

# Computer Science

College of Engineering and Computer Science

**Chair: Steven Stepanek**  
**Jacaranda Hall (JD) 4503**  
**(818) 677-3398**  
**www.csun.edu/compsci/**

## Staff

Julie Corelli (Office Manager), Jennifer Wong

## Faculty

Jack Alanen, Prasanta Barkataki, Michael Barnes, Richard Covington, Steven Fitzgerald, Peter Gabrovsky, Robert Lingard, Richard Lorentz, Robert McIlhenny, Gloria Melara, John Noga, Son Pham, Diane Schwartz, Steven Stepanek, Brenda Timmerman, Ginter Trybus, George (Taehyung) Wang, Jeff Wiegley, Bahram Zartoshty

## Emeritus Faculty

Philip Gilbert, Ruth Horgan, Larry Lichten, Dorothy Miller, John Motil, David Salomon

## Programs

Undergraduate:

**B.S., Computer Science**  
**Minor in Computer Science**

Graduate:

**M.S., Computer Science**  
**M.S., Software Engineering**

## The Major

Computing technology has an impact on almost every aspect of daily life. Computer applications abound in art, business, entertainment, science, engineering and medicine. For students who think logically, enjoy solving problems and have an interest in software development, Computer Science is a good study choice.

Students develop skills in logical thinking, creative problem-solving and communication. Classes often incorporate a team approach, requiring clear communication among members as they solve a problem and explain their solution to others.

Students gain both hands-on design experience as well as theoretical knowledge. This combination of skills provides an advantage to graduating students because of the broad range of skills possessed.

Classes are generally small, averaging less than 25 students. The faculty work on such fascinating topics as virtual reality, high-speed networks, parallel computing, computer security, embedded applications, Internet technologies and multimedia. Students work alongside faculty in department labs equipped with state-of-the-art computing equipment.

Students can gain extra experience in the Student Chapter of the ACM (affiliated with the national organization), which hosts technical and social activities as well as the Honors Co-op Program which provides paid internships during the senior year at local companies.

## Academic Advisement

Contact the Department Office regarding undergraduate advisement. Graduate students are initially advised by the Graduate Coordinator, Richard Lorentz. After the formation of their Graduate Committees, graduate students are advised by the Committee Chair.

## Educational Objectives

The education objectives of the Bachelor of Science in Computer Science are to ensure that each graduate:

1. Understands the principles of computer science and problem solving.

2. Has an awareness of computing practices in industry and emerging technologies, emphasizing a working knowledge of current software design and development techniques.
3. Understands the impact of computing technologies in a societal context.
4. Has an education that enables them to pursue rewarding professional careers, graduate studies and lifelong learning.

## Student Learning Outcomes of the Undergraduate Program

Graduates of the Bachelor of Science in Computer Science at California State University, Northridge will be able to:

- a. Demonstrate an understanding of algorithms and data structures.
- b. Demonstrate an understanding of computer organization and architecture.
- c. Demonstrate proficiency in using a high-level computer language.
- d. Demonstrate an understanding of programming language concepts and knowledge of a variety of programming paradigms.
- e. Demonstrate an ability to apply mathematical skills appropriate to the computer science discipline.
- f. Demonstrate an ability to analyze a problem, and identify and define the computing requirements appropriate to its solution.
- g. Demonstrate an ability to apply knowledge of computing to design, implement and evaluate a computer-based solution to a problem to meet desired needs.
- h. Demonstrate an understanding of emerging technologies and a working knowledge of currently available software tools, techniques and skills necessary for computing practice.
- i. Demonstrate an understanding of the principles and practices for software design and development.
- j. Apply mathematical foundations, algorithmic principles, computer science theory and software engineering practices in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices.
- k. Apply the principles and practices for software design and development in the construction of software systems of varying complexity.
- l. Effectively communicate orally with a range of audiences.
- m. Effectively communicate in written form with a range of audiences.
- n. Work effectively on a team to accomplish a common goal.
- o. Analyze the local and global impact of computing on individuals, organizations and society.
- p. Demonstrate an understanding of the professional, ethical, legal, security and societal responsibilities with respect to computing.
- q. Demonstrate the knowledge and capabilities necessary for pursuing a professional career or graduate studies.
- r. Recognize the need for, and show an ability for, continuing professional development.

## Student Learning Outcomes of the Graduate Program in Computer Science

Graduates of the Master of Science in Computer Science at California State University, Northridge will be able to:

- a. Demonstrate a knowledge and competence in fundamental areas of computer science such as: algorithms, design and analysis, computational theory, computer architecture and software based systems.
- b. Demonstrate the analytic skills necessary to effectively evaluate the relative merits of software and computer systems, and algorithmic approaches.
- c. Demonstrate a breadth of knowledge in a choice of application areas in computer science, including: networks, artificial intelligence, graphics, human computer interfaces, databases, embedded applications and information security.
- d. Understand computer science topics (such as database management,

- data security, program efficiency, etc.) in a global context (ethics, privacy, human expectations, etc.)
- e. Effectively communicate in both written and oral form, especially in areas related to computer science.
  - f. Work productively in team or collaborative settings to achieve common goals or purposes including the ability to lead a team.
  - g. Analyze, evaluate and synthesize research and apply theoretical ideas to practical settings.
  - h. Independently continue their studies in computer science throughout their life.

