COMPUTER SCIENCE (CMPSC)

CMPSC 100: Computer Fundamentals and Applications

3 Credits

Introduction to computer fundamentals and applications to data processing environments.

Prerequisite: 2 entrance units in mathematics

CMPSC 101: Introduction to Programming

3 Credits

This course introduces the fundamental concepts and processes of solving computational problems through the design, implementation, testing, and evaluation of basic computer programs. The concepts include basic computational constructs such as calculation, iteration, conditions, functions, and data types. These provide the basic building blocks found in virtually all programming languages. The processes include the step-by-step refinement of a problem description into individual components that can be implemented, tested, and integrated into an effective solution. As a general education course, the central theme to the course is computational thinking which includes a wide range of approaches to solving problems and designing systems that draw upon concepts fundamental to computer science. Computational thinking includes thinking about various types and sources of data, and the correctness, efficiency, elegance, and simplicity of various potential solutions. Computational thinking is applying principles of abstraction at multiple levels to focus on important details; it is applying problem decomposition to identify small problems that can be individually solved then combined to form a solution to the original problem. Upon completion of this course, the student will be able to conceptualize and implement computational solutions to problems; to utilize the imperative model of computation to solve problems; to reason about problems at multiple levels of abstraction; and to analyze code for its behavior, efficiency, and correctness. A student may receive credit for only one of the following courses: CMPSC 101, CMPSC 121, CMPSC 131, CMPSC 200, CMPSC 201

Prerequisite: 2 entrance units in mathematics
Bachelor of Arts: Quantification
General Education: Quantification (GQ)
GenEd Learning Objective: Crit and Analytical Think
GenEd Learning Objective: Key Literacies

CMPSC 102: Introduction to Visual Programming

3 Credits

Problem solving for non-majors; high-level language programming; controlstructures, functions, parameters, recursion, arrays, records/ structures; verification; debugging; documentation.

Prerequisite: 2 entrance units in mathematics

CMPSC 109: Introduction to Data Processing with COBOL

3 Credits

Study of the COBOL programming language and its applications in industry. CMPSC 109 Introduction to Data Processing with COBOL (3) CMPSC 109 is an introductory COBOL course, designed particularly for students in the Management Information Systems major at Behrend, but also appropriate for students in other Business majors and students from any major with an interest in programming in a data processing environment. The course assumes no prior study in programming.

Prerequisite: 3 credits of programming

CMPSC 111: Logic for Computer Science

1 Credits

An introduction to logic and its application to problem solving and computer science. CMPSC 111S Logic for Computer Science (1)Computer Science provides the fundamental tools for analyzing problems and designing solutions to these problems which can be implemented on a computer. Logic plays an important role in this process, from a general-purpose tool for reasoning about knowledge to a special-purpose language for specifying the behavior of programs and designing hardware. This course examines the role of logic in problem solving and its application to computer science and computer engineering. Example problems will be drawn from a variety of sources, including brain teasers, puzzles, and mathematics. We will show how these problems and their solutions apply to real problems involving computers. We will also explore a number of the important areas of computer science and computer engineering including Boolean and Digital Logic, Designing Arithmetic Hardware, Cryptography and Security Programming Languages, Networking and Wireless Communication, Artificial Intelligence, and Computer Ethics.

First-Year Seminar

CMPSC 121: Introduction to Programming Techniques

3 Credits

Design and implementation of algorithms. Structured programming. Problem solving techniques. Introduction to a high-level language, including arrays, procedures, and recursion.

Prerequisite: MATH 110 or prerequisite or concurrent MATH 140
Bachelor of Arts: Quantification
General Education: Quantification (GQ)

CMPSC 122: Intermediate Programming

3 Credits

Object-oriented programming, recursion, fundamental data structures (including stacks, queues, linked lists, hash tables, trees, and graphs), the basics of algorithmic analysis, and an introduction to the principles of language translation.

Prerequisite: CMPSC121

CMPSC 122H: Intermediate Programming

3 Credits

Object-oriented programming, recursion, fundamental data structures (including stacks, queues, linked lists, hash tables, trees, and graphs), the basics of algorithmic analysis, and an introduction to the principles of language translation.

Honors

CMPSC 131: Programming and Computation I: Fundamentals

3 Credits

This course introduces the fundamental concepts and processes of solving computational problems through the design, implementation, testing and evaluation of efficient and robust computer programs. The concepts include basic computational constructs found in imperative, object-oriented and functional programming languages such as iteration, conditionals, functions, recursion, and datatypes. These provide the basic building blocks found in virtually all programming languages. The processes include the stepwise refinement of a problem description into individual components that can be implemented, tested, and integrated into an effective solution. A central theme to the course is computational thinking which includes a wide range of approaches to solving problems and designing systems that draw upon concepts fundamental to computer science. Computational thinking includes thinking recursively, considering parallel processing, thinking about types and type checking, judging a program not just for correctness and efficiency but also for its esthetics, and judging a system design for its simplicity and elegance. Computational thinking is applying principles of abstraction at multiple levels to focus on important details; it is applying problem decomposition to identify small problems that can be individually solved then combined to form a solution to the original problem. Computational thinking uses program invariants to describe a system's behavior succinctly and declaratively. Computational thinking considers multiple models of computation when designing an effective solution to a problem.

CONCURRENT COURSES: MATH 110; MATH 140

CMPSC 132: Programming and Computation II: Data Structures

3 Credits

This course builds upon the foundations of programming and computation by introducing and studying the data structures and programming language features that support the design and construction of large-scale software systems. It introduces the foundations of object-oriented programming, the design and analysis of efficient algorithms using important data structures, and programming techniques that support reusable and modular program components, including data abstraction, polymorphism, and higher-order functions. Topics from object-oriented programming include classes, objects, inheritance, methods, message passing, static and dynamic type checking. These topics form the core of most object-oriented languages and provide a foundation for learning more advanced language topics. Data structures capture the common organization of many kinds of data arising in the design of efficient solutions to computational problems. Specific data structures covered include stacks, queues, trees, graphs and linked lists. The design and analysis of efficient algorithms using these data structures provide a foundation for the study of computing, where understanding the complexity of a problem and the availability of efficient solutions are essential skills. Finally, topics including higher-order functional programming, data abstraction and parametric polymorphism, as well as principles from object-oriented programming, come together to support the design and implementation of modular, reusable and robust code.

