SOFTWARE ENGINEERING (S E)

Any experimental courses offered by S E can be found at:
registrar.iastate.edu/faculty-staff/courses/explistings/ **(**http://www.registrar.iastate.edu/faculty-staff/courses/explistings/**)**

Courses primarily for undergraduates:

S E 101: Software Engineering Orientation
Cr. R.
Introduction to the procedures, policies, and resources of Iowa State University and the Software Engineering Program. Offered on a satisfactory-fail basis only.

S E 166: Careers in Software Engineering
Cr. R.
Overview of the nature and scope of the software engineering profession, relationship of coursework to careers, and program of study planning. Offered on a satisfactory-fail basis only.

S E 185: Problem Solving in Software Engineering
(2-2) Cr. 3.
Prereq: Credit or concurrent enrollment in MATH 143 (or satisfactory scores on mathematics placement examinations)
Introduction to software engineering and computer programming. Systematic thinking process for problem solving in the context of software engineering. Group problem solving. Solving software engineering problems and presenting solutions through computer programs, written documents and oral presentations. Introduction to principles of programming, software design, and extensive practice in design, writing, running, debugging, and reasoning about programs. Satisfactory placement scores can be found at: https://math.iastate.edu/academics/undergraduate/aleks/placement/. Only one of ENGR 160, A B E 160, AER E 160, C E 160, CH E 160, CPR E 185, E E 185, I E 148, M E 160, and S E 185 may count toward graduation.

S E 186: Problem Solving in Software Engineering II
(0-2) Cr. 1. S.
Prereq: S E 185
Group projects in software engineering. Work effectively in teams to solve problems and provide technical reports and presentations. Self-directed team based projects that are representative of problems faced by software engineers.

S E 309: Software Development Practices
(Cross-listed with COM S). (3-1) Cr. 3. F.S.
Prereq: Minimum of C- in (COM S 228; MATH 165)
Practical introduction to methods for managing software development. Software engineering concepts, practices and tools. Requirements analysis, structured and object-oriented design, coding, testing, and maintenance. Software process models, software tools and environments. Programming projects that provide exposure to information management techniques, client/server model, networking and communication.

S E 317: Introduction to Software Testing
Cr. 3.
Prereq: (COM S 230 or CPR E 310); (COM S 309 or S E 309)
Basic principles and techniques for software testing. Test requirements and management. Test design techniques, evaluation metrics, model-based testing, unit testing, system and integration testing. Software testing tools and programming projects.

S E 319: Construction of User Interfaces
(Cross-listed with COM S). (3-0) Cr. 3. F.S.
Prereq: COM S 228
Overview of user interface design. Evaluation and testing of user interfaces. Review of principles of object orientation, object oriented design and analysis using UML in the context of user interface design. Design of windows, menus and commands. Developing Web and Windows-based user-interfaces. Event-driven programming. Introduction to Frameworks and APIs for the construction of user interfaces.

S E 329: Software Project Management
(Cross-listed with CPR E). (3-0) Cr. 3.
Prereq: COM S 309
Process-based software development. Capability Maturity Model (CMM). Project planning, cost estimation, and scheduling. Project management tools. Factors influencing productivity and success. Productivity metrics. Analysis of options and risks. Version control and configuration management. Inspections and reviews. Managing the testing process. Software quality metrics. Modern software engineering techniques and practices.

S E 339: Software Architecture and Design
(Cross-listed with CPR E). (3-0) Cr. 3.
Prereq: S E 319
Modeling and design of software at the architectural level. Architectural styles. Basics of model-driven architecture. Object-oriented design and analysis. Iterative development and unified process. Design patterns. Design by contract. Component based design. Product families. Measurement theory and appropriate use of metrics in design. Designing for qualities such as performance, safety, security, reliability, reusability, etc. Analysis and evaluation of software architectures. Introduction to architecture definition languages. Basics of software evolution, reengineering, and reverse engineering. Case studies. Introduction to distributed system software.

S E 342: Principles of Programming Languages
(Cross-listed with COM S). (3-1) Cr. 3. F.S.
Prereq: Minimum of C- in COM S 228 and MATH 165; COM S 230 or CPR E 310
Study of concepts in programming languages, especially functional programming concepts. Overview of major programming paradigms, their relationship, and tradeoffs among paradigms enabling sound choices of programming language for application-specific development. Programming projects.

S E 362: Object-Oriented Analysis and Design
(Cross-listed with COM S). (3-0) Cr. 3. F.S.
Prereq: Minimum of C- in (COM S 228; MATH 165); ENGL 250
Object-oriented requirements analysis and systems design. Analysis and design methodologies including use case and Unified Modeling Language (UML). Design principles, heuristics, and patterns. Architectural patterns and alternative programming paradigms. Group design and programming project.

S E 409: Software Requirements Engineering
(Dual-listed with COM S 509). (3-0) Cr. 3.
Prereq: COM S 309; for graduate credit: graduate standing or permission of instructor
The requirements engineering process including elicitation, requirements analysis fundamentals, requirements specification and communication, and requirements evaluation. Modeling of functional and nonfunctional requirements, traceability, and requirements change management. Case studies and software projects.

