3	Inductions
6	Recursion
4	Relations and functions
4	Graphs

225 Data Structures and Software Principles

Data abstractions: elementary data structures: lists, stacks, queues, trees; searching and sorting techniques. Introduction to the principles of software engineering, including semester programming project.

Course Objectives This is the foundation course in data structures and computer software for those who completing a major or minor in computer science or computer engineering. Electrical engineering students may find it useful to take this course.
General Education Requirement No
Prerequisite CS 125 (strongly recommended) or both "C" section CS 110 and junior standing; CS 173, or consent of instructor.
Credit 3 hours
Format 3 hours of lecture per week, 2 hours of lab/discussion per week, plus unscheduled lab work.
Semester Usually offered fall and spring.
Course Web Site Ask your local course administrator.
Recent Textbook Required: <i>Data Structures and Their Algorithms</i> , 1st edition; by Lewis & Denenberg; Addison-Wesley; 0-673-39736-X Recommended: <i>The C++ Programming Language</i> , 3rd edition; by Stroustrup; Addison-Wesley; 0-201-70073-5 **updated to reflect Spring 2003 selections
Laboratory Work Computer programming in C++ under Windows 2000 OS.

Course Spec Table	
Hours	Topics
2	Introduction
3	Program design concepts
4	Abstract data types
7	Basic data structures
13	Abstract data types for sets: operations and implementations
5	Sorting
3	Memory management
4	Graph algorithms
4	String algorithms

231 Computer Architecture I

Introduction to computer architecture, working up from the logic gate level: combinational and sequential networks; computer arithmetic; arithmetic/logic units; memory organization; control unit design.

Course Objectives This is the first computer architecture (logic design) course for computer science majors.
General Education Requirement No
Prerequisite CS 125
Credit 3 hours.
Format 2 hours of lecture and 3 hours of lab per week
Semester Usually offered fall, spring and summer.
Course Web Site Ask your local course administrator.
Recent Textbook Required: <i>Logic and Computer Design Fundamentals and Xilinx 4.2 Package;</i> by Mano & Kime; Prentice Hall; 0-13-179473-6 **updated to reflect Spring 2003 selections
Laboratory Work Schematic capture and logic simulation on computer workstations.

Course Spec Table	
-------------------	--

Hours	Topics
9	Combinational logic networks
8	Computer arithmetic; arithmetic/logic unit
9	Sequential logic networks
6	Memory hierarchy
8	CPU design
6	I/O architecture

232 Computer Architecture II

Second-level course in computer architecture: machine-level programming, instructions sets, data representations; subroutines; input/output hardware and software; linking and loading; relation to high-level languages.

Course Objectives This course provides an introduction to the basics of computer hardware organization, instruction execution, and the relationships between higher-level programming languages and machine language.
General Education Requirement No
Prerequisite CS 231
Credit 3 hours.
Format 2 hours of lecture and 1 hour of discussion per week
Semester Usually offered fall and spring.
Course Web Site Ask your local course administrator.
Recent Textbook Required: <i>Computer Organization and Design: The Hardware/Software Interface;</i> by Hennessy & Patterson; Morgan Kaufmann Publishers; 1-55860-428-6 **updated to reflect Spring 2003 selections
Laboratory Work Computer programming on Sun workstations running Unix.

Course Spec Table	
Hours	Topics
4	Introduction and review
6	Instruction sets, addressing modes, linking and loading
4	Subroutines
6	ALU design
7	Basic processor design
5	Basic pipelining
7	Memory hierarchy design
3	Input/output
3	Parallel processing

257 Numerical Methods

Introduction to numerical methods for students in science and engineering; topics include floating-point computation, systems of linear equations, approximation of functions and integrals, the single nonlinear equation, and the numerical solution of ordinary differential equations; discusses various applications in science and engineering; includes some programming as well as the use of high quality mathematical library routines.