Prerequisite: CMPSC 121; CMPSC 131

CMPSC 199: Foreign Studies

1-12 Credits/Maximum of 12

Courses offered in foreign countries by individual or group instruction.

International Cultures (IL)

CMPSC 200: Programming for Engineers with MATLAB

3 Credits

Development and implementation of algorithms in a procedure-oriented language, with emphasis on numerical methods for engineering problems. A student may receive credit for only one of the following courses: CMPSC 101, 102, 200, 201, or 202. CMPSC 200 CMPSC 200 Programming for Engineers with MATLAB (3) CMPSC 200 is a service course offered to engineering and science majors. The course teaches basic programming concepts including: algorithm development, data types, number representation, control structures, functions, plotting and basic numerical analysis techniques. The course enables students to develop computer programs in MATLAB to solve simple engineering problems. The basic numerical analysis techniques covered in the course include matrix operations, systems of equations, solving equations, roots, curve fitting, interpolation, numerical integration and ordinary differential equations.Students analyze physics-based and engineering problems; develop algorithms to solve the problems; implement the algorithms in the MATLAB programming environment; and produce informative output in both numerical and graphical form. The general programming concepts learned in the course are commonly found in most programming language environments. The problem-solving skills learned in the course can be utilized in upper-level engineering and science courses.The lecture portion of the course gives students the conceptual and syntactical background needed for the successful completion of practical programming assignments during the laboratory portion of the course. The laboratory instruction involves hands-on programming by individual students or student teams assisted by a teaching assistant and/or instructor.Evaluation methods may include examinations, in-class labs, and programming projects. The course is generally held in a STEC room where each student has access to a computer. The course will be offered during the Spring semester.

Prerequisite: MATH 140; Concurrent: MATH 141
General Education: Quantification (GQ)

CMPSC 201: Programming for Engineers with C++

3 Credits

Development and implementation of algorithms in a procedure-oriented language, with emphasis on numerical methods for engineering problems. A student may receive credit for only one of the following courses: CMPSC 101, CMPSC 102, CMPSC 200, CMPSC 201, or CMPSC 202.

Prerequisite: MATH 140; Concurrent: MATH 141
Bachelor of Arts: Quantification
General Education: Quantification (GQ)

CMPSC 203: Introduction to Spreadsheets and Databases

4 Credits

Design, use, and programming of spreadsheets and data bases with applications from a range of disciplines.

Prerequisite: 2 entrance units in mathematics
Bachelor of Arts: Quantification
General Education: Quantification (GQ)

CMPSC 208: Technical Game Development

3 Credits

Introduction to the tools and techniques required to implement games in a virtual environment. GAME 250 / CMPSC 208 Technical Game Development. First, students learn about game and player elements by creating characters and objects and the means of user interactivity. Both orthographic and perspective views are introduced to assist in character design. Objects and characters are created using fundamental geometric primitives like scale, rotation, translation and extrusion. The set operations, union, intersection, and subtraction, are applied to create compound objects. Bezier and NURB curves are introduced to create objects with irregular contours. Students also learn to design graphical user interfaces (GUIs) and handle mouse and keyboard events to support user interactions. Second, students are introduced to methods of storytelling and guide them to build narratives for games. Methods of proximity and collision detection in the environment are studied for both static and dynamic objects. Dynamic objects are programmed to move and behave in a deterministically, random, or probabilistically under a variety of lighting methods including ambient, directional, point and diffuse lights are introduced. A number of particle systems are developed with different considerations of randomness, vector direction and velocity. The concept of linear interpolation is illustrated and applied to texture mapping to improve the look and feel of objects. Third, students are introduced to functions, propositional logic, loops, and randomness to model game behavior. Students will learn to combine a series of primitive actions into a function for control and reuse. Propositional logic will guide students to define conditions and develop game rules. Loops are introduced to simplify the implementation of repeated game behavior. Randomness enables the simulation of many life-like object movements. Students will learn and practice how to write concurrent, event drive and sequential processing algorithms for game objects. Fourth, students are introduced to the game development process of pre-production, production and post-condition phases and have them apply it to their own project. The topic of maintenance will be introduced with an emphasis on refactoring techniques, critical to improving the quality of game and providing flexibility for future updates. This course has a significant applied element. Game engine tools are used to develop prototypes of games and playtest them. Lab assignments are given throughout the semester and a final project requires students to demonstrate mastery of all aspects of the course.

Prerequisites: MATH 21
Cross-listed with: GAME 250
Bachelor of Arts: Quantification
General Education: Quantification (GQ)
GenEd Learning Objective: Key Literacies

CMPSC 221: Object Oriented Programming with Web-Based Applications

3 Credits

The course covers advanced object-oriented principles and their application to web-based, net-centric computing. Major topics include virtual machines, intermediate code generation (Java-specific), graphical user interfaces (GUI) design, event handling, server-side programming with database queries, and security, permissions and file management concepts for client/server systems. Extensive programming assignments provide an understanding of the entire process of client/server development including interface prototyping, program design, implementation of both client and server programs, unit testing, and documentation. This course prepares students to meet immediate demands in solving complex computational problems.

Prerequisite: CMPSC 122 or CMPSC 132

CMPSC 296: Independent Studies

1-18 Credits/Maximum of 18

Creative projects, including research and design, which are supervised on an individual basis and which fall outside the scope of formal courses.

CMPSC 297: Special Topics

1-9 Credits/Maximum of 9

Formal courses given infrequently to explore, in depth, a comparatively narrow subject which may be topical or of special interest.

CMPSC 299: Foreign Studies

1-12 Credits/Maximum of 12

Courses offered in foreign countries by individual or group instruction.