### Student Learning Outcomes of the Graduate Program in Software Engineering

Graduates of the Master of Science in Software Engineering at California State University, Northridge will be able to:

- a. Understand software engineering concepts, techniques, practices and tools, and apply them to real problems in a variety of contexts.
- b. Define and apply a software process to large-scale real-world problems including requirements analysis and specification, software design and implementation, verification, validation and quality assurance, and the maintenance of software.
- c. Analyze and estimate software process costs and manage software development from concept to delivery.
- d. Identify, analyze and apply software standards in software engineering practice.
- e. Analyze, assess and interpret professional codes of ethics and regulatory documents pertaining to software engineering and understand societal issues.
- f. Generate and apply appropriate solutions to solve problems based on reasoned rationale.
- g. Work productively in term or collaborative settings to achieve common goals or purposes including the ability to lead a team.
- h. Analyze, evaluate and synthesize research and apply theoretical ideas to practical settings.
- i. Effectively present ideas, designs and solutions in a logical framework in a variety of forms with proper language structure and mechanics, and to produce appropriate written documentation.
- j. Recognize the need for, and show an ability for, dealing with constantly changing technology and continuing professional development.

### Careers

A computer science major can have a career as a software engineer, designing, implementing, testing and maintaining large software systems. Careers are available in such specialties as computer graphics, computer security, robotics, expert systems, distributed systems, embedded applications, network applications, and networking. The degree can lead to a career in almost any industry, including aerospace, manufacturing, banking, health, information technology, entertainment and education.

### Programs Offered

The undergraduate program, leading to a B.S. in Computer Science, provides a broad knowledge of computing. It consists of core courses in programming languages, computer system organization, operating systems, data structures, computation theory, and societal implications in computing.

The freshman year program includes courses in calculus, algorithms and programming, computer architecture and assembly language, data structures and program design, and symbolic logic.

Sophomores take courses in linear algebra, computer organization, programming language concepts, and advanced data structures.

As juniors, students take courses in combinatorial algorithms,

operating systems, software engineering, discrete mathematics, probability or statistics, automata, languages and computation.

With the help of an advisor, seniors choose 15 units of senior electives related to their career objectives. They also take a course in societal issues in computing.

A minor in computer science calls for 31 units of study, including courses in computer architecture and assembly language, algorithms and programming, data structure and program design, computer organization, programming language concepts, and advanced data structures, along with a choice of electives.

Students in the M.S. program complete 30 units of graduate work, including 6 units involving a thesis or graduate project.

The core of the graduate program comprises advanced courses in computation theory, algorithms and data structures, system architecture, and software engineering. The electives may be chosen to form a concentration in an area of specialization or to provide a broadly based program of study, whichever is more consistent with the selected thesis or graduate project.

### Scholarships and Awards

The College of Engineering and Computer Science administers a substantial scholarship program, dispersing over \$60,000 each year to high-achieving engineering and computer science students. The College also administers memorial scholarships and scholarships donated by friends of the University.

Applications and information are available in January with applications due in early March. Specific dates and further information can be obtained from the College administrative offices.

### Honors Cooperative Internship Program

The College offers an opportunity for highly qualified students to work in the local industry throughout an entire calendar year. Students work full-time during the summer and half time during the academic year. Students receive 6 units of academic credit in conjunction with this experience. The program is open to undergraduates who are nearing their senior year, have a minimum 3.0 cumulative GPA and have passed the Upper Division Writing Proficiency Exam. Graduate students who wish to participate must have a minimum 3.5 cumulative GPA. Applicants are matched to employer-supplied job descriptions and scheduled for interviews with prospective employers. The competitive nature of the program usually generates more applicants than there are positions available. The application period begins in early March and the period of employment is typically from July 1 through June 30.

### Bachelor of Science

The B.S. in Computer Science program requires a total of 120 units, including general education requirements, pre-major core, major core, and a 15 unit sequence of elective courses referred to below as the senior electives. A Computer Science major must complete a minimum of 18 residency units of Upper Division computer science courses. These must include 12 units of senior electives, in addition to all other institutional residency requirements.

### Requirements for Admission to the Major

To qualify for admission into the Computer Science major program, students must first complete a pre-major program in Computer Science consisting of seven (7) lower division courses covering math, computer science and the university General Education requirements for Basic Skills.

## Grade Requirements for Admission into the Computer Science

### Major Program:

1. C or better in each of the seven courses taken to satisfy the requirements of the Computer Science pre-major program.
2. Overall GPA of 2.0 in all courses taken at CSUN.

After successfully completing all requirements for the pre-major in Computer Science (including GE Basic Skills), students may apply for admission into the Computer Science major program by completing a pre-major to major evaluation form available from the Computer Science Department office. Admission into the Computer Science major program is required prior to enrolling in Upper Division Computer Science courses.

### Special Grade Requirements

Note: No grade lower than a C will be accepted on transfer from another institution to satisfy Computer Science requirements. Where specific grade requirements are not specified, no CSUN grade lower than a C- will be accepted for courses required in the Computer Science program.

#### 1. Lower Division Required Courses (35 Units)

- a. The following seven lower division courses constitute the Computer Science Pre-Major program:

General Education Basic Skills:  
Analytical Reading/Expository Writing  
Oral Communication

Note that the remaining components of GE Basic Skills are satisfied by the requirements of the Computer Science program.

COMP	110/L	Introduction to Algorithms and Programming and Lab (3/1)
COMP	122/L	Computer Architecture and Assembly Language and Lab (1/1)
COMP	182/L	Data Structures and Program Design and Lab (3/1)
MATH	150A	Calculus I (5)
PHIL	230	Symbolic Logic I (3)

- b. The following five lower division courses are part of the requirements of the Computer Science Major program. Prior to enrolling in these courses, students must complete all of the Computer Science Pre-Major requirements listed above. Computer Science Pre-Major students may not enroll in the following courses without the consent of the instructor.