Course Objectives This is a foundation course in numerical analysis for students in this program.
General Education Requirement
Prerequisite A 100-level computer science course; Math 225.
Credit 3 hours.
Format 3 hours of lecture per week
Semester Usually offered fall, spring and summer.
Course Web Site Ask your local course administrator.
Recent Textbook Required: "Getting Started with Matlab a Quick Introduction for Scientists and Engineers" ISBN 0-19-515014-7, Rudra Pratap, Oxford University Press "Numerical Methods Using Matlab" ISBN 0-13-012641-1, George Lindfield/John Penny, Prentice Hill
Laboratory Work Computer programming using Mathematica or Matlab on Windows 2000 OS.

Course Spec Table

Hours	Topics
3	Basics; numerical algorithms and mathematical software. Sources and propagation of errors. Condition of problems and stability of algorithms (qualitative notions).
5	Floating-point computation
7	Systems of linear equations; Gaussian elimination and triangular factorization. Ill-conditioning. Error estimates. Least squares approximation.
8	Interpolation; existence and uniqueness of the interpolating polynomial. Lagrange form of the interpolating polynomial. Newton form. Error in polynomial interpolation.
7	Numerical integration. Approximation based on the interpolating polynomial including Gauss rules. Adaptive quadrature.
8	The single nonlinear equation; convergence of functional iteration. Bisection, and Newton's method and its variants including secant.
8	Initial-value problems for ordinary differential equations; differential equations; Taylor methods. Runge-Kutta methods.

311 Database Systems

Examines the logical organization of databases: the entity- relationship model; the hierarchical, network, and relational data models and their languages. Functional dependencies and normal forms. Design, implementation, and optimization of query languages; security and integrity; concurrency control, and distributed database systems.

Course Objectives This course provides a thorough background in database concepts, as well as practical information on database system design.
General Education Requirement No
Prerequisite CS 225 or consent of instructor.
Credit 3 hours.
Format 3 hours of lecture per week
Semester Usually offered fall only.
Required: <i>Database Systems: The Complete Book</i> , ISBN 0-130-319-953, Hector Garcia Monila, Jeffrey Ullman, Jennifer Widom, Prentice Hall, 1st Edition
Laboratory Work Computer programming using database management packages such as Informix, Sybase, Oracle, FoxPro, and Encina on PCs.

Laboratory Work

Computer programming using database management packages such as Informix, Sybase, Oracle, FoxPro, and Encina on PCs, Sun workstations, and HP 800 computers.

Course Spec Table

Hours	Topics
5	Introduction: an overview of a database management system. The entity-relationship model.
10	Logical organization of databases: the relational model. Relational algebra. SQL. Examples of existing relational database management systems.
7	Physical organization of databases. Characteristics of disks and disk blocks. Storage of relations. Indexing: B-trees and hashing.
4	Query processing and optimization
10	Concurrency control, transactions, serializability, locking, logging and recovery.
4	Distributed databases
2	Functional dependencies and normal forms
3	Information services for unstructured data

314 Multimedia Systems

Organization and structure of modern multimedia systems; audio and video encoding; quality of service concepts; scheduling algorithms for multimedia within OS and networks; multimedia protocols over high-speed networks; synchronization schemes; user-interface design; multimedia teleservices.

Course Objectives
This course discusses the design of modern multimedia systems, emphasizing the integration of processing and communication concepts for high-quality support of continuous media such as audio and video
General Education Requirement
No
Prerequisite
CS 225.
Credit
3 hours. Additional work is required.
Format
3 hours of lecture per week.
Semester
Usually offered spring only.
Course Web Site
Ask your local course administrator.
Recent Textbook
Required: <i>Multimedia Fundamentals, Volume 1: Media Coding and Content Processing</i> , 2nd edition; by Nahrstedt & Steinmetz; Prentice Hall; 0-13-031399-8 <i>Multimedia Fundamentals, Volume 2: Media Processing and Communications</i> , 2nd edition; by Nahrstedt & Steinmetz; Prentice Hall; 0-13-046187-3 **updated to reflect Spring 2003 selections
Laboratory Work
Computer programming on the multimedia laboratory computers.