International Cultures (IL)

CMPSC 302: Intermediate Visual Programming

3 Credits

OO programming, visual programming, classes, objects, ADTs, inheritance, recursion, regular expressions, user-defined controls, documentation, testing, verification, productivity tools. CMPSC 302 Intermediate Visual Programming (3) This course forms the second of a two course sequence of courses for non-major students. It is designed to build upon concepts and skills presented in the first course, CMPSC 102, with the intent of enabling the student to develop larger scale programs and interface with databases and Web servers using a visual programming language. Some of the topics covered in this course will be object-oriented programming, inheritance, string manipulation, regular expressions, creating custom controls, creating controls dynamically, interfacing with databases and using an appropriate platform, such as ASPX.net to create web pages.This course forms the second of a two course sequence of courses for non-major students. It covers advanced features of the target language , building larger scale programs and interfaces to databases and web servers. It builds on the skills covered in CMPSC-102 and introduces creating new controls, dynamically placing controls at run time, arrays and lists of controls, the use of regular expressions, more in-depth treatment of classes and objects, including inheritance and polymorphism, multi-dimensional arrays, lists, unit testing and project deployment.

Prerequisite: CMPSC102 or CMPSC121

CMPSC 311: Introduction to Systems Programming

3 Credits

Unix system programming in C; organization of programs and data; program analysis and support tools; software standards; common system functions. CMPSC 311 CMPSC 311 Introduction to Systems Programming (3) System Programming concerns the development of software components and methods for their combination, independent of any particular application. This course will provide information and experience required to understand, design and implement components of large software systems.In general, students should be able to evaluate design alternatives according to standard practice, specifications, performance analysis, robustness, etc. To concentrate attention, we investigate one system and one programming language in detail, through demonstration programs, short- and long-term programming assignments. The specific system is Unix, a family of operating systems forming a complete standardized programming environment based on the idea of software tools. The specific language is C, which is widely used for operating system implementations, and which forms the basis for the C++ and Java languages studied in the prerequisite courses. This will help students understand operating system services available to application programmers, and provide a firm ground for study of operating systems in general.There are several themes of the course: (1) Understand computer systems, especially low-level influences on high-level goals. This includes the machine-level representation of programs and data structures; the memory hierarchy and its impact on performance; access to stored information via file systems, and access to other computer systems via networks.(2) Understand existing system software and software standards, especially the UNIX toolset. This includes preparing a program (editors, static analysis, development environments); running a program (compilers and interpreters, assembler, linker, loader, debugger, profiler, tracer); controlling parts of a program (memory management, threads); communication between programs (within one system using signals, between systems using sockets and communication protocols); and combinations of software tools with scripting languages.(3) Understand "real code", such as selections from the Linux operating system kernel and GNU utilities and libraries, and through comparative selections from Solaris, Linux, and Mac OS X.(4) Understand system performance, including experiments on program performance and optimization techniques.

Prerequisite: CMPSC221

CMPSC 312: Computer Organization and Architecture

3 Credits

Data representation, digital logic, instruction set/control logic, machine/assembly languages, advanced architectures, memory hierarchy, I/O devices, overall system design.

Prerequisite: CMPSC121 or equivalent

CMPSC 313: Assembly Language Programming

3 Credits

Program design, addressing modes, subroutines, parameter passing, stacks, bit manipulation, text processing, DOS functions, macros, I/O, high level language interfaces. CMPSC 313CMPSC 313 Assembly Language Programming (3)This is a course in assembly language programming for IBM PCs and compatibles. Students will gain experience writing efficient, well-documented programs that are easily maintained. The course investigates the architecture and instruction set of a typical microcomputer based on the Intel 80x86 microprocessors. Topics include the basic structure of computers, the internal behavior of computers, program design, testing, debugging, machine architecture, addressing, BCD and binary arithmetic, subroutines and parameter passing, stacks, text processing, bit manipulation, DOS functions, macros, I/O routines, high level language interfaces and the assembly process.This course is important because assembly language is often used in programs where small size or fast execution is critical. Knowledge of assembly language is also useful in debugging programs written in high level languages. It also helps bridge the gap between hardware and high level languages.After successfully completing CMPSC 313, the student should be able to: explain the 80x86 architecture, including registers and segment:offset addressing; describe different ways data are represented in a computer and work with binary and hexadecimal numbers; describe the functions of an assembler; implement program designs in 80x86 assembly language, including: writing, documenting, testing and debugging a program in PC assembly language; manipulating strings; coding basic algorithms such as searching and sorting in assembly language; calling and passing parameters to subroutines; utilizing DOS functions; and interfacing with a high level language; explain how the underlying hardware affects software design and performance; appreciate the factors that contribute to program efficiency.Students will be evaluated on homework (35% of grade), semester exams (35%), and a final comprehensive exam (30%).The major only requires that a student have experience with assembly language programming. This course is intended for students who have not had any experience with assembly language programming before entering the program. It will also serve as an elective. The material learned in this course is beneficial in understanding concepts in the required courses CMPSC 422, CMPSC 472, and CMPSC 460, as well as in the elective courses CMPSC 428 and CMPSC 470.No special facilities are required for this course. The software necessary is available in the computer labs or for students to use at home. This course will be offered once per year, with an expected enrollment of 55.

Prerequisite: CMPSC312

CMPSC 335: Fundamentals of Communication Networks

3 Credits

Introduction to the composition of communciation networks, including trans- mission mediums and protocols, transfer methods, topologies and software, and communications hardware.

Prerequisite: 3 credits of programming

CMPSC 360: Discrete Mathematics for Computer Science

3 Credits

Discrete mathematics and foundations for modern computer science. Topics include sets, relations, logic, algorithms, graphs, finite state machines and regular expressions.

Concurrent: CMPSC122

CMPSC 395: Internship

1-18 Credits/Maximum of 18

Supervised off-campus, nongroup instruction including field experiences, practica, or internships. Written and oral critique of activity required.

CMPSC 397: Special Topics

1-9 Credits/Maximum of 9

Formal courses given infrequently to explore, in depth, a comparatively narrow subject which may be topical or of special interest.

CMPSC 399: Foreign Studies

1-12 Credits/Maximum of 12

Courses offered in foreign countries by individual or group instruction.

International Cultures (IL)

CMPSC 402: UNIX and C

3 Credits

UNIX OS including file system, utilities, and shell scripting; C programming, including I/O, pointers, arrays, dynamic memory, macros, and libraries. CMPSC 402CMPSC 402 Unix and C (3)The primary goal of the course is to provide students with sufficient information to enable them to write structured and readable C programs for realistic applications. In particular, after completing the course students should be familiar with and be able to use pointers and dynamic memory management techniques.A secondary goal is for the students to be fluent in using the Unix operating system, particularly those parts needed for program development.Students will be evaluated on midterm and final exams, and four to five problem sets. The exams will be worth 50% of the total grade and the problem sets will be worth the remaining 50%.This course is an elective in the computer science (COMP) BS program. Students in other programs generally use it as an elective. Students cannot take this course after having taken CMPSC 422.No special Facilities, are needed for this course. The course will usually be offered once a year with an expected enrollment of 30-40.