S E 412: Formal Methods in Software Engineering
(Dual-listed with COM S 512). (Cross-listed with COM S, CPR E). (3-0) Cr. 3.
Prereq: COM S 311; STAT 305 or STAT 330 or STAT 341; for graduate credit: graduate standing or permission of instructor
A study of formal techniques for model-based specification and verification of software systems. Topics include logics, formalisms, graph theory, numerical computations, algorithms and tools for automatic analysis of systems. Graduate credit requires in-depth study of concepts.

S E 413: Program Analysis
(Dual-listed with COM S 513). (Cross-listed with COM S). (3-0) Cr. 3.
Prereq: (COM S 327 or CPR E 288); COM S 342
Algorithms, AI techniques and tools for automatically reasoning about code and program executions. Theory and foundations related to control flow analysis, dataflow analysis, abstract interpretation, and symbolic execution. Applications of program analysis to bug detection, test input generation, debugging, program repair, specification inference and trustworthy AI engineering. Concepts, algorithms, tools, benchmarks, methodologies for solving problems using program analysis and for preparing research in program analysis.

S E 416: Software Evolution and Maintenance
(Cross-listed with CPR E). (3-0) Cr. 3.
Prereq: COM S 309
Practical importance of software evolution and maintenance, systematic defect analysis and debugging techniques, tracing and understanding large software, impact analysis, program migration and transformation, refactoring, tools for software evolution and maintenance, experimental studies and quantitative measurements of software evolution. Written reports and oral presentation.

S E 417: Software Testing
(Cross-listed with COM S). (3-0) Cr. 3.
Prereq: COM S 309; (COM S 230 or CPR E 310); ENGL 250
An introduction to software testing principles and techniques. Test models, test design, test adequacy criteria; regression, integration, and system testing; and software testing tools.

S E 419: Software Tools for Large Scale Data Analysis
(Cross-listed with CPR E). (3-3) Cr. 4.
Prereq: COM S 363 or COM S 352 or CPR E 308; COM S 228
Software tools for managing and manipulating large volumes of data, external memory processing, large scale parallelism, and stream processing, data interchange formats. Weekly programming labs that involve the use of a parallel computing cluster.

S E 421: Software Analysis and Verification for Safety and Security
(Cross-listed with CPR E). Cr. 3. F.S.
Prereq: (COM S 230 or CPR E 310); COM S 309
Significance of software safety and security; various facets of security in cyber-physical and computer systems; threat modeling for software safety and security; and categorization of software vulnerabilities. Software analysis and verification: mathematical foundations, data structures and algorithms, program comprehension, analysis, and verification tools; automated vs. human-on-the-loop approach to analysis and verification; and practical considerations of efficiency, accuracy, robustness, and scalability of analysis and verification. Cases studies with application and systems software; evolving landscape of software security threats and mitigation techniques. Understanding large software, implementing software analysis and verification algorithms.

S E 422: Cloud Computing - Software Development
(3-0) Cr. 3. F.
Prereq: (S E 309 or S E 339); (CPR E 381 or COM S 321)
A comprehensive view of cloud computing with respect to software development from platforms and services to programming and infrastructure. Virtualization and containerization; cloud computing platforms, with examples from currently available cloud services; cloud services for data analytics, machine learning, and devops; programming frameworks for parallel computing in the cloud; distributed storage in the cloud; Container management. Includes homeworks and programming assignments. The programming assignments will be done in AWS.

S E 440: Principles and Practice of Compiling
(Dual-listed with COM S 540). (Cross-listed with COM S). (3-1) Cr. 3.
Prereq: COM S 331 or COM S 342; COM S 309; ENGL 250; for graduate credit: graduate standing or permission of instructor
Theory of compiling and implementation issues of programming languages. Programming projects leading to the construction of a compiler. Projects with different difficulty levels will be given for 440 and 540. Topics include: lexical, syntactic and semantic analyses, syntax-directed translation, code generation, runtime environment and library support.

S E 490: Independent Study
Cr. arr. Repeatable.
Prereq: Senior classification in software engineering; Permission of Instructor
Investigation of an approved topic.

S E 491: Senior Design Project I and Professionalism
(Cross-listed with CPR E, E E). (2-3) Cr. 3. F.S.
Prereq: CPR E major: CPR E 232, cr or enrollment in CPR E 308, completion of 29 cr in the CPRE prof prog, ENGL 314. E E major: E E 232, cr or enrollment in E E 322, completion of 24 cr in the EE prof prog, ENGL 314. S E major: S E 317 and S E 339, ENGL 309 or ENGL 314. Co-req: CPR E 308 or COM S 352
Preparing for entry to the workplace. Selected professional topics. Use of technical writing skills in developing project plan and design report; design review presentation. First of two-semester team-oriented, project design and implementation experience.

S E 492: Senior Design Project II
(Cross-listed with CPR E, E E). (1-3) Cr. 2. F.S.
Prereq: CPR E 491 or E E 491 or S E 491
Second semester of a team design project experience. Emphasis on the successful implementation and demonstration of the design completed in E E 491, Cpr E 491, or S E 491 and the evaluation of project results. Technical writing of final project report; oral presentation of project achievements; project poster.