Course Spec Table

Hours	Topics
15	Audio and video. Basic concepts: sampling, Nyquist theorem, quantization, image characteristics, audio and video characteristics (telephone, CD, NTSC, PAL, HDTV qualities). Compression: lossless and lossy coding, run-length coding, Huffmann and Arithmetic Encoding, JPEG, MPEG.
7	Multimedia operating systems issues. Quality of service concept, guaranteed services, negotiation, monitoring, adaptation, translation. Resource management: admission, reservation, and allocation. Linear arrival process model, real-time system model, guaranteed scheduling for processor and disks, multimedia file systems, buffer management for continuous media, integrated multimedia device management, case studies: quick time, Windows, OS/2.
9	Multimedia networking and communication. Bandwidth reservation schemes; asynchronous, synchronous, isochronous services; case studies: Fast Ethernet, FDDI, DQDB, ATM. Network guaranteed scheduling, traffic shaping, error detection and correction schemes for continuous media, multimedia network and transport protocols; case studies: RSVP, ST-II, UDP/RTP (internet protocols). Multimedia session protocols, conferencing, floor control, application sharing.
6	Synchronization. Live and synthetic synchronization, lip synchronization. Reference model for multimedia synchronization. Synchronization specification: timed petri nets. Case studies: MHEG, Hytime, Firefly, Mode.
5	User interface. Information characteristics for presentation, presentation function, presentation design knowledge, effective human-computer interaction. Audio and video at the user interface.
6	Teleservices. Conversational services - video conferencing. Messaging services: multimedia mail (MIME, X.400). Retrieval services: video on demand, video servers, web services. Interactive TV.

335 Computer-Assisted Instruction

Computer-assisted instruction (CAI) and its relation to classroom teaching; the teacher's role in development, management, and criticism of CAI lessons; treatment of topics including instructional capabilities of CAI systems, instructional programming, and the design of CAI lessons.

Course Objectives This course discusses the use of the computer in delivering instruction, from the point of view of a teacher
General Education Requirement No
Prerequisite 100-level computer science course or consent of instructor
Credit 4 hours.
Format 3 hours of lecture and 2 hours of laboratory per week
Semester Usually offered fall, spring and summer.
Course Web Site Ask your local course administrator.
Recent Textbook Bernard Poole, <i>Education for an Information Age - Teaching in the Computerized Classroom</i> , 2nd edition, McGraw Hill, 1997.
Laboratory Work Observation and evaluation of CAI lessons; exercises with instructional tools; execution of an instructional project.

Course Spec Table

Hours	Topics
3	Computers in the classroom; historical perspective and overview
6	Eliciting episodes; a basic element of interactive instruction
6	Evaluating instructional software
12	The computer as teacher. Tutoring. Drill or practice. Computer-based testing. Computer management of instruction.
9	The computer as instructional partner. Simulation. Instructional games.
6	The computer as classroom tool
3	Defining the role of computers in an instructional environment

318 Computer Graphics

Software, hardware, and mathematical tools for the representation, manipulation, and display of topological and two- and three-dimensional objects, applications of these tools to specific problems.

Course Objectives This is a first course in the hardware and software design of computer graphics systems.
General Education Requirement No
Prerequisite CS 225, and understanding of analytic geometry
Credit 3 hours.
Format 3 hours of lecture per week
Semester Usually offered fall only.
Course Web Site Ask your local course administrator.
Recent Textbook Required: <i>Interactive Computer Graphics: A Top-Down Approach Using OpenGL</i> (bundled with <i>OpenGL: A Primer</i>), 3rd edition; by Angel; Addison-Wesley; 0-321-14955-6 Recommended: <i>OpenGL Programming Guide</i> , 3rd edition; by Woo et al.; Addison-Wesley; 0-201-60458-2 **updated to reflect Spring 2003 selections
Laboratory Work Computer programming on HP and SGI workstations.