Prerequisite: CMPSC121

CMPSC 410: Programming Models for Big Data

3 Credits

Recommended Preparations: DS 310; CMPSC 448 This course introduces modern programming models and related software stacks for performing scalable data analytics and discovery tasks over massive and/or high dimensional datasets. The learning objectives of the course are that the students are able to choose appropriate programming models for a big data application, understand the tradeoff of such choice, and be able to leverage state-of-the art cyber infrastructures to develop scalable data analytics or discovery tasks. Building on data models covered in DS 220, this course will introduce programming models such as MapReduce, data flow supports for modern cluster computing environment, and programming models for large-scale clustering (either a large number of data samples or a large number of dimensions). Using these frameworks and languages, the students will learn to implement data aggregation algorithms, iterative algorithms, and algorithms for generating statistical information from massive and/or high-dimensional data. The realization of these algorithms will enable the students to develop data analytic models for massive datasets.

Enforced Prerequisite at Enrollment: (CMPSC 122 or CMPSC 132) and DS 220. Recommended Preparation: DS 310 or CMPSC 448
Cross-listed with: DS 410

CMPSC 412: Data Structures Lab

1.5 Credits

Programming with common data structures; recursion; stacks, queues, dictionaries, priority queues; string searching and manipulation; sorting; trees; combinatorics.

Concurrent: CMPSC462or CMPSC465

CMPSC 413: Algorithms Lab

1.5 Credits

Programming with common algorithm design techniques; divide and conquer, greedy method, dynamic programming, and tree and graphy traversals.

Concurrent: CMPSC463

CMPSC 414: Contest Programming

1 Credits/Maximum of 4

Programming Contest Questions; Common Data Structures; Strings; Sorting; Searching; Combinatorics; Number Theory; Graph Algorithms; Dynamic Programming. This course provides hands-on practice with a wide range of computer science topics that are used to solve programming contest questions. These topics include common data structures; strings; sorting; combinatorics; number theory; graph traversal and other algorithms; dynamic programming. In the course, students will solve a range of programming contest questions, both using an online judge in practice sessions and during actual programming contests. In addition to solving programming problems, the course time will also be used to explore topics mentioned above and the relationship to specific problems, solution techniques, and the analysis of proposed solutions to specific problems. This course is envisioned to be a hands-on lab, with instructor supported, self-guided study. The course topics will be chosen from topics that commonly appear in current programming contests, including but not limited to: - Contest Programming - Common Data Structures - Strings - Sorting - Combinatorics - Number Theory - Graph Traversal and Other Algorithms - Dynamic Programming

CONCURRENT: CMPSC 221

CMPSC 421: Net-centric Computing

3 Credits

This course introduces JavaScript and AJAX for creating Rich Internet Applications, and XML for client-server communication and Web Services. CMPSC 421 Net-centric Computing (3) This course will build on the client-server computing concepts and techniques that students learned about in prerequisite courses. The goal of the course is to introduce students to the most significant and fundamental of those technologies that are used in the computing paradigm known by a number of terms including "Net-centric", "Web 2", and "cloud" computing. On the client: We will use Dynamic XHTML, Cascading Style Sheets, JavaScript and AJAX to develop the client side of Rich Internet (or

Web) Applications. For client-server-communication: We will learn how to create and validate XML documents and use these as the primary language for transmission of data from the server to the client. We will also consider how JavaScript Object Notation (JSON) can sometimes be used as a viable alternative to XML for server to client data transmission. On the server: We will learn about a variety of server-side technologies for consuming, storing, transforming, and generating content. We will use the three main types of XML parsers to consume, transform, and generate XML; we will use XSL and XPath to style and transform XML; we will use XML binding tools to convert XML to and from classes (in some high-level language); we will use Data Access Objects and object-relational mapping tools for data persistence. We will learn how servers use Web services and RSS feeds to provide XML structured content, and we will consume existing Web services and RSS feeds and produce simple Web services.

Prerequisite: CMPSC221 or SWENG311

CMPSC 426: Object-oriented Design

3 Credits

Object-oriented analysis and design; design patterns such as creational, structural, and behavioral patterns; UML; and unified process. CMPSC 426 Object-Oriented Design (3) The primary goal of this course is to study the object-oriented design paradigm, including modeling languages, classes and objects, the inheritance relationship, polymorphism, and software engineering topics relating to object-oriented design. Study of this topic should provide a solid understanding of object-orientation for students to use in studying diverse topics such as operating systems, software engineering, and database design. This course is an elective for students in the BS COMP program. The course builds on topics learned in earlier object-oriented programming courses.

Prerequisite: CMPSC221 , CMPSC462

CMPSC 430: Database Design

3 Credits

The main goal of this course is to explore the relational database model, with special emphasis on the design and querying of relational databases. Secondary goals include exploration of the mathematical basis for relational databases and exploration of the relationship of database to the rest of computer science. Study of these topics should improve student skills in programming, modeling the structure of data and using and administering databases. After completing CMPSC 430 the student should be able to: (1) Demonstrate comprehension of general database concepts (2) Identify key issues in developing database systems and applications (3) Explain the general organization of a relational database and explain the functions of the basic relational operators (4) Use query languages, in particular, Structured Query Language (SQL), to define, maintain, and query relational databases (5) Model a relational database through entity-relationship (ER) modeling and construct ER diagrams (6) Explain the methodologies used to conceptualize and design database systems (7) Apply decomposition and analyze/construct functional dependencies (8) Demonstrate an understanding of the concepts of relation normalization and the supporting fundamental knowledge of efficient database implementation

Concurrent Courses: CMPSC 462

CMPSC 431W: Database Management Systems

3 Credits

Topics include: conceptual data modeling, relational data model, relational query languages, schema normalization, database/Internet applications, and database system issues.

