Bachelor of Science in Computer Science

Program Description

The Department of Computer Science develops graduates who can process information in digital computers, design computer hardware and software, and work successfully with several different computing applications. The department offers the degree of Bachelor of Science in Computer Science. Within this degree, a student may choose (not required) one of six available emphasis areas in Business, Computer Engineering, Data Science, Research Honors, Robotics and Intelligent Systems, or Space.

BS in Computer Science

Computing is ubiquitous, impacting almost every aspect of modern life, and playing an important role in many technological advances. Computing jobs are among the highest paid, and computing professionals generally report high job satisfaction. Graduates from our program have found employment with many different types of companies including technology, engineering, and financial companies.

The CS degree at Mines is designed to be accessible to students with or without prior programming experience. The Introduction to Computer Science course introduces students to the building blocks of CS and provides a brief introduction to procedural programming in Python. The second computing course, Programming Concepts, emphasizes development of programming skills in an object-oriented language. The third introductory course, Data Structures, provides an understanding of the classic data representation schemes, algorithms, and algorithm analysis that form the foundation for all advanced work in computing.

Required CS courses provide the fundamental skills and knowledge that are critical to success in computing. These courses reflect a mixture of theory and practice, including discrete structures, design and analysis of algorithms, principles of programming languages, computer architecture, operating systems, software engineering, and database management. The capstone field session course provides students an opportunity to work in teams to create software products for real clients.

Elective courses in CS allow students to explore a variety of important computing topics, such as graphics and visualization, artificial intelligence, mobile applications, and web programming. Elective courses often relate to recent trends in computing, covering topics such as security, high performance computing, and machine learning.

Computing is a broad field with applicability to most science and engineering domains. The CS minor is designed for students in other disciplines to receive a solid grounding in the basics, which should enable them to apply their computing skills to solve problems in other domains.

PROGRAM EDUCATIONAL OBJECTIVES (BACHELOR OF SCIENCE IN COMPUTER SCIENCE)

In addition to contributing toward achieving the educational objectives described in the Mines' Graduate Profile, the Computer Science Program at Mines has established the following program educational objectives:

Students will demonstrate technical expertise within computer science by:

- Designing and implementing solutions to practical problems in science and engineering,
- Using appropriate technology as a tool to solve problems in computer science, and
- Creating efficient algorithms and well-structured computer programs.

Students will demonstrate a breadth and depth of knowledge within computer science by:

- Extending course material to solve original problems,
- Applying knowledge of computer science to the solution of problems, and
- Identifying, formulating, and solving computer science problems.

Students will demonstrate an understanding and appreciation for the relationship of computer science to other fields by:

- Applying computer science to solve problems in other fields,
- Working in cooperative multidisciplinary teams, and
- Choosing appropriate technology to solve problems in other disciplines.

Students will demonstrate an ability to communicate computer science effectively by:

- Giving oral presentations,
- Completing written explanations,
- Interacting effectively in cooperative teams,
- Creating well-documented programs, and
- Understanding and interpreting written material in computer science.

STUDENT LEARNING OUTCOMES

1. An ability to identify, formulate, and solve complex engineering problems by applying principles of engineering, science, and mathematics.
2. An ability to apply engineering design to produce solutions that meet specified needs with consideration of public health, safety, and welfare, as well as global, cultural, social, environmental, and economic factors.
3. An ability to communicate effectively with a range of audiences.
4. An ability to recognize ethical and professional responsibilities in engineering situations and make informed judgments, which must consider the impact of engineering solutions in global, economic, environmental, and societal contexts.
5. An ability to function effectively on a team whose members together provide leadership, create a collaborative and inclusive environment, establish goals, plan tasks, and meet objectives.
6. An ability to develop and conduct appropriate experimentation, analyze and interpret data, and use engineering judgment to draw conclusions.
7. An ability to acquire and apply new knowledge as needed, using appropriate learning strategies.

Primary Contact

Jeffrey Paone
303.384.2587
jpaone@mines.edu

https://cs.mines.edu/

Bachelor of Science in Computer Science Degree Requirements:

First Year

		lec	lab	sem.hrs
MATH111	CALCULUS FOR SCIENTISTS AND ENGINEERS I			4.0
HASS100	NATURE AND HUMAN VALUES			3.0
CHGN121	PRINCIPLES OF CHEMISTRY I			4.0
CSM101	FRESHMAN SUCCESS SEMINAR			1.0
CSCI128	COMPUTER SCIENCE FOR STEM			3.0
MATH112	CALCULUS FOR SCIENTISTS AND ENGINEERS II			4.0
PHGN100	PHYSICS I - MECHANICS			4.0
S&W	SUCCESS AND WELLNESS			1.0
EDNS151	CORNERSTONE - DESIGN I			3.0
CSCI200	FOUNDATIONAL PROGRAMMING CONCEPTS & DESIGN			3.0
				30.0

Sophomore

Fall		**lec**	**lab**	**sem.hrs**
MATH213	CALCULUS FOR SCIENTISTS AND ENGINEERS III			4.0
CSM202	INTRODUCTION TO STUDENT WELL-BEING AT MINES			1.0
PHGN200	PHYSICS II-ELECTROMAGNETISM AND OPTICS			4.0
CSCI358	DISCRETE MATHEMATICS			3.0
CSCI210	SYSTEMS PROGRAMMING			3.0
				15.0

Spring		**lec**	**lab**	**sem.hrs**
MATH225	DIFFERENTIAL EQUATIONS			3.0
HASS215	FUTURES			3.0
FREE	Free Elective			3.0
CSCI220	DATA STRUCTURES AND ALGORITHMS			3.0
Focus Area Course 1				3.0
				15.0

Junior

Fall		**lec**	**lab**	**sem.hrs**
MATH332	LINEAR ALGEBRA			3.0
MATH334	INTRODUCTION TO PROBABILITY			3.0
CSCI306	SOFTWARE ENGINEERING			3.0
Focus Area Course 2				3.0
Focus Area Course 3				3.0
				15.0