Course Spec Table

Hours	Topics
1	Introduction: overview of graphics systems.
4	Components of graphics systems, display devices, processors, software standards, introduction to GKS (Graphical Kernel System), PHIGS (Programmer's Hierarchical Interactive Graphics System), and OpenGL.
6	Basic raster algorithms, generation of output primitives, attributes (color, area filling, etc.), geometric transformations.
7	Structure of graphics packages, two-dimensional viewing, structures/segments, hierarchical model, graphical user interfaces, interactive input methods.
6	Three-dimensional object representations and manipulations, polygon mesh, spline surfaces, superquadrics, fractal geometry, octrees, visualization of three-dimensional data sets, geometric transformations.
6	Three-dimensional viewing, parallel and perspective projections, three-dimensional view volumes, clipping.
5	Visible surface identification methods, depth-buffer, scan-line, depth sorting, area subdivision, octree, and ray-casting methods.
7	Illumination models and surface rendering, constant intensity, Gouraud shading, Phong shading, ray tracing, radiosity.
3	Color models, basic concepts; RGB, CMY, HSV, HLS models.

319 Advanced Topics in Computer Graphics

Advanced methods for representing, displaying, and rendering two-, three-, and four-dimensional scenes. General algebraic curves and surfaces, splines, Gaussian and bump-function representations, fractals, particle systems, constructive solid geometry methods, lighting models, radiosity, advanced ray-tracing methods, surface texturing, animation techniques, data visualization methods.

Course Objectives This course provides advanced study of computer graphics representation schemes and rendering algorithms.
General Education Requirement No
Prerequisite CS 318
Credit 3 hours.
Format 3 hours of lecture per week
Semester Usually offered spring only.
Course Web Site Ask your local course administrator.
Recent Textbook Required: <i>Advanced Animation and Rendering Techniques, 1st edition;</i> by Watt & Watt; Addison-Wesley; 0-201-54412-1 **updated to reflect Spring 2003 selections.
Laboratory Work Computer programming on HP workstation computers .

Course Spec Table

Hours	Topics
4	Review of computer graphics fundamentals
4	Nonparametric object representations, conics, algebraic surfaces, bump functions.
6	Parametric object representations, quadrics, superquadrics, splines.
7	Non-Euclidean representations, fractals, particle systems.
6	Rendering, lighting models, fast-Phong algorithm, A-buffer, V- buffer, radiosity.
5	Ray-tracing algorithms, distributed methods, space subdivision, parallel methods.
3	Texture mapping
7	Animation, key-frame systems, animation languages, kinetic vs. dynamic systems, modeling human and animal motion.
3	Scientific data visualization

321 Programming Languages and Compilers

Introduction to the structure of programming languages and their implementation. Basic language design principles: abstract data types (lists, arrays, user-defined types); functional languages; type systems; object-oriented languages. Basics of lexing, parsing, syntax-directed translation, semantic analysis, and code generation.

Course Objectives This course covers the principle modern programming language paradigms - imperative, functional, object-oriented, and logic - and their implementation. It will enable students to undertake a complete language development project, from informed design through efficient implementation.
General Education Requirement No
Prerequisite CS 225 and CS 231.
Credit 3 hours.
Format 3 hours of lecture per week
Semester Usually offered fall and spring.
Course Web Site Ask your local course administrator.
Recent Textbook Required: <i>Compilers Principles, Techniques, and Tools</i> ; by Aho, Sethi, & Ullman; Addison-Wesley; 0-201-10088-6 **updated to reflect Spring 2003 selections
Laboratory Work Computer programming on HP workstations.

Course Spec Table

Hours	Topics
	Language Design
2	Elements of imperative languages
7	Data types: arrays, lists, user-defined types
6	Functional programming
3	Control operations
4	Object-oriented programming
3	Types
	Compilation
4	Lexical analysis: transition diagrams, regular expressions, using lex
7	Syntactic analysis: context-free grammars, top-down and bottom-up parsing, using yacc
3	Abstract syntax; syntax-directed translation
6	Code generation

322 Programming Language Design

Advanced course in principles of language design. Using imperative and functional programming as unifying themes, major language design paradigms will be explored. Tools in this study will include both practical language processor construction and theoretical models. Emphasis will be on reasoning about programs and languages.