Prerequisite: CMPSC 221;ENGL 202C
Writing Across the Curriculum

CMPSC 436: Communications and Networking

3 Credits

Data transmission, basic signaling, data encoding, error control, communication protocols, security, network topologies, routing, switching, internetworking, emerging high speed networks. CMPSC 436CMPSC 436 Communications and Networking (3)This course introduces the elements and architecture of computer and data communication networks, demonstrates the fundamental principles of computer networking, and provides experience in the practical use of current networking technology. Topics in this course include: data communications (basic signaling, data transmission, data encoding, errors and error control), communications architecture and protocols (communication protocols, internetworking, transport protocols, layered network architecture, network security) and computer networks (WANs, LANs, network topologies, internetworking, routing and switching strategies and emerging high speed networks).After taking CMPSC 436, students should be able to:1) understand the fundamentals of networking concepts and terminology2) define and contrast the classifications local area network (LAN), metropolitan area network (MAN), and wide area network (WAN)3) name and describe basic networking elements4) define the roles of clients, servers, and peers as they relate to computer networks5) define the term "protocol" and explain how it relates to computer networks6) identify specific network management areas and describe the organizational issues relating to each of themStudents will be evaluated on homework (35% of grade), semester exams (35%), and a final comprehensive exam (30%).This course is an elective in the computer science (COMP) BS curriculum. This course is intended to be taken by second semester juniors or seniors.No special facilities are required for this course. This course will be offered once per year, with an expected enrollment of 60.

Prerequisite: CMPSC312

CMPSC 438: Computer Network Architecture and Programming

3 Credits

Network architectures, communication protocols, internetworking, network security, client-server computing, web application development, programming with APIs. CMPSC 448 Machine Learning and Algorithmic AI (3) Machine learning and artificial intelligence are closely-related branches of computer science that deal with the development of software that can "learn" how to perform useful tasks from prior data. Machine learning is mostly concerned with inferring trends from data in order to use them for future predictions. Artificial intelligence is mostly concerned about how to use knowledge gained from previous data to achieve specific goals.This course provides an introduction to important concepts in machine learning and artificial intelligence, as well as probabilistic and mathematical tools needed for applications of technology from both fields. Students will learn about important models and algorithmic frameworks used in machine learning such as linear

models, neural networks, decision trees, support vector machines, k-nearest neighbor, adaboost, k-means clustering, and agglomerative clustering as well as methods for evaluating and tuning these models. Students will also learn about key artificial intelligence concepts such as A* search and reinforcement learning which are used by software agents (such as game AI's) to navigate and explore their environment.

Prerequisite: CMPSC221 , CMPSC312

CMPSC 440: Data-driven Security Analytics

3 Credits

Today's security event monitoring and correlation tools are being taxed more than ever before. The constant deluge of data being output by the many devices in our security infrastructures and input into log managers, SIEM (Security information and event management) and other security management interfaces creates quite a load on our analytics systems. In the noise of all this security and event-related data, it is difficult to sort through and obtain real, actionable events that need attention. The market is in need of a new generation of security tools - ones that process much larger datasets, are capable of deep-dive analytics and rely more on intelligence than attack signatures. The proposed course will cover the fundamental data science techniques that are applied to enhance security intelligence. This course will complement the existing computer science and software engineering curriculum. It also prepares students for both a career as a data scientist and as a cybersecurity analyst.

Prerequisites: CMPSC 335; CMPEN 461

CMPSC 441: Artificial Intelligence

3 Credits

Problem solving, search techniques including local search and genetic algorithms, knowledge representation, planning, learning, and neural networks. CMPSC 441 Artificial Intelligence (3) The primary goals of this course are (1) to provide the students with an introduction to Artificial Intelligence concentrating on some fundamental areas of AI, and (2) to provide the students with a working knowledge of LISP so that they can investigate some basic problems in AI using LISP as a vehicle language.

Prerequisite: CMPSC122 , CMPSC360

CMPSC 442: Artificial Intelligence

3 Credits

This course provides an overview of the foundations, problems, approaches, implementation, and applications of, artificial intelligence. Topics covered include problem solving, goal-based and adversarial search, logical, probabilistic, and decision theoretic knowledge representation and inference, decision making, and learning. Through programming assignments that sample these topics, students acquire an understanding of what it means to build rational agents of different sorts as well as applications of AI techniques in language processing, planning, vision.

Enforced Prerequisite at Enrollment: CMPSC 221. Enforced Concurrent at Enrollment: CMPSC 465
Cross-listed with: DS 442

CMPSC 443: Introduction to Computer and Network Security

3 Credits

Introduction to theory and practice of computer security with an emphasis on Internet and operating system applications.

Prerequisite: CMPSC473 , CMPEN362

CMPSC 444: Secure Programming

3 Credits

Secure software design principles/practices, common threats, applied cryptography, trust management, input validation, OS-/programming language- specific issues, software validation. CMPSC 444 Secure Programming (3) This course presents an overview of the principles and practices of secure software design. The course begins with a presentation of overarching principles of secure software development that enable the design, implementation, and testing of secure systems that can withstand attacks. These principles and strategies for realizing them will be illustrated through an analysis of common security issues and pitfalls in the software development process. The course will cover a variety of programming languages including C/C++, Java, and scripting languages; different classes of systems including standalone applications, client/server systems, and peer-to-peer applications; and development issues specific to different operating systems. Students will develop and analyze programs that demonstrate security principles, strategies, coding techniques, and the use of tools that can help make code more resistant to attacks.

Prerequisites: CMPSC 221

CMPSC 445: Applied Machine Learning in Data Science

3 Credits

Applied machine learning techniques are used in many different areas, such as the classification, visualization and analysis of data, clustering, and understanding of natural languages for human-computer interactions. These applications are crossing the boundaries of computer science and data science. Big technology firms have all started offering their own cloud machine learning platforms. This course will start with an overview of supervised and unsupervised learning, and introduce the associated libraries. It covers basic machine learning concepts, tasks, and workflow using an example classification problem based on K-nearest neighbors, Naïve Bayes, Support Vector Machine (SVM), K-means, and implementation using Python libraries. Natural language processing (NLP) techniques including n-gram models, grammar, parse trees, and part-of-speech tagging will be discussed. The issue of dimensionality of data will be discussed, and the task of clustering data, as well as supervised approaches for creating predictive models will be described, and learners will be able to apply Python predictive modeling methods while understanding process issues related to data generalizability (e.g. cross validation, overfitting). The course will also look at more advanced techniques, such as neural network, feed-forward network, back-propagation and deep learning with cloud AI services. Technological differences between using cloud services at a higher level of abstraction and coding locally will be discussed. Students will be able to identify the difference between a supervised (classification) and unsupervised (clustering) technique, identify which technique they need to apply for a particular dataset, manage and understand data, and engineer features to meet that need. Students will work in teams to develop web applications

that use industry standard cloud services provided by one of the AI cloud service providers.