Spring		**lec**	**lab**	**sem.hrs**
EBGN321	ENGINEERING ECONOMICS*For the 2023 Catalog EBGN321 replaced EBGN201 as a Core requirement. EBGN321 was added to the core, but has a prerequisite of 60 credit hours. Students whose programs that required EBGN201 the sophomore year may need to wait to take EBGN321 until their junior year. For complete details, please visit: https://www.mines.edu/registrar/core-curriculum/			3.0
ELECTIVE	CULTURE AND SOCIETY (CAS) Mid-Level Restricted Elective			3.0
FREE	Free Elective			3.0
CSCI341	COMPUTER ORGANIZATION			3.0
CSCI406	ALGORITHMS			3.0
Focus Area Course 4				3.0
				18.0

Summer		**lec**	**lab**	**sem.hrs**
CSCI370	ADVANCED SOFTWARE ENGINEERING			5.0
				5.0

Senior

Fall		**lec**	**lab**	**sem.hrs**
ELECTIVE	CULTURE AND SOCIETY (CAS) Mid-Level Restricted Elective			3.0
CSCI400	PRINCIPLES OF PROGRAMMING LANGUAGES			3.0
Focus Area Course 5				3.0
Focus Area Course 6				3.0
Focus Area Course 7				3.0
				15.0

Spring		**lec**	**lab**	**sem.hrs**
ELECTIVE	CULTURE AND SOCIETY (CAS) 400 Level Restricted Elective			3.0
CSCI442	OPERATING SYSTEMS			3.0
Focus Area Course 8				3.0
CSCI ELECT	Computer Science Elective			3.0
FREE	Free Elective			3.0
				15.0

Total Semester Hrs: 128.0

Focus Areas

The Department of Computer Science offers seven focus areas:

1. Computer Science
2. CS + Business
3. CS + Computer Engineering
4. CS + Data Science
5. CS + Entrepreneurship & Innovation

6. CS + Robotics & Intelligent Systems
7. CS + Space

Computer Science electives can be chosen from any CSCI400-level course, any CSCI500-level course (with advisor approval), CSED435, EENG383,or MATH307. EDNS491 and EDNS492, when both courses are taken together, can both be counted as Computer Science electives. In a given focus area, a required course for that focus area cannot also be counted as a CSCI technical elective in that focus area.

Computer Science

Computer Science Electives		
CSCI ELECT	Computer Science Elective	3.0
CSCI ELECT	Computer Science Elective	3.0
CSCI ELECT	Computer Science Elective	3.0
CSCI ELECT	Computer Science Elective	3.0
CSCI ELECT	Computer Science Elective	3.0
Free Electives		
FREE	Free Elective	3.0
FREE	Free Elective	3.0
FREE	Free Elective	3.0
Total Semester Hrs		**24.0**

CS + Business

Required Data Courses		
CSCI403	DATA BASE MANAGEMENT	3.0
CSCI413	DATA SCIENCE FOR COMPUTER SCIENCE	3.0
CSCI475	INFORMATION SECURITY AND PRIVACY	3.0
Front End Application		
CSCI448	MOBILE APPLICATION DEVELOPMENT	3.0
or CSCI445	WEB PROGRAMMING	
or CSCI446	WEB APPLICATIONS	
Business Core		
EBGN201	PRINCIPLES OF ECONOMICS	3.0
or EBGN280	INTRODUCTION TO BUSINESS ANALYTICS	
or EBGN305	SURVEY OF ACCOUNTING	
or EBGN308	PRINCIPLES OF MARKETING	
or EBGN309	FUNDAMENTALS OF MANAGEMENT	
Business Application		
EBGN320	ECONOMICS AND TECHNOLOGY	3.0
or EBGN315	THE ECONOMICS OF STRATEGY	
or EBGN360	INTRODUCTION TO ENTREPRENEURSHIP	
Two Additional Business Core/Application Courses		
EBGN301	INTERMEDIATE MICROECONOMICS	3.0
or EBGN201	PRINCIPLES OF ECONOMICS	
or EBGN280	INTRODUCTION TO BUSINESS ANALYTICS	
or EBGN302	INTERMEDIATE MACROECONOMICS	
or EBGN305	SURVEY OF ACCOUNTING	
or EBGN308	PRINCIPLES OF MARKETING	
or EBGN309	FUNDAMENTALS OF MANAGEMENT	
or EBGN315	THE ECONOMICS OF STRATEGY	
or EBGN320	ECONOMICS AND TECHNOLOGY	
or EBGN345	PRINCIPLES OF CORPORATE FINANCE	
or EBGN346	INTRODUCTION TO INVESTMENTS	
or EBGN360	INTRODUCTION TO ENTREPRENEURSHIP	
or EBGN381	PREDICTIVE BUSINESS ANALYTICS	
or EBGN382	PRESCRIPTIVE BUSINESS ANALYTICS	
or EBGN453	PROJECT MANAGEMENT	
or EBGN459	SUPPLY CHAIN ANALYTICS	
or EBGN461	STOCHASTIC MODELS IN MANAGEMENT SCIENCE	
EBGN302	INTERMEDIATE MACROECONOMICS	3.0
or EBGN201	PRINCIPLES OF ECONOMICS	
or EBGN280	INTRODUCTION TO BUSINESS ANALYTICS	
or EBGN301	INTERMEDIATE MICROECONOMICS	
or EBGN305	SURVEY OF ACCOUNTING	
or EBGN308	PRINCIPLES OF MARKETING	
or EBGN309	FUNDAMENTALS OF MANAGEMENT	
or EBGN315	THE ECONOMICS OF STRATEGY	
or EBGN320	ECONOMICS AND TECHNOLOGY	
or EBGN345	PRINCIPLES OF CORPORATE FINANCE	
or EBGN346	INTRODUCTION TO INVESTMENTS	
or EBGN360	INTRODUCTION TO ENTREPRENEURSHIP	
or EBGN381	PREDICTIVE BUSINESS ANALYTICS	
or EBGN382	PRESCRIPTIVE BUSINESS ANALYTICS	
or EBGN453	PROJECT MANAGEMENT	
or EBGN459	SUPPLY CHAIN ANALYTICS	
or EBGN461	STOCHASTIC MODELS IN MANAGEMENT SCIENCE	
Total Semester Hrs		**24.0**