Course Objectives This course teaches advanced principles of computer programming language design. It includes major programming language paradigms in a unified framework, providing a sound practical and theoretical basis for programming language design. It emphasizes the view that programs and languages are mathematical structures amenable to rigorous analysis
General Education Requirement No
Prerequisite CS 321
Credit 3 hours.
Format 3 hours of lecture per week
Semester Usually offered fall only.
Course Web Site Ask your local course administrator.
Recent Textbook Required: <i>The Formal Semantics of Programming Languages: An Introduction;</i> by Winskel; MIT Press; 0-262-23169-7 <i>Essentials of Programming Languages</i> , 2nd edition; by Friedman, Wand, & Haynes; MIT Press; 0-262-06217-8 **updated to reflect Spring 2003 selections
Laboratory Work Computer programming on Sun Sparc workstations.

Course Spec Table	
Hours	Topics
	Operational symantics
6	Call-by-value languages (Scheme)
3	Call-by-name languages (Lambda-Calculus)
4	Objects
3	Inheritance
	Analysis of programs and languages
6	Functional languages - equational reasoning, type checking, abstract data types
6	Imperative languages - Hoare logi
3	Procedures in imperative languages
3	Objects and side effects
	Advanced topics
3	Continuations - control operations, recursion elimination
5	Concurrency - CSP, actors
3	Logic programming

323 Operating Systems Design

The organization and structure of modern operating systems and concurrent programming concepts. Deadlock, virtual memory, processor scheduling, and disk systems. Performance, security, and protection.

Course Objectives This course discusses the design of modern computer operating systems and concurrent programming.
General Education Requirement No
Prerequisite CS 225 and CS 232.
Credit 3 hours.
Format 3 hours of lecture per week
Semester Usually offered fall and spring.
Course Web Site Ask your local course administrator.
Recent Textbook Required: <i>Modern Operating Systems</i> , 2nd edition; by Tanenbaum; Prentice Hall; 0-13-031358-0 **updated to reflect Spring 2003 selections
Laboratory Work Computer programming on Sun Sparc workstations.

Course Spec Table

Hours	Topics
14	Processes and concurrent programming. Basic concepts: states, transitions. Mutual exclusion, synchronization, semaphores, monitors, Ada rendezvous. Deadlock and indefinite postponement; prevention, avoidance, detection, recovery.
16	Operating system components. Real and virtual memory; paging and segmentation; fetch, placement, and replacement algorithms; thrashing. Processor scheduling; disk space management and allocation; seek and rotational optimization; blocking and buffering. File systems; directory structures; access methods; access control.
15	Advanced topics. Performance evaluation. Distributed and parallel operating systems. Object orientation. Security and protection; encryption. Case Studies.

326 Compiler Construction

Compiler structure, syntax analysis, syntax-directed translation, automatically constructed recognizers, semantic analysis, code generation, intermediate language, optimization techniques.

Course Objectives This course introduces the principles, techniques, and tools for the design and construction of compilers for modern programming languages
General Education Requirement No
Prerequisite CS 232 or and CS 321.
Credit 3 hours.
Format 3 hours of lecture per week
Semester Usually offered spring only.
Course Web Site Ask your local course administrator.
Recent Textbook Required: <i>Compilers: Principles, Techniques, and Tools;</i> by Aho, Sethi, & Ullman; Addison-Wesley; 0-201-10088-6 **updated to reflect Spring 2003 selections
Laboratory Work Computer programming on HP 400 computers.