Prerequisites: STAT 318, MATH 220, (CMPSC 122; CMPSC 132)

CMPSC 447: Software Security

3 Credits

This course explores the fundamental concepts and engineering processes of software development and testing to produce software that is designed for security. This course is intended as a senior-level course for computational majors such as computer science and computer engineering since it covers the exploitation of programs based on computer architecture, systems, and software concepts. First, software engineering considerations associated with a variety of software vulnerabilities will be analyzed, along with defensive programming techniques to avoid such vulnerabilities. The next part of this course will introduce systematic software engineering principles for building secure software to defend its attack surface, such as reference monitors, privilege separation, information flow, and program verification. The third part will focus on methods for security testing of software including fuzz testing, symbolic execution, grey-box testing, and forensics. The final week of the course will examine adding security into the software engineering life cycle. The design and implementation of techniques to develop reference monitors, information-flow secure programs, testing mechanisms and enhancements, as well as defensive programming against prominent software vulnerabilities will be studied and analyzed. Upon completion of the course students will be able to critically analyze the design and implementation of software for security flaws and build security mechanisms to prevent exploitation of such flaws.

Prerequisites: CMPSC 443 Recommended preparations: CMPSC 360

CMPSC 448: Machine Learning and Algorithmic AI

3 Credits

Evaluation and use of machine learning models; algorithmic elements of artificial intelligence.

Prerequisite: STAT 319 or STAT 415 and CMPSC122 or prior programming experience

CMPSC 450: Concurrent Scientific Programming

3 Credits

Problems of synchronization, concurrent execution, and their solution techniques. Design and implementation of concurrent software in a distributed system.

Prerequisite: CMPSC121 , CMPSC201 or CMPSC202 ; MATH 220 ; MATH 230 or MATH 231

CMPSC 451: Numerical Computations

3 Credits

ALGORITHMS FOR INTERPOLATION, APPROXIMATION, INTEGRATION, NONLINEAR EQUATIONS, LINEAR SYSTEMS, FAST FOURIER TRANSFORM, AND DIFFERENTIAL EQUATIONS EMPHASIZING COMPUTATIONAL PROPERTIES AND IMPLEMENTATION. STUDENTS MAY TAKE ONLY ONE COURSE FOR CREDIT FROM MATH 451 AND 455.

Prerequisite: 3 credits of programming; MATH 230 or MATH 231
Cross-listed with: MATH 451
Bachelor of Arts: Quantification

CMPSC 452: Numerical Analysis

3 Credits

Algorithm efficiency and accuracy, function interpolation and polynomial approximation, numerical differentiation and integration, initial-value problems, and approximation of eigenvalues. CMPSC 452 Numerical Analysis I (3) General principles for evaluating the accuracy and efficiency of floating point algorithms; methods for solving single equations and systems of linear equations, function interpolation and polynomial approximation, numerical differentiation and integration, initial-value problems, approximation of eigenvalues.

Prerequisite: MATH 220

CMPSC 455: Introduction to Numerical Analysis I

3 Credits

Floating point computation, numerical rootfinding, interpolation, numerical quadrature, direct methods for linear systems. Students may take only one course for credit from MATH 451 and MATH 455.

Enforced Prerequisite at Enrollment: (CMPSC 201 or CMPSC 202 or CMPSC 121 or CMPSC 131) and MATH 220 and (MATH 230 or MATH 231)
Cross-listed with: MATH 455
Bachelor of Arts: Quantification

CMPSC 456: Introduction to Numerical Analysis II

3 Credits

Polynomial and piecewise polynomial approximation, matrix least squares problems, numerical solution of eigenvalue problems, numerical solution of ordinary differential equations.

Enforced Prerequisite at Enrollment: MATH 455
Cross-listed with: MATH 456
Bachelor of Arts: Quantification

CMPSC 457: Computer Graphics Algorithms

3 Credits

Graphics systems/hardware, color models, transformations, projections, clipping, hidden line/surface removal, aliasing, parametric curves/surfaces, 3D modeling animation. CMPSC 457CMPSC 457 Computer Graphics Algorithms I (3)Concepts and techniques needed to draw geometrical objects with a discrete device: Coordinate systems, clipping, curves and regions, geometric transformations, parallel and projective projections, hidden line and surface removal, animation.

Prerequisite: CMPSC122 ; MATH 220

CMPSC 458: Fundamentals of Computer Graphics

3 Credits

Fundamentals of computer graphics: input/output devices, transformation, projection, clipping, hidden line/surface elimination.

Prerequisite: CMPSC311 ; MATH 220 ; MATH 230 or MATH 231

CMPSC 459: Scientific Visualization

3 Credits

Visualization techniques for data analysis and presentation. Applying visualization and perceptual theory. Using extending platform independent visualization software. CMPSC 459CMPSC 459 Scientific Visualization (3)Visualization of scientific data and processes has always been important for gaining insights into scientific phenomena. Historically, such visualization has taken place in the scientist's imagination and was then rendered in drawings, graphs and diagrams. The rapid advance of computer technology, and in particular, computer graphics, has made new tools available to the scientist to aid in the interpretation and communication of scientific information. In this course students will study a variety of computer graphics, scientific visualization, and virtual reality techniques and apply them to scientific visualization projects. The projects will be drawn from all of the sciences and the resulting projects will then be available to faculty and students to use as tools in their disciplines.The prerequisites for this course are CMPSC 122. Students will apply the writing skills gained in ENGL 202C and refine them in the context of scientific writing. They will also have the opportunity to apply the knowledge and skills gained in CMPBD 360 and its predecessors, CSE 103 and CSE 120 within the context of a significant natural science or mathematical visualization problem. Software and languages used in this course will change as the discipline of scientific visualization evolves. Currently, programming will be done in C++ and Java; VRML and other virtual reality languages, and scientific specialty languages such as IDL, muPad, xpp, Mathematica, Maple, etc. Projects initiated in this course can form the basis for further development as a 494 research project.The course will take advantage of a variety of computing platforms available at Behrend including Windows NT and Unix.