COMP	222	Computer Organization (3)
COMP	232	Concepts of Programming Language (3)
COMP	282	Advanced Data Structures (3)
MATH	150B	Calculus II (5)
MATH	262	Introduction to Linear Algebra (3)

#### 2. Lower Division Electives (12-14 Units)

**Select one of the following science sequences (8-10 units):**

PHYS 220A/L, 220B/L (3/1), (3/1)  
BIOL 106/L, 107/L (3/1), (3/1)  
CHEM 101/L, 102/L (4/1), (4/1)

*Note: BIOL 107/L has a recommended prerequisite of CHEM 101/L.*

**Select an additional science course with corresponding lab outside of the sequence selected above (4-5 units):**

BIOL 106/L (3/1)  
CHEM 101/L (4/1)  
GEOG 101/102 (Lab) (3/1)  
GEOG 103/105 (Lab) (3/1)  
GEOL 101/102 (Lab) (3/1)  
GEOL 110/112 (Lab) (3/1)  
PHYS 220A/L (3/1)

#### 3. Upper Division Required Courses (22 Units)

Before taking Upper Division courses in Computer Science, students must be admitted to the Computer Science major/minor programs or the Computer Engineering major program.

Note that all students must attempt the Upper Division Writing Proficiency Exam prior to enrolling in any 400 level Computer Science course. The Upper Division Writing Proficiency Exam must be passed prior to enrolling in COMP 450.

COMP	310	Automata, Languages and Computation (3)
COMP	322/L	Introduction to Operating Systems and System Architecture and Lab (3/1)
COMP	380/L	Introduction to Software Engineering and Lab (2/1)
MATH	326	Discrete Mathematics (3)
MATH	341	Applied Statistics I (3)
COMP	450	Societal Issues in Computing (3)
MATH	482	Combinatorial Algorithms (3)

#### 4. Upper Division Electives (15 Units)

Computer Science majors are required to take 15 units of senior electives. The senior electives must consist of 15 units of 400 or 500-level courses in computer science (not including COMP 450, COMP 494 or COMP 499). The electives may include MATH 481A (Numerical Analysis) as 3 of the 15 units.

Requests for taking a 400 or 500-level course as a senior elective that does not meet the requirements stated above must be approved by the student's faculty advisor and by the Department Chair prior to enrollment in the course.

A student may wish to discuss their career goals with an advisor prior to selecting their senior electives. The advisor will suggest appropriate sequences of courses for the student to select from.

**General Education:** Computer Science majors follow a modified general education program depending upon the year they enter the program and their enrollment status as a college student. Returning and transfer students should consult an advisor before planning their general education programs. The requirements for students entering in Fall 2006 under the new GE Plan described here.

Computer Science students are required to take courses in the following GE sections: Analytical Reading and Expository Writing (3 units), Oral Communication (3 units), Social Sciences (6 units), Arts and Humanities (6 units), Comparative Cultural Studies (6 units), U.S. Government and History (6 units). Nine units of the GE requirements must be Upper Division (300+) courses that are certified as writing intensive. Two GE courses must meet the Information Competence requirement. All other GE requirements are met through completion of courses in the major.

Total Units in the Pre Major and Major	84-86
General Education Units	30
Additional Units	4-6
<b>Total Units Required for a B.S. Degree</b>	<b>120</b>

### Minor in Computer Science

Students who wish to get a minor in computer science must seek advice from a department advisor and get their minor program approved by the Department Chair before they begin taking any of the 200, 300 or 400-level elective courses. There are many prerequisites in the major and students need to choose course sequences for which they have or will have the prerequisites or the consent of the instructor. With the approval of the department chair, students may substitute an additional 3-unit 300 or 400 or 500-level Computer Science course for COMP 122/L.

**1. Minor Core\* (10 units)**

COMP	110/L	Introduction to Algorithms and Programming and Lab (3/1)
COMP	122/L	Computer Architecture and Assembly Language and Lab (1/1)
COMP	182/L	Data Structures and Program Design and Lab (3/1)

**2. Select one course (3 units)**

COMP	222	Computer Organization (3)
COMP	232	Concepts of Programming Languages (3)
COMP	270/L	Business Programming (2/1)
COMP	282	Advanced Data Structures (3)

**3. Select one course (3-4 units)**

COMP	310**	Automata, Languages and Computation (3)
COMP	322/L	Intro to Operating Systems and System Arch and Lab (3/1)
COMP	380/L	Introduction to Software Engineering (2/1)

**4. Upper Division Computer Science Courses (6 Units)**

Select any two upper division (300 or 400 or 500) Computer Science courses (COMP) for which you have the prerequisites or consent of the instructor. Students may select a 300 level course from COMP 310, COMP 380/L or COMP 322/L if it was not used to meet the 300 level elective requirement above.

\*MATH 102, 104, 150A are corequisites for courses in the minor core.

\*\*COMP 310 has prerequisites of PHIL 230 and MATH 326.

Total Units in the Minor

2-23

**Master of Science Degree**

Students in the M.S. program in either Computer Science or Software Engineering complete 30 units of graduate work including 6 units involving a thesis or graduate project.

The core of the graduate program in Computer Science comprises advanced courses in computation theory, algorithms and data structures, system architecture, and software engineering.