CS + Computer Engineering

Required Courses		
EENG281	ELECTRICAL CIRCUITS (Circuits)	3.0
or EENG282	ELECTRICAL CIRCUITS LABORATORY	
or PHGN215	ANALOG ELECTRONICS	
EENG284	DIGITAL LOGIC (Digital Logic)	4.0
or PHGN317	SEMICONDUCTOR CIRCUITS- DIGITAL	
EENG383	EMBEDDED SYSTEMS	4.0
Information Processing & Security		
CSCI471	COMPUTER NETWORKS I	3.0
or CSCI474	INTRODUCTION TO CRYPTOGRAPHY	
or CSCI475	INFORMATION SECURITY AND PRIVACY	
or CSCI560	FUNDAMENTALS OF COMPUTER NETWORKS	
or CSCI574	THEORY OF CRYPTOGRAPHY	
or CSCI585	INFORMATION SECURITY PRIVACY	
or EENG411	DIGITAL SIGNAL PROCESSING	
or EENG427	WIRELESS COMMUNICATIONS	
Software for Systems		
CSCI403	DATA BASE MANAGEMENT	3.0
or CSCI423	COMPUTER SIMULATION	
or CSCI425	COMPILER DESIGN	
or CSCI440	PARALLEL COMPUTING FOR SCIENTISTS AND ENGINEERS	
or CSCI581	QUANTUM PROGRAMMING	

Hardware for Systems		
CSCI410	ELEMENTS OF COMPUTING SYSTEMS	3.0
or CSCI564	ADVANCED COMPUTER ARCHITECTURE	
or CSCI565	DISTRIBUTED SYSTEMS	
or CSCI580	ADVANCED HIGH PERFORMANCE COMPUTING	
or CSCI582	COMPUTING BEYOND CPU'S	
or EENG350	ELECTRICAL ENGINEERING INTEGRATION AND DESIGN	
or EENG423	INTRODUCTION TO VLSI DESIGN	
or PHGN435	INTERDISCIPLINARY MICROELECTRONICS PROCESSING LABORATORY	
Computer Engineering Elective		
CSCI423	COMPUTER SIMULATION	0-3
or CSCI403	DATA BASE MANAGEMENT	
or CSCI410	ELEMENTS OF COMPUTING SYSTEMS	
or CSCI425	COMPILER DESIGN	
or CSCI437	INTRODUCTION TO COMPUTER VISION	
or CSCI440	PARALLEL COMPUTING FOR SCIENTISTS AND ENGINEERS	
or CSCI507	INTRODUCTION TO COMPUTER VISION	
or CSCI564	ADVANCED COMPUTER ARCHITECTURE	
or CSCI565	DISTRIBUTED SYSTEMS	
or CSCI572	COMPUTER NETWORKS II	
or CSCI580	ADVANCED HIGH PERFORMANCE COMPUTING	
or CSCI581	QUANTUM PROGRAMMING	
or CSCI582	COMPUTING BEYOND CPU'S	
or EENG350	ELECTRICAL ENGINEERING INTEGRATION AND DESIGN	
or EENG423	INTRODUCTION TO VLSI DESIGN	
or EENG484	ADVANCED DIGITAL DESIGN	
or MEGN441	INTRODUCTION TO ROBOTICS	
or PHGN435	INTERDISCIPLINARY MICROELECTRONICS PROCESSING LABORATORY	
Additional Free Elective		
FREE	Free Elective	1.0
Total Semester Hrs		**21-24**

CS + Data Science

Required Data Courses		
CSCI403	DATA BASE MANAGEMENT	3.0
CSCI413	DATA SCIENCE FOR COMPUTER SCIENCE	3.0
CSCI470	INTRODUCTION TO MACHINE LEARNING	3.0
Required Statistic Courses		
MATH201	INTRODUCTION TO STATISTICS	3.0
MATH324	STATISTICAL MODELING	3.0
MATH335	INTRODUCTION TO MATHEMATICAL STATISTICS	3.0
Advanced Statistics		
MATH432	SPATIAL STATISTICS	3.0
or MATH433	TIME SERIES AND ITS APPLICATIONS	
or MATH436	ADVANCED STATISTICAL MODELING	
or MATH437	MULTIVARIATE ANALYSIS	
or MATH438	STOCHASTIC MODELS	
or MATH560	INTRODUCTION TO KEY STATISTICAL LEARNING METHODS I	
Application/Advanced Data Sciece		
CSCI404	ARTIFICIAL INTELLIGENCE	3.0
or CSCI423	COMPUTER SIMULATION	
or CSCI474	INTRODUCTION TO CRYPTOGRAPHY	
or CSCI478	INTRODUCTION TO BIOINFORMATICS	
or CSCI575	ADVANCED MACHINE LEARNING	
Total Semester Hrs		**24.0**

CS + Entrepreneurship & Innovation

Required Entrepreneur Courses		
EBGN320	ECONOMICS AND TECHNOLOGY	3.0
EBGN360	INTRODUCTION TO ENTREPRENEURSHIP	3.0
Design		
CSCI460	SOFTWARE STARTUPS: FROM IDEA TO LAUNCH	3.0
Business		
EBGN201	PRINCIPLES OF ECONOMICS	3.0
or EBGN280	INTRODUCTION TO BUSINESS ANALYTICS	
or EBGN305	SURVEY OF ACCOUNTING	
or EBGN308	PRINCIPLES OF MARKETING	
or EBGN453	PROJECT MANAGEMENT	
or EBGN460	BUSINESS MODEL DEVELOPMENT	
Social & Ethical Impact		
EDNS315	ENGINEERING FOR SOCIAL AND ENVIRONMENTAL RESPONSIBILITY	3.0
or EDNS301	HUMAN-CENTERED PROBLEM DEFINITION	
or EDNS401	PROJECTS FOR PEOPLE	
or EDNS430	CORPORATE SOCIAL RESPONSIBILITY	
or EDNS450	DESIGN FOR THE BUILT ENVIRONMENT	
or EDNS478	ENGINEERING AND SOCIAL JUSTICE	
or EDNS515	INTRODUCTION TO SCIENCE AND TECHNOLOGY STUDIES	
Front End Application		
CSCI422	USER INTERFACES	3.0
or CSCI445	WEB PROGRAMMING	
or CSCI446	WEB APPLICATIONS	
or CSCI448	MOBILE APPLICATION DEVELOPMENT	
Prototyping		
CSCI421	INTRODUCTION TO HUMAN COMPUTER INTERACTION	3.0
or CSCI460	SOFTWARE STARTUPS: FROM IDEA TO LAUNCH	
or EDNS301	HUMAN-CENTERED PROBLEM DEFINITION	
or EDNS401	PROJECTS FOR PEOPLE	
or EDNS445	PRODUCT REDESIGN	
Additional CS Elective		
CSCI ELECT	Computer Science Elective	3.0
Total Semester Hrs		**24.0**