Course Spec Table

Hours	Topics
2	Context-free languages and grammars.
6	Bottom-up parsing.
3	Syntax-directed translation.
3	Storage allocation.
4	Review of symbol tables, type checking, semantic analysis.
3	Project logistics.
	Code generation.
1	Basic blocks/dags.
2	Expressions.
2	Instruction selection, optimization, integrated techniques.
	Control and data flow
1	Flow graphs, dominators.
2	Iterative and interval analysis.
2	Def-use, use-def, live variable analysis.
2	Dead code, redundant computation elimination.
2	Constant propagation, strength reduction.
2	Program representations (SSA, PDG).
3	Loop optimization.
2	Register allocation.
5	Advanced topics. arbage collection. Dynamic data structures, pointer analysis, aliasing. Code scheduling, pipelining. Dependence testing. Loop level optimization. Superscalar optimization. Profile-driven optimization. Debugging support. Incremental parsing. Type inference. Advanced parsing algorithms (Tomita/Early). Practical attribute evaluation. Function in-lining and partial evaluation.

327 Software Engineering I

Software process analysis and design. Topics include software development paradigms, system engineering, function-based analysis and design, and object-oriented analysis and design. The course will use team-projects for hands-on exercises. Builds on basic programming skills (CS 125, CS 225) to introduce concepts of software engineering and programming-in-the-large.

Course Objectives To teach principles, models and techniques of software analysis and design.
General Education Requirement No
Prerequisite CS 225
Credit 3 hours.
Format 3 hours of lecture per week
Semester Usually offered fall only.
Course Web Site Ask your local course administrator.
Recent Textbook Required: <i>The Engineering of Software: A Technical Guide for the Individual</i> ; by Hamlet & Maybee; Addison-Wesley; 0-201-70103-0 <i>Writing Effective Use Cases</i> , 1st edition; by Cockburn; Addison-Wesley; 0-201-70225-8 <i>UML Distilled: A Brief Guide to the Standard Object Modeling Language</i> , 2nd edition; by Fowler & Scott; Addison-Wesley; 0-201-65783-X Recommended: <i>The Rational Unified Process: An Introduction</i> , 2nd edition; by Kruchten; Addison-Wesley; 0-201-70710-1 <i>Extreme Programming Explained: Embrace Change</i> , 1st edition; by Beck; Addison-Wesley; 0-201-61641-6 **updated to reflect Spring 2003 selections
Laboratory Work Computer programming on HP workstations

Course Spec Table	
Hours	Topics
2	Software development paradigms
3	Software process
5	System Engineering. Planning, software economics, prototyping and simulation
2	Fundamentals of system analysis and design
14	Function-based analysis and design. Data modeling, flow modeling, behavioral modeling, formal specification, DF-oriented design, DS-oriented design
10	Object-oriented analysis and design. E-R modeling, use-case analysis, patterns, frameworks
4	Interface design and HCI
5	Analysis and design of real-time, distributed and multi-media systems

328 Computer Networks and Distributed Systems

Covers topics needed for a basic understanding of distributed computer systems: Protocols, specification techniques, global states and their determination, reliable broadcast, transactions and commitment, security, and real-time systems.

Course Objectives The course objective is to provide a solid introduction to distributed systems from which students can go on to jobs, research, etc. This course builds on a basic operating systems course to treat the additional complexities of communications, mutual exclusion, etc., needed to manage resources in a distributed environment. It looks in depth at the design of operating systems for distributed and parallel systems.
General Education Requirement No
Prerequisite CS 323 or consent of the instructor
Credit 3 hours.
Format 3 hours of lecture per week
Semester Usually offered fall only.
Course Web Site Ask your local course administrator.
Recent Textbook Required: <i>Distributed Systems: Concepts and Design</i> , 3rd edition; by Coulouris, Dollimore, & Kindberg; Addison-Wesley; 0-201-70103-0 Recommended: <i>Distributed Systems: Principles and Paradigms</i> , 1st edition; by Tanenbaum & van Steen; Prentice Hall; 0-13-088893-1 **updated to reflect Spring 2003 selections
Laboratory Work None