The core of the graduate program in Software Engineering comprises advanced courses in software engineering processes, including requirements analysis, software design and implementation, verification and validation, quality assurance, software maintenance and software project management.

The electives in either program may be chosen to form a concentration in an area of specialization or to provide a broadly based program of study, whichever is more consistent with the selected thesis or graduate project

**Requirements for the Master of Science Degree In Computer Science**

Students in the Computer Science M.S. program complete 30 units of graduate work, including a 6-unit thesis or graduate project.

**A. Requirements for Admission**

For admission to the Master of Science Program in Computer Science, applicants must meet the requirements of the University as listed in the catalog, take the Graduate Record Examination (General Test), submit the results to the University and be accepted to the program by the Computer Science Department. Each applicant's transcripts and GRE scores will be reviewed by the Computer Science Department to determine if the student shows high promise of success in the program. Applicants who have completed an ABET accredited Computer Science bachelors of science program and have meet all other entry requirements are exempt from the GRE requirement.

To attain fully classified graduate status in the program, students must complete any required prerequisite undergraduate material, pass the Upper Division Writing Proficiency Exam and have a 3.0 grad point average for all work taken as a Conditionally Classified Student. Information about the prerequisite material can be obtained from the graduate coordinator.

**B. Special Requirements**

Each Computer Science M.S. candidate must submit a proposal for a thesis or graduate project to be done under the supervision of a faculty member. When the thesis or project is approved by that faculty member, the Graduate Coordinator, and the Department Chair, the proposal becomes a contract between the student and the department as to the work to be done for the thesis or graduate project. A three member project/thesis committee is formed with that faculty member as its chair. When the work is done, the student must prepare a report and defend or present the results of the thesis or graduate project before the committee. The report and presentation must be approved by the student's project/thesis committee.

All courses in the student's graduate program must be completed with a grade of C or better. No course taken more than seven years prior to the date of which all requirements for the degree are completed may be counted as part of the 30 units in the degree program. No time limit applies to courses taken to satisfy Computer Science M.S. prerequisite requirements.

**1. Required Courses (15 Units)****a. Breadth Requirement**

Select three of the following four areas of study and complete one course from each of those three areas. The areas of study and the courses available for selection in each area are shown below:

**Algorithms:**

COMP 610 Data Structures and Algorithms (3)

**Systems:**

COMP 620 Computer System Architecture (3)

**Software Engineering:**

COMP 680 Software Engineering (3)

**Foundations:**

COMP 615 Advanced Topics in Computation Theory (3)

COMP 630 Formal Semantics of Programming Languages (3)

**b. Project/Thesis (6 Units)**

COMP 696 Directed Graduate Research (3)

COMP 698 Thesis or Graduate Project (3)

**2. Electives (15 Units)**

Courses at the 400, 500 and/or 600 level in a single area of specialization approved by the Project/Thesis Committee Chair, Graduate Coordinator, and Department Chair. At least six units must be at the 500 level or above.

Total Units Required for the Degree

30

**Requirements for the Master of Science Degree in Software Engineering**

Students in the Software Engineering M.S. program complete 30 units of graduate work, including a 6-unit thesis or graduate group project.

**Requirements for Admission**

For admission to the Master of Science Program in Software Engineering, applicants must meet the requirements of the University as listed in the catalog, take the Graduate Record Examination (General Test), submit the results to the University and be accepted to the program by the Computer Science Department. Each applicant's transcripts and

GRE scores will be reviewed by the Computer Science Department to determine if the student shows high promise of success in the program. Applicants who have completed an ABET accredited Computer Science bachelors of science program and have meet all other entry requirements are exempt from the GRE requirement.

To attain fully classified graduate status in the program, students must complete any required prerequisite undergraduate material or demonstrate equivalent work experience, pass the Upper Division Writing Proficiency Exam, and have a 3.0 grad point average for all work taken as a Conditionally Classified Student. Information about the prerequisite material can be obtained from the graduate coordinator.

### Special Requirements

Each Software Engineering M.S. candidate must submit a proposal for a thesis, or along with a group of other graduate students submit a proposal for a group project, to be done under the supervision of a faculty member. When the thesis or group project is approved by that faculty member, the Graduate Coordinator, and the Department Chair, the proposal becomes a contract between the student(s) and the department as to the work to be done for the thesis or graduate group project. A three member project/thesis committee is formed with that faculty member as its chair. When the work is done, the student(s) must prepare a report and defend or present the results of the thesis or graduate project before the committee. In the case of a group project, each member of the group must present and defend his or her contribution to the final result. The accompanying report must clearly identify the contributions of each member of the group. Each member of the group will be evaluated separately by the committee. The report and presentation, or relevant portion for each member of a group, must be approved by the project/thesis committee.

All courses in the student's graduate program must be completed with a grade of C or better. No course taken more than seven years prior to the date of which all requirements for the degree are completed may be counted as part of the 30 units in the degree program. No time limit applies to courses taken to satisfy Software Engineering M.S. prerequisite requirements.

### 1. Required Courses (12 Units)

#### a. Breadth Requirement

COMP 682	Requirements Analysis and Specification (3)
COMP 684	Software Architecture and Design (3)
COMP 686	Software Engineering Management (3)
COMP 680	Advanced Topics in Software Engineering (3)

#### b. Project/thesis (6 Units)

COMP 696	Directed Graduate Research (3)
COMP 698	Thesis or Graduate Project (3)

### 2. Electives (12 Units)

Select two courses from each of the areas shown below:

#### Software Engineering Electives:

COMP 584	Secure Software Engineering (3)
COMP 585	Graphical User Interfaces (3)
COMP 586	Object-Oriented Software Development (3)
COMP 587	Software Verification and Validation (3)
COMP 588	Software Engineering Metrics (3)

**Other Electives:** Computer Science courses at the 400, 500, or 600 level approved by the Project/Thesis Committee Chair, the Graduate Coordinator, and the Department Chair.