CS + Robotics & Intelligent Systems

Required Robotics Courses		
CSCI404	ARTIFICIAL INTELLIGENCE	3.0

EENG307	INTRODUCTION TO FEEDBACK CONTROL SYSTEMS	3.0
MEGN441	INTRODUCTION TO ROBOTICS	3.0
Electrical Circuits		
EENG281	ELECTRICAL CIRCUITS	3.0
or EENG282	ELECTRICAL CIRCUITS LABORATORY	
or PHGN215	ANALOG ELECTRONICS	
Perception		
CSCI437	INTRODUCTION TO COMPUTER VISION	0-3
or CSCI507	INTRODUCTION TO COMPUTER VISION	
or EENG519	ESTIMATION THEORY AND KALMAN FILTERING	
Cognition		
CSCI470	INTRODUCTION TO MACHINE LEARNING	3.0
or CSCI534	ROBOT PLANNING AND MANIPULATION	
Interaction and Society		
CSCI432	ROBOT ETHICS	3.0
or CSCI421	INTRODUCTION TO HUMAN COMPUTER INTERACTION	
or CSCI436	HUMAN-ROBOT INTERACTION	
Additional Perception, Cognition, Interaction and Society Course		
CSCI421	INTRODUCTION TO HUMAN COMPUTER INTERACTION	3.0
or CSCI432	ROBOT ETHICS	
or CSCI436	HUMAN-ROBOT INTERACTION	
or CSCI437	INTRODUCTION TO COMPUTER VISION	
or CSCI507	INTRODUCTION TO COMPUTER VISION	
or CSCI534	ROBOT PLANNING AND MANIPULATION	
or EENG519	ESTIMATION THEORY AND KALMAN FILTERING	
or MEGN544	ROBOT MECHANICS: KINEMATICS, DYNAMICS, AND CONTROL	
Total Semester Hrs		**21-24**

CS + Space

Electrical Circuits		
EENG281	ELECTRICAL CIRCUITS	3.0
or EENG282	ELECTRICAL CIRCUITS LABORATORY	
or PHGN215	ANALOG ELECTRONICS	
Required Space Courses		
MEGN452	INTRO TO SPACE EXPLORATION AND RESOURCES	3.0
MEGN455	AEROSPACE SYSTEMS ENGINEERING	3.0
MEGN456	SPACE OPERATIONS AND MISSION DESIGN	3.0
Logic		
CSCI410	ELEMENTS OF COMPUTING SYSTEMS	3.0
or EENG284	DIGITAL LOGIC	
Control		
EENG307	INTRODUCTION TO FEEDBACK CONTROL SYSTEMS	3.0
or EENG383	EMBEDDED SYSTEMS	
Data		
CSCI404	ARTIFICIAL INTELLIGENCE	3.0
or CSCI413	DATA SCIENCE FOR COMPUTER SCIENCE	
or CSCI475	INFORMATION SECURITY AND PRIVACY	
Simulation		
CSCI423	COMPUTER SIMULATION	3.0
or CSCI440	PARALLEL COMPUTING FOR SCIENTISTS AND ENGINEERS	
or CSCI471	COMPUTER NETWORKS I	
Total Semester Hrs		**24.0**

Major GPA

During the 2016-2017 academic year, the Undergraduate Council considered the policy concerning required major GPAs and which courses are included in each degree's GPA. While the GPA policy has not been officially updated, in order to provide transparency, council members agreed that publishing the courses included in each degree's GPA is beneficial to students.

The following list details the courses that are included in the GPA for this degree:

- CSCI200 through CSCI799 inclusive, excluding CSCI*99

COURSES

CSCI1XX. COMPUTER SCIENCE ELECTIVE. 1-6 Semester Hr.

CSCI101. INTRODUCTION TO COMPUTER SCIENCE. 3.0 Semester Hrs.
Introduction to Computer Science is a 3-credit hour **breadth**
CS course. We cover several topics in this course to help students understand how computers work, e.g., binary numbers, Boolean logic and gates, circuit design, machine language, computer hardware, assembly, operating systems, networking, the Internet protocols, cybersecurity, data science, machine learning, and robotics.
Course Learning Outcomes

- 1. Demonstrate how data is represented in computers
- 2. Define the building blocks and organization of computer hardware
- 3. Describe the differences between machine, assembly, and high-level languages
- 4. Design an efficient algorithm and analyze its time/space complexity
- 5. Implement algorithms that solve problems
- 6. Explain how the Internet works
- 7. Detail how an operating system manages processes
- 8. Evaluate whether one is keeping data private
- 9. Categorize the different threats to computer systems
- 10. Perform the data science process
- 11. Explain the Turing test and how one defines intelligence
- 12. Categorize the different types of machine learning algorithms
- 13. Describe key computer ethics topics
- 14. Recognize the impact of computing on the world
- 15. Design and program Python applications,
- 16. Write loops and decision statements in Python,
- 17. Use lists, tuples, sets, dictionaries, and strings in Python,
- 18. Define the structure/components of a Python program,
- 19. Write functions and pass arguments in Python,
- 20. Write code that reads/writes files in Python,

- 21. Design object-oriented programs with Python classes, and
- 22. Read (trace) basic Python code.