Course Spec Table

Hours	Topics
5	Introduction. What is a distributed system? Advantages, problems
5	Networks. Protocols, layered architecture, packet switching vs. ATM
5	Specifying systems, especially distributed systems and protocols; state machines, Petri nets.
5	Global states of distributed systems. Definitions. "Consistent" state sequences and global state diagrams. "Viewing" the global state: monitors, snapshots, etc.
5	Fault-tolerant (reliable) broadcast. Types, uses, implementation
5	Transactions. Distributed databases, recovery, concurrency control, atomic commitment
4	Non-blocking Atomic Commitment. Implementations, including use of reliable broadcast.
4	Security. Models. Extra problems in a distributed system. Intro to encryption. Implementing secure channels -- including Kerberos-type schemes
4	Real-time distributed systems.
3	Implementation of distributed systems: Standards, such as distributed computing environment (DCE), CORBA, etc.

329 Software Engineering II

Software development, management and maintenance. Topics include project and configuration management, collaborative development models, software quality assurance, interoperability domain engineering and software reuse, and software re-engineering. This course is a follow-up to CS 327 (Software Engineering I), and completes the sequence by in-depth coverage of implementation and post-development issues of software engineering.

Course Objectives This course teaches principles, models and techniques of software development, maintenance, and re-engineering.
General Education Requirement No
Prerequisite CS 327
Credit 3 hours.
Format 3 hours of lecture per week
Semester Usually offered spring.
Course Web Site Ask your local course administrator.
Recent Textbook Required: <i>The Engineering of Software: A Technical Guide for the Individual;</i> by Hamlet & Maybee; Addison-Wesley; 0-201-70103-0 Recommended: <i>Style: Toward Clarity and Grace</i> , 1st edition; by Williams; University of Chicago Press; 226899144 **updated to reflect Spring 2003 selections
Laboratory Work Computer programming on HP workstations

Course Spec Table

Hours	Topics
6	Software project and configuration management
4	Software development issues
5	Software metrics
4	Reliability, security and performance assurance
9	Software testing and verification
7	Development and maintenance of real-time, distributed and multi-media systems
4	Domain modeling and software reuse
5	Software re-engineering
2	Case tools

338 Communication Networks

Introduction to International Standards Organization Open System Interconnection (ISO-OSI) reference model, design issues and protocols in the physical layer, data link layer and network layer; architectures and control algorithms of local-area networks, point-to-point networks and satellite networks; standards in network access protocols; models of network interconnection; and overview of networking and communication software.

Course Objectives This course discusses the characteristics and low-level protocols of computer networks. It provides basic background, design, and evaluation skills in telecommunication and communication networks. (See CS 328 for a course on high-level communication protocols and distributed system software.)
General Education Requirement No
Prerequisite CS 231
Credit 3 hours.
Format 3 hours of lecture per week
Semester Usually offered fall and spring.
Course Web Site Ask your local course administrator.
Recent Textbook Required: <i>Computer Networks: A Systems Approach</i> , 2nd edition; by Peterson & Davies; Rosen; McGraw-Hill; 0-07-242434-6 Recommended: <i>UNIX Network Programming - Volume I: Networking APIs: Sockets and XTI</i> , 2nd edition; by Stevens; Prentice Hall; 0-13-490012-X <i>Internetworking with TCP/IP: Principles, Protocols, and Architectures</i> , 4th edition; by Comer; Prentice Hall; 0-13-018380-6 <i>Computer Networks</i> , 4th edition; by Tanenbaum; Prentice Hall; 0-13-066102-3 **updated to reflect Spring 2003 selections
Laboratory Work None

Course Spec Table

Hours	Topics
2	Overview. Examples and concepts of layered architecture; overview of higher layer protocols.
9	Transport layer. Internet addressing and Internet protocols; socket interface; TCP/IP protocols; client-server models.
3	Network layer. Taxonomies; relevant parameters of network and traffic.
3	Performance evaluation and queuing theory
11	Multiple-access methods for broadcast networks. Taxonomies of multiple access methods; contention methods; polling methods; reservation methods.
11	Switched networks. Architectures of switches: circuit, packet, and ATM switches; scheduling and admission control; routing, flow control, and congestion control.
3	Interconnections of networks
3	Logical data link protocols