Total Units Required for the Degree

30

## Course List

### COMP 100. Computers: Their Impact and Use (3)

Not open to Computer Science majors. Introduction to the uses, concepts, techniques, and terminology of computing. Places the possibilities and problems of computer use in historical, economic, and social contexts. Shows how computers can assist in a wide range of personal, commercial, and organizational activities. Typical computer applications, including word processing, spread-sheets, and databases. Some sections of this course are taught "on-line" (OL) and/or with a community service learning opportunity (CS or OLC) with activities relating to concepts and theories presented. Check the schedule of classes for the CS, OL, or OLC designation. Prior Internet experience is required for entry into on-line sessions. (Available for General Education, Lifelong Learning) (IC)

### COMP 101. Introduction to Algorithms (2)

Not open to students who have completed COMP 110/L. Introduction to the design, development and expression of algorithms and their stepwise refinement. Expression of algorithms in a formal language. First course in a two-course sequence, the second being a programming language lab. See COMP 105.

### COMP 105. Computer Programming (1)

*Prerequisite:* COMP 101 or 110/L or 106/L. Instruction and practice in a particular computer programming language as listed below.

Three hours of lab per week.

COMP105BAS Computer Programming in BASIC (1)

COMP105C Computer Programming in C (1)

COMP105COB Computer Programming in COBOL (1)

COMP105FOR Computer Programming in FORTRAN (1)

COMP105JAV Computer Programming in JAVA (1)

### COMP 106/L. Computing in Engineering and Science (2/1)

*Prerequisite:* MATH 150A. *Corequisites:* COMP 106L; MATH 150B. Introduction to computing, problem solving and programming intended for science and engineering majors. Programming practice in a high-level structured language. Lab projects involve both micro computers and main frames. Lab: three hours per week.

### COMP 108. Orientation to Computer Science (3)

*Prerequisite:* Passing score on or exemption from the ELM or credit in MATH 093. Not a required course in the major. Recommended for incoming Computer Science majors with limited computing experience as well as those considering a major in Computer Science. Introduction to the Computer Science major and profession. Main focus on developing problem solving, algorithm development, and programming skills, and acquiring critical thinking abilities especially when applied to Computer Science. Additional emphasis on orientation to the University, campus resources, study skills, motivation, and career awareness.

### COMP 110/L. Introduction to Algorithms and Programming (3/1)

*Prerequisite:* Grade of C or better in MATH 102 or 104 or 105 or 150A or MPT I score of at least 27 or passing score on or exemption from the ELM. *Corequisites:* COMP 110L; MATH 104 or 105 or 150A. Introduction to algorithms, their representation, design, structuring, analysis and optimization. Implementation of algorithms as structured programs in a high level language. Lab: three hours per week.

### COMP 122/L. Computer Architecture and Assembly Language (1/1)

*Prerequisites:* Grade of C or better in each: COMP 110/L; Lower Division writing requirement; MATH 104 or 105 or 150A or MPT II score of at least 22 or passing score on or exemption from the ELM. *Corequisites:*

*COMP 122L and MATH 150A.* Introduction to computer architecture, assembly language programming, system software and computer applications. Number systems and data representation. Internal organization of a computer. Primitive instructions and operations. Assembly language. Integrated lecture/lab environment. Lab: three hours per week.

**COMP 182/L. Data Structures and Program Design (3)**

*Prerequisites: Grade of C or better in each: COMP 110/L; Lower Division writing requirement; MATH 104 or 105 or 150A or MPT II score of at least 22 or passing score on or exemption from the ELM. Corequisites: COMP 182L and MATH 150A.* Introduction to data structures and the algorithms that use them. Review of composite data types such as arrays, records, strings, and sets. Role of the abstract data type in program design. Definition, implementation, and application of data structures such as stacks, queues, linked lists, trees, and graphs. Recursion. Use of time complexity expressions in evaluating algorithms. Comparative study of sorting and searching algorithms. Lab: three hours per week.

**COMP 196A-Z. Experimental Topics Courses in Computer Science (1-4)**

**COMP 222. Computer Organization (3)**

*Prerequisites: Grade of C or better in each: COMP 122/L, 182/L.* Extension of basic addressing concepts to more advanced addressability such as base register and self-relative addressing. Comparative computer architecture focusing on such organizations as multiple register processors and stack machines. Basics of virtual memory input-output. Introduction to the concept of microprogrammable systems. Low level language translation process associated with assemblers. System functions such as relocatable loading and memory management. Application of data structure and hashing techniques to the above. Other related topics.

**COMP 232. Concepts of Programming Languages (3)**

*Prerequisites: COMP 122/L, 182/L.* Discussion of issues in the design, implementation, and use of high-level programming languages. Historical background. How languages reflect different design philosophies and user requirements. Technical issues in the design of major imperative (procedural) programming languages. Other approaches to programming: functional programming, logic programming, and object-oriented programming.

**COMP 270/L. Business Programming (2/1)**

*Prerequisites: Grade of C or better in each: COMP 182/L, MATH 150A. Corequisite: COMP 270L.* Introduction to file-based data structures, database concepts and the manipulation of database content from user written software. Theoretical and practical concepts are covered. Lab: three hours per week.