CSCI102. INTRODUCTION TO COMPUTER SCIENCE - LAB. 1.0 Semester Hr.
CSCI 102 is our Introduction to Computer Science LAB course. CSCI 102 is a 1-credit hour programming course in Python that is (A) extremely valuable for those who have never programmed and (B) required for some majors (e.g., MechE). While CSCI 102 is not required for some majors, students with little (or no) prior programming experience are strongly encouraged to enroll.
Course Learning Outcomes

- Design and program Python applications
- Write loops and decision statements in Python
- Use lists, tuples, sets, dictionaries, and strings in Python
- Define the structure/components of a Python program
- Write functions and pass arguments in Python
- Write code that reads/writes files in Python
- Design object-oriented programs with Python classes
- Read (trace) basic Python code

CSCI128. COMPUTER SCIENCE FOR STEM. 3.0 Semester Hrs.
Introduction to programming. Intended for students with no prior experience. Teaches basic programming constructs including data types, conditionals, loops, file I/O, functions, and objects in Python 3. Also covers topics vital to STEM computing, such as data science, best practices for code development, and software ethics. Prerequisite: None Co-requisite: None.
Course Learning Outcomes

- Analyze a simple empirical problem, break it down into smaller more manageable components, and design algorithmic solutions to these subproblems
- Implement an existing prompt, plan, or design into programmatically correct Python code that produces the expected text, file, or graph output
- Communicate in the language of programming with a computer and other programmers through code reading, writing, and critique
- Critically discuss and reflect on the role technology has in modern society, and the positive and negative impacts their software may have on future users
- Model how basic numeric and non-numeric data is represented in a computer; identify when, why, and how these physical representations diverge from their conceptual equivalents
- Navigate and utilize a computer file system through a GUI, the text console, and code
- Demonstrate effective debugging practices in an IDE to find, characterize, and correct code errors

CSCI195. CS@MINES BRIDGE SEMINAR COURSE. 1.0 Semester Hr.
The purpose of this course is to support Bridge students for success in CS@Mines. Through this course, students will: 1. Engage in activities that show how computing changes the world and impacts daily lives, 2. Delve into a some of the foundational computer science topics (e.g., Binary Numbers, Networking, Operating Systems, Cybersecurity, Cyber Physical Systems, Artificial Intelligence, Machine Learning, Data Science, Bioinformatics, Robotics), and 3. Explore different career paths in the computer science industry. Most importantly, this course will offer students the opportunity to build relationships with their cohort and their program advisors (e.g., Bridge Program Director, Graduate Program Manger), as well as be a source of strength for each other. Prerequisite: Current CS@Mines Bridge Student .
Course Learning Outcomes

- 1. Describe a breadth of foundational computer science topics and how they relate to emphasis areas in computing.
- 2. Implement networking and relationship building techniques within their cohort and other like-minded students.
- 3. Explain the importance of personal development tools and preparedness for careers in computing (e.g., imposter syndrome, the elevator pitch, resume building, navigating career day).
- 4. Summarize the different types of careers that exist in the computer science field.
- 5. Analyze the specific fields in computer science that are of most interest for deeper exploration and potential degree focus area.

CSCI198. SPECIAL TOPICS. 1-6 Semester Hr.
(I, II) Pilot course or special topics course. Topics chosen from special interests of instructor(s) and student(s). Usually the course is offered only once. Prerequisite: none. Variable credit; 1 to 6 credit hours. Repeatable for credit under different titles.

CSCI199. INDEPENDENT STUDY. 1-6 Semester Hr.
(I, II) Individual research or special problem projects supervised by a faculty member, when a student and instructor agree on a subject matter, content, and credit hours. Prerequisite: "Independent Study" form must be completed and submitted to the Registrar. Variable credit; 1 to 6 credit hours. Repeatable for credit.

CSCI200. FOUNDATIONAL PROGRAMMING CONCEPTS & DESIGN. 3.0 Semester Hrs.
This course teaches students C++, how to manage memory properly & efficiently at run time, the principles of object-oriented programming, and how to create an algorithm using data structures & abstraction to solve a problem. Recursive data structures & algorithms will be constructed & analyzed when solving problems. Initial principal components of software engineering and design will be introduced and used when drafting a solution to a problem. Programs will be developed using a command line interface. Prerequisite: CSCI101 or CSCI128.
Course Learning Outcomes

- L1. Design an algorithm to solve a problem by breaking the overarching problem into smaller modular components using abstraction and object-oriented design with inheritance.
- L2. Translate the algorithm into a program using proper C++ syntax and fundamental programming constructs (e.g. control structures, I/O, classes).
- L3. Recite & apply frequently used Linux command line commands and compile a program using a command line build system.
- L4. Diagram memory usage, dynamically allocate & deallocate objects at run-time using "The Big Three," and trace the call stack of a program’s run-time.
- L5. Define recursion and construct common recursive data structures (e.g. linked list, stack, queue) & algorithms (e.g. search & sort).
- L6. Diagram & construct dynamically allocated data structures (e.g. array, vector, string), recursive data structures, (e.g. linked list, stack, queue), and implement common list operations (e.g. traversal, insertion, removal)

- L7. Define "Big-O" notation, list complexities in increasing order, and analyze an algorithm to compute its run-time performance & memory complexities

CSCI210. SYSTEMS PROGRAMMING. 3.0 Semester Hrs.
The Systems Programming course will teach students how to become proficient with using a Linux operating system from the command line and programming Linux systems. Topics will include: shell scripts; compiling and linking programs; redirecting input and output; controlling jobs from the command line; using advanced SSH functions such as port forwarding and dynamic proxying; file system hierarchy;, package management; kernel compilation; network management; C programming with dynamic memory management, function pointers, c-style polymorphism, recursive functions, and data structures; learning how to use a code repository maintenance tool, such as git, from the command line effectively; security, privacy, and encryption concepts; inter-process communication and client-server architectures. Prerequisite: CSCI200.
Course Learning Outcomes

- 1. Identify, select, and apply appropriate Linux commands for file, process, and network management.
- 2. Utilize the command line by designing and creating shell scripts; compiling and linking programs; redirecting input and output of processes; and executing commands for controlling jobs/processes.
- 3. Explain the purpose of and apply advanced SSH functions such as port forwarding, dynamic proxying, effectively.
- 4. Administer a Linux system applying knowledge of the file system hierarchy, package management, kernel compilation, and network management.
- 5. Design and write C programs composed of dynamic memory management, function pointers, c-style polymorphism, recursive functions, and data structures.
- 6. Examine the capabilities of a code repository maintenance tool, such as git, through command line version control.
- 7. Select and implement appropriate security, privacy, and encryption protocols at the operating system level.
- 8. Design and implement programs with client-server architecture using inter-process communication.
- 9. Discuss and apply appropriate time and project management strategies through the development of multiple substantial programming projects throughout the semester.