Prerequisite: CMPSC122

CMPSC 460: Principles of Programming Languages

3 Credits

Design and implementation of high level programming languages and survey of language paradigms including imperative, functional, and object-oriented programming. CMPSC 460 Principles of Programming Languages (3) The primary topics of this course include run-time systems for imperative programming languages and aspects of the object-oriented, functional and declarative paradigms that have applications in industrial software development. Study of these topics should improve student skills in programming, debugging and problem solving.

Prerequisite: CMPSC312 ; CMPSC462; Concurrent: CMPSC469

CMPSC 461: Programming Language Concepts

3 Credits

Fundamental concepts of programming language design, specifications, and implementation; programming language paradigms and features; program verification.

Prerequisite: CMPSC221 ; CMPSC360

CMPSC 462: Data Structures

3 Credits

In-depth theoretical study of data structures such as balanced trees, hash tables, priority queues, B-trees, binomial heaps, and Fibonacci heaps.

CMPSC 462 Data Structures (3) The primary goals of this course are (1) to provide the students with a set of basic data structures useful in the design of efficient algorithms, and (2) to provide the students with the ability to design and analyze new data structures as needed to solve problems. The secondary goal of this course is to introduce basic algorithm analysis techniques to prepare the students for the follow up course CMPSC 463, Design and Analysis of Algorithms. This is a required course in the BS COMP program. It is also a prerequisite for a number of other courses in the COMP program such CMPSC 463, 460, 430, etc.

Prerequisite: CMPSC360

CMPSC 463: Design and Analysis of Algorithms

3 Credits

Recurrences, algorithms design techniques, searching, sorting, selection, graph algorithms, NP-completeness, approximation algorithms, local optimization algorithms. CMPSC 463 Design and Analysis of Algorithms (3) The primary goals of this course are (1) to provide the students with fundamental techniques for designing and analyzing algorithms, and (2) to introduce some techniques for dealing with inherently intractable problems.This is a required course in the BS COMP program.

Prerequisite: CMPSC462; Concurrent: MATH 318, STAT 301 or STAT 318

CMPSC 464: Introduction to the Theory of Computation

3 Credits

Computability/Complexity: finite automata, regular & context-free languages, Turing machines, Church-Turing Thesis, undecidability, reducibility, completeness, time/space complexity, P versus NP. CMPSC 464 Introduction to the Theory and Computation (3)CMPSC 464 introduces students to an essential part of theoretical computer science: how to define abstract mathematical models of computational devices (automata), how to characterize their computational power by studying the family of languages that they can recognize (formal languages), and what the limitations of even the most powerful computational devices are (computability). The course studies regular languages by means of deterministic and nondeterministic finite-state automata and regular expressions; it studies context-free languages through the use of context-free grammars and pushdown automata; and it studies computability by means of Turing machines and recursive and recursively-enumerable languages. The unsolvability of the halting problem for Turing machines is proved by a diagonalization argument, and this result is then used to show that various problems about languages are unsolvable, such as the problem of determining whether two context-free grammars generate the same language. Finally, the concept of computational complexity is introduced, and the classes P and NP are defined. (Informally, the former class consists of problems that can be solved computationally in a manageable amount of time, and the latter consists of problems for which a proposed solution can be verified in a manageable amount of time.) The concept of an NP-complete problem is defined, and some specific problems are proved to be values to the variable of a Boolean formula that will make the formula true).

Prerequisite: CMPSC465

CMPSC 465: Data Structures and Algorithms

3 Credits

Fundamental concepts of computer science: data structures, analysis of algorithms, recursion, trees, sets, graphs, sorting.

Prerequisite: CMPSC122 ; CMPSC360 or MATH 311W

CMPSC 467: Factorization and Primality Testing

3 Credits

Prime sieves, factoring, computer numeration systems, congruences, multiplicative functions, primitive roots, cryptography, quadratic residues. Students who have passed MATH 465 may not schedule this course.

Enforced Prerequisite at Enrollment: MATH 311W
Cross-listed with: MATH 467
Bachelor of Arts: Quantification

CMPSC 469: Formal Languages with Applications

3 Credits

Regular, context free, recursive, and recursively enumerable languages; associated machine models; applications. CMPSC 469 Formal Languages with Applications (3) The primary goal of this course is to explore formal language theory, including regular, context free and recursively enumerable languages. Notations for specifying these languages (regular expressions, finite automata, context free grammars and turing machines) are emphasized. Applications of these languages, including pattern recognition, scanning, parsing, specification of programming language syntax and Unix shell programming, are also discussed. Study of these topics should provide a solid theoretical basis for students to draw on in studying diverse areas such as algorithm analysis, complexity theory and compiler construction.

Prerequisite: CMPSC360

CMPSC 470: Compiler Construction

3 Credits

Compiler design and implementation; scanning, parsing, semantic analysis, optimization (including static analysis), code generation, garbage collection, and error detection. CMPSC 470 Compiler Construction (3) The primary topics of this course are areas of compiler construction that are applicable both in building compilers and in many other areas of computer science. Both the concepts and the implementation of these techniques will be emphasized. Study of these topics should improve student skills in programming, debugging and software engineering.This course is an elective for students in both the BS COMP and MS COMP programs. The course builds on concepts learned in earlier programming, data structure and computer organization courses.

Prerequisite: CMPSC221 , CMPSC312 , CMPSC462 , CMPSC469

CMPSC 471: Introduction to Compiler Construction

3 Credits

Design and implementation of compilers; lexical analysis, parsing, semantic actions, optimization, and code generation.