**COMP 282. Advanced Data Structures (3)**

*Prerequisites: Grade of C or better in each: COMP 182/L, MATH 150A.* Introduction to advanced data structures (particularly persistent structures) using object-oriented design. Main memory structures: hash tables, trees. Architectural foundations for files. Large-scale sorting. Hash-based persistent structures. Indexed files. Introduction to databases.

**COMP 296A-Z. Experimental Topics Courses in Computer Science (1-4)**

Upper Division

**COMP 300. Computer Fluency (3)**

*Prerequisite: Completion of the lower division writing requirement.* Not open to students who have credit in COMP 100; does not provide credit towards a computer science major. Study of fundamental computing concepts related to: information technology, data and its digital representation, technological power, computing limitations and social impact. Survey of essential and advanced applications designed to process different forms of information other than text. Promote

computing skills such as basic algorithmic thinking, debugging, logical reasoning and critical use of information. Develop capabilities for applying the technology. Some sections of this course are taught "on-line" (OL) and/or with a community service learning opportunity (CS or OLC) with activities related to concepts and theories presented. Prior Internet experience is required for entry into on-line sessions. Check the schedule for classes for the CS, OL, or OLC designation. (Available for General Education, Lifelong Learning) (IC)

**COMP 310. Automata, Languages and Computation (3)**

*Prerequisites: COMP 232; MATH 326; PHIL 230.* Study of the relation of languages (i.e. sets of strings) and machines for processing these languages, with emphasis on classes of languages and corresponding classes of machines. Phrase structure languages and grammar. Types of grammar and classes of languages. Regular languages and finite state automata. Context-free languages and pushdown automata. Unrestricted languages and Turing Machines. Computability models of Turing, Church, Markov, and McCarthy. Applications to programming languages, compiler design, and program design and testing.

**COMP 322/L. Introduction to Operating Systems and System Architecture (3/1)**

*Prerequisites: COMP 222. Corequisite: COMP 322L; Recommended Prerequisites: COMP 105C or knowledge of "C" Language.* Examination of the principal types of systems including batch, multi-programming, and time-sharing. Discusses networked system. Considers the salient problems associated with implementing systems including interrupt of event driven systems, multi-tasking, storage and data base management, and input-output. Emphasizes some of the simple algorithms used to solve common problems encountered such as deadlocks, queue service, and multiple access to data. Projects are implemented to reinforce the lectures. One three-hour lab per week.

**COMP 380/L. Introduction to Software Engineering (2/1)**

*Prerequisites: COMP 282 or 270/L. Corequisite: COMP 380L.* Concepts and techniques for systems engineering, requirements analysis, design, implementation and testing of large scale computer systems. Principles of software engineering for production of reliable, maintainable and portable software products. Emphasis on object-oriented analysis and design techniques. Topics include unit, integration and systems testing, configuration management, software quality assurance practices, and an introduction to Computer Aided Software Engineering (CASE). This is a lecture portion of a course in software engineering involving the design and partial implementation of a software system as a group project. Lab: three hours per week.

**COMP 396A-Z. Experimental Topics Courses in Computer Science (1-4)**

**COMP 410. Logic Programming (3)**

*Prerequisites: COMP 232; 282; 310; attempted UDWPE.* Programming techniques in the logic programming language PROLOG. Prenex conjunctive normal form and grammatical algorithms. Tableaux, sequenzen, resolution, and other semi-decision procedures. Closures of relations, fixed point theory, control mechanisms, relationship to functional programming.

**COMP 420. Advanced Operating System Concepts (3)**

*Prerequisites: COMP 322/L; attempted UDWPE.* In-depth discussion of selected issues related to the study of operating systems. Areas of coverage may include concurrency issues, resource allocation, storage management, and multiprocessor environments. Discusses underlying theory and algorithms related to the issues.

**COMP 421. The Unix Environment for Programmers (3)**

*Prerequisites: COMP 322/L; attempted UDWPE.* Usage of UNIX in a software development environment; rapid prototyping of large projects.

Study available utilities, programming styles, efficiency issues, and operating system interfaces. Gain an advanced understanding of UNIX and its use in improving programmer productivity.

**COMP 424. Computer System Security (3)**

*Prerequisites:* COMP 322/L; 380/L; attempted UDWPE. Analysis of the need for computer system security, and the security techniques in operating systems, data bases, and computer networks. Supporting techniques such as auditing, risk analysis, and cost-benefit tradeoffs are discussed.

**COMP 426. Fault-Tolerant Software and Computing (3)**

*Prerequisites:* COMP 322/L; 380/L; attempted UDWPE. Examination of dependability requirements in computing, and the basic principles of system-level reliability and fault-tolerance. Software-based implementation of fault-tolerance in distributed systems. Fault-tolerant software and data bases: reliability modeling, fault-tolerance techniques (e.g., recovery blocks, N-version programming), and design approaches.

**COMP 429. Computer Network Software (3)**

*Prerequisites:* COMP 322/L; attempted UDWPE. Basic software design and analysis considerations in networking computers into coherent, cooperating systems capable of processing computational tasks in a distributed manner. Network topology, routing procedures, message multiplexing and process scheduling techniques.

**COMP 430. Language Design and Compilers (3)**

*Prerequisites:* COMP 310; 380/L; attempted UDWPE. Examination of the issues involved in the design and subsequent implementation of programming languages. Considerations of the implementation difficulties of including various features in a programming language. Tools and techniques to facilitate both the processing of programming languages and the building of programming processors.