CSCI220. DATA STRUCTURES AND ALGORITHMS. 3.0 Semester Hrs.
This course teaches students the design and construction of data structures such as hash tables, trees, heaps, and graphs, analysis of operations on data structures, and design and analysis of algorithms on data structures such as graph search and minimum spanning tree algorithms. Applications of data structures and algorithms on them are discussed in the context of computer systems. Students will further refine programming skills in C++ by producing software implementations of selected data structures and algorithms. Prerequisite: CSCI200 with a C- or better, CSCI358 or MATH334.
Course Learning Outcomes

- Apply algorithmic analysis methods including solving recurrence relations, the “unifying theorem,” and amortized analysis to compute best case, worst case, and average case run-time complexity of algorithms and operations on data structures; and discuss trade-offs between time and space complexity.
- Describe hash tables and write software implementing a hash table; explain and evaluate collision handling methods in the context of storage hierarchies; and assess the relative merits of different hash functions for data types such as strings and integers.
- Describe search trees and their applications and write software implementing a search tree; explain and contrast trees such as red-black or AVL trees, splay trees, b-trees, and radix tries in application contexts such as operating systems, network routing, and databases.
- Describe heaps and priority queues and write software implementing a heap; apply priority queues to problem domains such as data compression and operating systems.
- Define unweighted, weighted, undirected, and directed graphs and explain data structures for storing graphs; explain and analyze algorithms for graph discovery and classification.
- Explain and analyze algorithms on graphs for finding shortest paths and minimum spanning trees; write software implementing graph algorithms; discuss applications of graph algorithms.
- Demonstrate professional conduct with respect to communication and time management.

CSCI250. PYTHON-BASED COMPUTING: BUILDING A SENSOR SYSTEM. 3.0 Semester Hrs.
This course will teach students the skills needed for data collection, analysis, and visualization on a small embedded device (e.g., Raspberry Pi). Students will learn basic Linux, Python, and the programming skills needed to control the hardware and associated sensors. This hands-on course includes a baseline project, four introductory projects (e.g., acoustic, acceleration, magnetic field, optical), and a final Capstone project. The Capstone project will have students create their own application using the techniques learned during the first half of the semester; students will then present their Capstone project through a formal presentation, write-up, and demonstration. We suggest the student take "Introduction to Computer Science" before this course. Prerequisite: CSCI128. Co-requisite: MATH213, PHGN200.
Course Learning Outcomes

- 1. Create, navigate, and manage files and directory structures using basic Linux shell commands.
- 2. Describe the functionality and purpose of the individual components of the Raspberry Pi Hardware.
- 3. Install the Raspian operating system onto the Raspberry Pi Hardware and setup basic configuration parameters.
- 4. Download, install, and develop programs using the Spyder Integrated Development Environment (IDE) on the Raspberry Pi Hardware.
- 5. Develop and run basic Python functions and programs in the Linux environment to collect data from sensors using the Raspberry Pi Hardware (e.g., acoustic, acceleration, magnetic field, optical).
- 6. Plot and analyze data from the sensor system and compare to mathematical models.

CSCI260. FORTRAN PROGRAMMING. 2.0 Semester Hrs.
(I) Computer programming in Fortran90/95 with applications to science and engineering. Program design and structure, problem analysis, debugging, program testing. Language skills: arithmetic, input/output, branching and looping, functions, arrays, data types. Introduction to operating systems. Prerequisite: none. 2 hours lecture; 2 semester hours.

CSCI261. PROGRAMMING CONCEPTS. 3.0 Semester Hrs.
This course introduces fundamental computer programming concepts using a high-level language and a modern development environment. Programming skills include sequential, selection, and repetition control structures, functions, input and output, primitive data types, basic data structures including arrays and pointers, objects, and classes. Software engineering skills include problem solving, program design, and debugging practices. Prerequisite: CSCI101.
Course Learning Outcomes

- unchanged

CSCI262. DATA STRUCTURES. 3.0 Semester Hrs.
Defining and using data structures such as linked lists, stacks, queues, binary trees, binary heap, and hash tables. Introduction to algorithm analysis, with emphasis on sorting and search routines. Language skills: abstract data types, templates, and inheritance. 3 hours lecture; 3 semester hours. Prerequisite: CSCI261 with a grade of C- or higher or CSCI200 with a grade of C- or higher.
Course Learning Outcomes

- unchanged

CSCI274. INTRODUCTION TO THE LINUX OPERATING SYSTEM. 1.0 Semester Hr.
Introduction to the Linux Operating System will teach students how to become proficient with using a Linux operating system from the command line. Topics will include: remote login (ssh), file system navigation, file commands, editors, compilation, execution, redirection, output, searching, processes, usage, permissions, compression, parsing, networking, and bash scripting. Prerequisite: CSCI200 or CSCI261.
Course Learning Outcomes

- unchanged

CSCI290. PROGRAMMING CHALLENGES I. 1.0 Semester Hr.
This course is the first of three courses in the Programming Challenges sequence which covers problem solving patterns and paradigms found in technical interviews and programming competitions. The students will learn more advanced data structures and algorithms while focusing on algorithmic complexity to solve problems in a finite amount of time. Co-requisite: CSCI220 or CSCI262.
Course Learning Outcomes