Prerequisite: CMPSC461

CMPSC 472: Operating System Concepts

3 Credits

Theoretical and practical issues of operating systems design and implementation, process management, concurrent programming, memory management, scheduling, I/O, and security. CMPSC 472CMPSC 472 Operating Systems Concepts (3)A course on operating systems is an essential part of a computer science education. This course is intended as an introduction to study the concepts, structure and mechanisms that underlie operating systems. A tremendous range and variety of computer systems exist for which operating systems are designed. Rather than focus on individual operating systems, this course discusses the key mechanisms of modern operating systems, the types of design trade-offs and decisions involved in operating system design and the context within which the operating system functions.After completing CMPSC 472 the student should be able to:(1) describe and understand the four major components of an operating system: process management (including synchronization, scheduling, mutual exclusion, deadlocks and concurrency), input/output (including disk scheduling and disk 1/0), memory management (including virtual memory, paging, segmentation and addressing) and management of the file systems(2) describe and understand how a centralized operating system functions(3) describe and understand the various components of an operating system(4) describe the various goals of protection and the security problem in general(5) compare centralized operating systems with distributed operating systemsStudents will be evaluated on homework (35% of grade), semester exams (35%), and a final comprehensive exam (30%).This course is required in the computer science (COMP) BS curriculum. It is intended for seniors to take this course in their fall semester. This course is also an admission requirement for the (COMP) MS program.No special facilities are required for this course. The software necessary is available in the computer labs or for students to use at home. This course will be offered once per year, with an expected enrollment of 80.

Prerequisite: CMPSC312 ; CMPSC462

CMPSC 473: Operating Systems Design & Construction

3 Credits

Design and implementation of computer operating systems; management of various system resources: processes, memory, processors, files, input/output devices.

Prerequisite: CMPSC311 ; CMPEN331

CMPSC 474: Operating System & Systems Programming

3 Credits

Operating Systems overview and principles; processes and signals; concurrency and synchronization; memory and file management; client-server computing; scripts; systems-programming.

Prerequisite: CMPSC122 ; CMPSC312

CMPSC 475: Applications Programming

3 Credits

Development of software for devices including smart phones, tablets, handheld units, and other general purpose computing platforms.

Prerequisite: CMPSC221 ; CMPSC311 or CMPSC312 ; CMPSC462 or CMPSC465

CMPSC 483: Software Design Methods

3 Credits

Applications of scientific knowledge and methods in the design and construction of computer software using engineering concepts.

Prerequisite: CMPSC 221;CMPSC 465;ENGL 202C
Writing Across the Curriculum

CMPSC 484: Computer Science Senior Project I

2 Credits

Computer science capstone project with documentation emphasis. CMPSC 484 CMPSC 484 Computer Science Senior Project I (2) This course is phase one preparation for completing a design for a project to serve as the capstone to the computer science degree program. The course provides instruction and specification of a simulated real-world work environment and associated activities to employ and integrate computer science concepts. Technical instruction and delivered products will be required.Students enrolled in the program will: 1) produce a design for a significant senior project using a cooperative, team approach, 2) present concepts, progress, and products to and interact with peer and faculty review boards. The course will: 1) provide the student with an opportunity to work in a team environment designed around sound development practice, 2) present to students current team organization and management techniques, 3) describe various forms of written communication targeted to different audiences, and 4) reinforce the technical knowledge attained through the computer science curriculum.

Prerequisite: ENGL 202C ; CMPSC221 ; CMPSC465

CMPSC 485: Computer Science Senior Project II

3 Credits

Computer science capstone project with documentation emphasis.

Prerequisite: CMPSC484
Writing Across the Curriculum

CMPSC 487: Software Engineering and Design

3 Credits

Software development process, life cycle; requirements analysis, specification, design, prototyping, testing, project management, and documentation. CMPSC 487W Software Engineering and Design (3) The primary goal of this course is to familiarize students with the wide variety of techniques and methodologies used in software engineering to assist in the development of large software systems. Issues discussed include the human factors involved in developing software, models of the software development process, the use of formal methods in software engineering, software validation and verification, and software maintenance.A second goal is to help students understand the importance of written communication in software engineering, and to provide opportunities for students to improve the quality of their writing - specifically in describing software systems. The primary means of accomplishing this goal is a semester long project in which students write requirements for a large software system. In writing these requirements, students describe the system for non-technical readers (clients and users) and specify it for technical readers (other system developers).A final goal is to emphasize the role of teams in software development. Modern software systems are simply too large to reasonably be produced by one person, so the ability to work as part of a team is vital. To support achieving this goal, techniques and tools for working in groups are discussed in the course, and students work on the semester project in teams.This course is a required course in the computer science (COMP) BS curriculum, and is intended to be taken by seniors as the capstone course for the major. As such, the course integrates material from many (potentially all) of the undergraduate computer science courses. This course is also available as an elective for students in the MS COMP program.

Prerequisite: ENGL 202C , CMPSC221 , CMPSC462
Writing Across the Curriculum

CMPSC 488: Computer Science Project

3 Credits

Project design and implementation with an emphasis on team work, documentation, and the employment and integration of computer science concepts. CMPSC 488 Computer Science Project (3) This class provides a hands-on experience designing and developing a real-world software system. The course will emphasize collaboration and teamwork to employ and integrate computer science concepts. Students will work on a project that will serve as the capstone to the computer science degree program. Technical instruction, research, software implementation and delivered products will be required.

Prerequisite: CMPSC487W

CMPSC 494: Senior Honors Thesis

1-6 Credits/Maximum of 6

Supervised Honors thesis research in computer science and engineering.

Honors

CMPSC 495: Internship

1-18 Credits/Maximum of 18

Supervised off-campus, nongroup instruction including field experience, practica, or internships. Written and oral critique of activity required.

Prerequisite: prior approval of proposed assignment by instructor
Full-Time Equivalent Course

CMPSC 496: Independent Studies

1-18 Credits/Maximum of 18

Creative projects, including research and design, which are supervised on an individual basis and which fall outside the scope of formal courses.

CMPSC 497: Special Topics

1-9 Credits/Maximum of 9

Formal courses given infrequently to explore, in depth, a comparatively narrow subject which may be topical or of special interest.

CMPSC 498: Special Topics

1-9 Credits/Maximum of 9

Formal courses given infrequently to explore, in depth, a comparatively narrow subject that may be topical or of special interest.

CMPSC 499: Foreign Studies

1-12 Credits/Maximum of 12

Courses offered in foreign countries by individual or group instruction.

International Cultures (IL)