**COMP 432. Object-Oriented Programming (3)**

*Prerequisite:* COMP 322/L; attempted UDWPE. Principles of object-oriented design and programming; object-oriented languages such as Simula, C++ and Smalltalk are compared to provide an understanding of the role of objects, methods, message passing, encapsulation, classes, inheritance and instance variables in a productive programming environment. Discusses language design and programming issues.

**COMP 440. Database Design (3)**

*Prerequisite:* COMP 380/L; attempted UDWPE. Database structure including: structure definition, data models, semantics of relations, operation on data models. Database schemas: element definition, use and manipulation of the schema. Elements of implementation. Algebra of relations on a database. Hierarchical data bases. Discussion of information retrieval, reliability, protection and integrity of databases.

**COMP 450. Societal Issues in Computing (3)**

*Prerequisites:* COMP 380/L; upper division writing exam requirement. Survey course on the role of the digital computer in modern society. The dangers of the misuse of computers (as in the invasion of privacy), as well as the proper and intelligent use of the machines, are discussed. Not available for graduate credit.

**COMP 465/L. Computer Graphic Systems and Design and Lab (2/1)**

*Prerequisites:* MATH 262; attempted UDWPE. Corequisite: COMP 465L. Fundamental concepts of computer graphics. Graphics devices; graphics languages; interactive systems. Applications to art, science, engineering and business. Trade-offs between hardware devices and software support. Lab: three hours per week.

**COMP 467. Multimedia Systems Design (3)**

*Prerequisite:* COMP 380/L; attempted UDWPE. Study of fundamentals of multimedia storage, processing, communication, presentation, and display by digital means with emphasis on audio, still images and

video media. Includes sampling theory, compression techniques and synchronization. Discussion of hypermedia and methodology issues. Multimedia programming; software tools for authoring multimedia applications and interfaces.

**COMP 469. Introduction to Artificial Intelligence (3)**

*Prerequisites:* COMP 310; 380/L; 410; attempted UDWPE. Exploration of the use of computers to perform computations normally associated with intelligence. These include game playing, theorem proving, problem solving, question answering and visual perception. Topics include languages, system architectures and heuristic strategies for advanced, high level computations. Covers computational models for knowledge representation, natural language and vision.

**COMP 480/L. Software System Development (2/1)**

*Prerequisite:* COMP 380/L. Corequisite: COMP 480L. Project-oriented course to allow the students to apply their knowledge to the design of a large system. Students identify a suitable computer problem, examine various methods of attacking it, choose a suitable one, and realize a solution in an appropriate computer language. Lab: three hours per week.

**COMP 484/L. E-business Technologies and Lab (2/1)**

*Prerequisites:* COMP 322/L, COMP 380/L. Corequisite: COMP 484L. Internet infrastructure and the underlying networking technologies. Study of system and software architectures for e-business/e-commerce systems. Principles of website design. Advances in web-engineering technologies. Principles of web-based based transaction processing. XML and the associated technologies. Web service technology. Security and privacy issues. Study of the emerging internet technologies. Lab: three hours per week.

**COMP 485. Human-Computer Interaction (3)**

*Prerequisite:* COMP 380/L; attempted UDWPE. Examines the information exchange between humans and computer systems. Discusses aspects of input/output devices, software engineering, and human factors with respect to human-computer interactions. Topics include: text and graphic display; user modeling; program design, debugging, complexity and comprehension; and current research studies and methodologies.

**COMP 494A/B/C. Academic Internship (1-2-3)**

*Prerequisites:* Junior standing or above in major, UDWPE requirement, prior approval of the Department, and in good standing as a matriculated student. Academic internship training program. Supervised off-campus professional computing experience for selected computer science students. Academic internship units do not count towards General Education units or major requirements. Maximum of six units of enrollment is allowed. Grading is Credit/No Credit only. Only one enrollment per semester permitted.

**COMP 496A-Z. Experimental Topics Courses in Computer Science (1-4)**

**COMP 499. Independent Study (1-3)**

## Graduate

**COMP 518/L. Algorithms and Data Structures (3/1)**

*Corequisite:* COMP 518L. Intensive course open only to graduate students and cannot be used to satisfy the requirement of 30 units of approved graduate work. Programming and data structures covering all of the topics of COMP 110/L and 182/L. Lab involves programming design of significant projects in a high level programming language. Lab: three hours per week.

**COMP 529. Advanced Network Topics (3)**

*Prerequisite:* COMP 429; MATH 340 or 341. Advanced course on design and analysis of high-speed networks (Broadband ISDN and Asynchronous Transfer Mode [ATM] networks) and their protocols. Topics include: multimedia services integrating techniques including synchronous and asynchronous transfer modes. ATM standards. ATM switch architecture, ATM network traffic control, ATM experimental networks, high-speed LAN/MANs, internetworking with high-speed networks, and simulation techniques.

**COMP 560. Expert Systems (3)**

*Prerequisite:* COMP 469. Extensive introduction to the concepts and techniques of expert systems. Rationale for such systems including evaluation of prospective domains. Explores existing systems, those under development and likely future areas. Basic architecture is demonstrated using both example and rule-based systems. Commercial tools for building expert systems are surveyed and evaluated. Of special concern, knowledge acquisition methods. Guidelines given for planning and managing development projects.