- Solve common programming problems and identify underlying patterns
- Argue and prove correctness of solution
- Determine and argue space and time complexity of solutions
- Combine common algorithms to solve problems
- Improve skills in technical interviews and programming competitions

CSCI295. INDUSTRY EXPLORATION I. 1.0 Semester Hr.
Industry Exploration I provides 1st and 2nd year students an opportunity to explore different career paths in computer science. Each week students meet (over Zoom) with a company that hires a number of computer scientists. Prior to the meeting, students research the company and determine 1-2 specific question(s) to ask during the meeting (i.e., questions specific to the company). During the meeting, students talk with employees at the company to learn more about the types of computer science jobs that exist. After the meeting, students reflect on what was learned during the meeting. At the end of the semester, students have a better understanding of different types of jobs in computer science and what they may want to do in their future careers.
Course Learning Outcomes

- 1. Understand different types of companies that hire CS majors through their research and visits with the companies
- 2, Have answers to questions they have generated, to further their understanding of positions and roles in CS industry
- 3. Improve their professional communication and networking skills through experience
- 4. Shape their future career path by relating what they learn in the course

CSCI298. SPECIAL TOPICS. 1-6 Semester Hr.
(I, II) Pilot course or special topics course. Topics chosen from special interests of instructor(s) and student(s). Usually the course is offered only once. Prerequisite: none. Variable credit; 1 to 6 credit hours. Repeatable for credit under different titles.

CSCI298. SPECIAL TOPICS. 1-6 Semester Hr.

CSCI299. INDEPENDENT STUDY. 1-6 Semester Hr.
(I, II) Individual research or special problem projects supervised by a faculty member, when a student and instructor agree on a subject matter, content, and credit hours. Prerequisite: "Independent Study" form must be completed and submitted to the Registrar. Variable credit; 1 to 6 credit hours. Repeatable for credit.

CSCI303. INTRODUCTION TO DATA SCIENCE. 3.0 Semester Hrs.
This course will teach students the core skills needed for gathering, cleaning, organizing, analyzing, interpreting, and visualizing data. Students will learn basic SQL for working with databases, basic Python programming for data manipulation, and the use and application of statistical and machine learning toolkits for data analysis. The course will be primarily focused on applications, with an emphasis on working with real (non-synthetic) datasets. Prerequisite: CSCI101 or CSCI102 or CSCI128.
Course Learning Outcomes

- 1. Acquire, clean, and organize data from a variety of sources, including raw data files, SQL databases, and online repositories.
- 2. Run simple SQL queries to manipulate and analyze data in relational databases and other data stores implementing SQL front ends.
- 3. Apply toolkits to preprocess large datasets for input to statistical and machine learning algorithms, including methods of feature extraction and dimensionality reduction.
- 4. Apply statistical and machine learning toolkits to large datasets, including applications of regression, classification, and clustering.
- 5. Perform simple visualizations of data.
- 6. Recognize and address the ethical issues arising from data collection and statistical and machine learning.

CSCI306. SOFTWARE ENGINEERING. 3.0 Semester Hrs.
Introduction to software engineering processes and object-oriented design principles. Topics include the Agile development methodology, test-driven development, UML diagrams, use cases and several object-oriented design patterns. Course work emphasizes good programming practices via version control and code reviews. Prerequisite: CSCI210 with grade of C- or higher, CSCI220 with grade of C- or higher or CSCI262 with grade of C- or higher.
Course Learning Outcomes

- Articulate the nature and purpose of Software Engineering along with the three major goals of Software Engineering.
- Demonstrate ability to quickly learn unfamiliar programming languages and characterize common traits of OOP across languages.
- Describe and utilize various agile and extreme programming methodologies.
- Design complex soft ware architecture using UML.
- Design and develop quality soft ware using specific methodologies including TDD, Customer Story Requirements, Coding Standards, Code Review, Pair Programming, and Refactoring.
- Analyze OOP architecture within the context of Software Principles and Software Design Patterns to effectively strengthen software quality.
- Demonstrate effective use of various version control/configuration management tools in a development environment.
- Effectively analyze and debug software using various debugging techniques, tools, and methodologies.
- Function effectively in a software development team environment.

CSCI340. COOPERATIVE EDUCATION. 0-3 Semester Hr.
(I, II, S) (WI) Supervised, full-time engineering-related employment for a continuous six-month period (or its equivalent) in which specific educational objectives are achieved. Prerequisite: Second semester sophomore status and a cumulative grade point average of at least 2.00. 0 to 3 semester hours. Cooperative Education credit does not count toward graduation except under special conditions. Repeatable.

CSCI341. COMPUTER ORGANIZATION. 3.0 Semester Hrs.
Covers the basic concepts of computer architecture and organization. Topics include machine level instructions and operating system calls used to write programs in assembly language, computer arithmetics, performance, processor design, and pipelining techniques. This course provides insight into the way computers operate at the machine level. Prerequisite: CSCI210 OR CSCI262.
Course Learning Outcomes

- Program in RISC-V assembly language
- Translate between C programs and RISC-V assembly code
- Translate between RISC-V assembly code and RISC-V machine code
- Represent a number using IEEE 754 floating point standard
- Describe how a CPU performs addition, subtraction, multiplication, and division
- Evaluate CPU performance
- Design a single-cycle processor and its control
- Design pipelining techniques to speed up a single-cycle processor
- Elaborate how memory hierarchy works

CSCI358. DISCRETE MATHEMATICS. 3.0 Semester Hrs.
This course is an introductory course in discrete mathematics and algebraic structures. Topics include: formal logic; proofs, recursion, analysis of algorithms; sets and combinatorics; relations, functions, and matrices; Boolean algebra and computer logic; trees, graphs, finite-state machines, and regular languages. Prerequisite: MATH112 or MATH122.
Course Learning Outcomes

- Think logically and mathematically.
- Understand mathematical reasoning.
- Perform combinatorial analysis.
- Represent discrete objects and relations using discrete structures.
- Model problems in diverse areas with discrete math.