**COMP 565. Advanced Computer Graphics (2/1)**

*Prerequisite:* COMP 322/L; COMP 465/L or equivalent. This course will cover the theory, design, implementation, and application of advanced computer graphics environments. Accelerated 3D graphics APIs; the modeling and simulation of light, sound, physical objects, motion, and collisions; and user interaction in single and multi-user virtual environments will be studied. The application domain for this class is interactive 3D computer games, scientific visualization, and Virtual Reality.

**COMP 585. Graphical User Interfaces (3)**

*Prerequisites:* COMP 322/L; 380/L. The design, development and analysis of programs requiring graphical, direct manipulation, user interfaces (GUIs) will be examined. The majority of modern software includes a GUI. The development tools, environments and style guides for common GUIs will be used in course assignments and discussed in lecture. The course involves the design and development of several GUI programs. The aesthetic and human computer interaction aspects and future trends in GUIs design and development will also be reviewed.

**COMP 586. Object-Oriented Software Development (3)**

*Prerequisites:* COMP 322/L; 380/L. Review of object oriented concepts. Comparison with functional methods. Benefits and pitfalls of object orientation. Fundamentals of object-oriented modeling-associations, links, states. Survey of object-oriented development methods. In-depth study of a current object-oriented method. Object-oriented software requirements analysis and modeling. Object-oriented preliminary design. Designing concurrent and multiprocessor systems. Object-oriented detailed design. Object-oriented and object-based implementations. Object-oriented testing.

**COMP 587 Software Verification and Validation (3)**

*Prerequisite:* COMP 380/L and passing score on the UDWPE. An in depth study of verification and validation strategies and techniques as they apply to the development of quality software. Topics include test planning and management, testing tools, technical reviews, formal methods and the economics of software testing. The relationship of testing to other quality assurance activities as well as the integration of verification and validation into the overall software development process are also discussed.

**COMP 589 Software Metrics (3)**

*Prerequisite:* COMP 380/L and passing score on the UDWPE. The role of metrics and quantitative models in software development. Product metrics, process metrics, measurement models and techniques for empirical validation. Measurement and analysis: implementation of a

metrics program. Measuring software size, complexity, and functionality at different stages of software development. Use of measures to predict effort and schedule required for software projects. Measures of software quality. Analyzing defect data to predict software reliability. Performance measures. Management applications for metrics. Tools that support metrics collection, analysis, summary, and presentation.

**COMP 595A-Z. Experimental Topics Courses (3)****COMP 598A-Z. Advanced Selected Topics (1-4)**

*Prerequisite:* Instructor consent.

**COMP 610. Data Structures and Algorithms (3)**

*Prerequisites:* COMP 310; MATH 482. Topics include: design strategies for data structures and algorithms; theoretical limits to space and time requirements time/space trade offs; open problems in the field.

**COMP 615. Advanced Topics in Computation Theory (3)**

*Prerequisites:* COMP 310; MATH 482. Languages and the theory of computation are studied in depth. Covers advanced material concerning regular and context free languages. Study of deterministic context free languages, context sensitive languages, recursive and recursively enumerable sets. Investigation of current areas of interest.

**COMP 620. Computer System Architecture (3)**

*Prerequisites:* COMP 322/L; 380/L. Analysis and evaluation of individual computers, networks of computers and the programs which support their operation and use. Emphasis on comparison of architectures and the risks and benefits associated with various approaches and configurations.

**COMP 630. Formal Semantics of Programming Languages (3)**

*Prerequisites:* COMP 310; 380/L. Rigorous verification and formal proofs of correctness. Denotational semantics, models of axiomatic systems, fixpoint theory of computation. Soundness and completeness of programming logics. Abstract data types and other issues in the formal definition of programming languages.

**COMP 667. CAD/CAM Systems Design (3)**

*Prerequisite:* COMP 465/L. Discuss and evaluate data structures and algorithms necessary to design and implement computer systems in manufacturing environments. Existing and anticipated technology. Students will design, implement, test, and evaluate CAD/CAM systems by building upon standard computer graphics packages.

**COMP 680. Software Engineering (3)**

*Prerequisites:* COMP 322/L; 380/L. Examination of the critical theoretical problems underlying the specification, design, development and evaluation of large software systems, and the extent to which existing techniques and methodologies cope with these problems.

**COMP 682. Software Requirements Analysis and Specification (3)**

*Prerequisite:* COMP 380/L. An in-depth study of the early phases of the software development life cycle commonly called software requirements analysis and specification. Topics include the gathering of both functional and nonfunctional requirements, customer communication, requirements prototyping, requirements modeling, requirements validation, the documentation of requirements in terms of a formal software requirements specification, and the management of software requirements.

**COMP 684. Software Architecture and Design (3)**

*Prerequisites:* COMP 380/L; 682. *Prerequisite:* COMP 586. Techniques, methods, and tools for designing, building, analyzing, and evaluating the structural, architectural, and behavioral properties of software systems. It includes the study of the fundamental concepts and principles of software architectural design, structured design, object-oriented design, component-level design, and design for reuse.

**COMP 686. Software Engineering Management (3)**

*Prerequisite: COMP 380/L; upper division writing requirement.* Provides a framework for understanding software engineering management models, technologies, trends, tools, and planning processes. Emphasizes the development of an individualized approach to managing software teams, projects, and systems. The role of management as an increasingly critical factor in software engineering is examined.

**COMP 695A-Z. Experimental Topics Courses (3)****COMP 696A-C. Directed Graduate Research (1-3)**

*Prerequisite: Permission of chair of project/thesis committee. (Credit/No Credit only)*

**COMP 698A-C. Thesis or Graduate Project (1-3)****COMP 699. Independent Study (1-3)**