CSCI370. ADVANCED SOFTWARE ENGINEERING. 5.0 Semester Hrs.
This capstone course has three primary goals: (1) to enable students to apply their course work knowledge to a challenging applied problem for a real client, (2) to enhance students' verbal and written communication skills, and (3) to provide an introduction to ethical decision making in computer science. Ethics and communication skills are emphasized in a classroom setting. The client work is done in small teams, either on campus or at the client site. Faculty advisors provide guidance related to the software engineering process, which is similar to Scrum. By the end of the course students must have a finished product with appropriate documentation. Prerequisite: CSCI306.
Course Learning Outcomes

- unchanged

CSCI390. PROGRAMMING CHALLENGES II. 1.0 Semester Hr.
This course is the second of three courses in the Programming Challenges sequence which covers problem solving patterns and paradigms found in technical interviews and programming competitions. The students will learn more advanced set, counting, & number theory and algorithms while focusing on algorithmic complexity to solve problems in a finite amount of time. Prerequisite: CSCI290. Co-requisite: CSCI358.
Course Learning Outcomes

- Solve common programming problems and identify underlying patterns
- Argue and prove correctness of solution
- Determine and argue space and time complexity of solutions
- Combine common algorithms to solve problems
- Improve skills in technical interviews and programming competitions

CSCI395. INDUSTRY EXPLORATION II. 1.0 Semester Hr.
Industry Exploration II provides 3rd and 4th year students an opportunity to explore different career paths in computer science. Each week students visit a company that hires a number of computer scientists at the company’s office. Prior to the visit, students research the company and determine 1-2 specific question(s) to ask during the meeting (i.e., questions specific to the company). During the visit, students are provided a tour of the company’s office and the opportunity to talk to employees and learn more about the types of computer science jobs that exist. After the visit, students reflect on what was learned during the visit. At the end of the semester, students have a better understanding of different types of jobs in computer science and what they may want to do in their future careers.
Course Learning Outcomes

- 1. Understand different types of companies that hire CS majors through their research and visits with the companies
- 2. Have answers to questions they have generated, to further their understanding of positions and roles in CS industry
- 3. Improve their professional communication and networking skills through experience
- 4. Discover different company cultures / work environments
- 5. Shape their future career path by relating what they learn in the course

CSCI398. SPECIAL TOPICS. 1-6 Semester Hr.
(I, II) Pilot course or special topics course. Topics chosen from special interests of instructor(s) and student(s). Usually the course is offered only once. Prerequisite: none. Variable credit; 1 to 6 credit hours. Repeatable for credit under different titles.

CSCI398. SPECIAL TOPICS. 0-6 Semester Hr.

CSCI399. INDEPENDENT STUDY. 1-6 Semester Hr.
(I, II) Individual research or special problem projects supervised by a faculty member, when a student and instructor agree on a subject matter, content, and credit hours. Prerequisite: "Independent Study" form must be completed and submitted to the Registrar. Variable credit; 1 to 6 credit hours. Repeatable for credit.

CSCI399. INDEPENDENT STUDY. 1-6 Semester Hr.

CSCI399. INDEPENDENT STUDY. 1-6 Semester Hr.

CSCI400. PRINCIPLES OF PROGRAMMING LANGUAGES. 3.0 Semester Hrs.
This course takes a broad view of programming languages, focusing on the fundamental abstractions and principles of language design that transcend the specifics of any particular programming language. The course will emphasize functional programming, develop experience via programming projects, and cover topics such as lambda calculus, higher-order functions, induction, persistence, type systems, syntax, and evaluation. Ultimately, students will have an opportunity to improve programming skills, and develop a deeper understanding of how programming languages are designed and implemented. Prerequisite: CSCI306 AND (CSCI358 or MATH300).
Course Learning Outcomes

- Design and implement program in a functional style in a functional language
- Implement the core of a programming language
- Explain programming language theory terms including lexical closure, lexing, parsing, operational semantics, lexical environment, type checking, type environment, type inference, and references;
- Relate common programming constructs and idioms to the lambda calculus
- Reason inductively about recursive functions and data structures;
- Analyze, compare, and contrast the suitability of mutable/ephemeral/imperative vs. immutable/persistent/functional programming approaches and data structures for new problems

CSCI403. DATA BASE MANAGEMENT. 3.0 Semester Hrs.
Design and evaluation of information storage and retrieval systems, including defining and building a database and producing the necessary queries for access to the stored information. Relational database management systems, structured query language, and data storage facilities. Applications of data structures such as lists, inverted lists and trees. System security, maintenance, recovery and definition. Interfacing host languages to database systems and object-relational mapping tools. NoSQL databases and distributed databases. Prerequisite: CSCI200 with a grade of C- or higher.
Course Learning Outcomes

- (Data Modeling and Database Design) Identify data elements and relationships between data elements required by an application. Design and implement an appropriate data representation for data and relationships using correct application of types, constraints, and keys. Justify the design decisions made in the data model. Illustrate the data model using Entity-Relation Diagrams.
- (Queries) Construct relational queries that produce precise and complete answers to questions about the data. Compose queries using advanced SQL features including Common Table Expressions, window functions, and User-Defined Functions. Evaluate correctness and efficiency of queries and evaluate tradeoffs between queries using different techniques such as joins, subqueries, and different methods of aggregation.
- (Performance Analysis and Tuning) Analyze database performance and tune the design and configuration of the database to improve query and insert performance. Propose, implement, and justify performance tuning decisions using indexes, materialized views, and other techniques.
- (Transactional data) Construct data modification statements on a relational database. Design transaction-based data updates for applications and select appropriate locking strategies under different application scenarios.
- (Normalization) Analyze data dependencies, formulate functional dependencies, and normalize a relational database schema. Explain the benefits of normalization in the context of transactional data applications. Assess tradeoffs and benefits of denormalization in different contexts.
- (Software engineering) Design and implement a software application which uses a relational database to manage data. Justify the design decisions made in the application, including when to process data through SQL and when to process data in the application. Identify security risks in application code and apply security best practices to prevent common security problems.
- (Non-relational models) Explain how NoSQL, Key-Value, Document, and Graph databases differ from relational databases and when to use them for different applications. Evaluate tradeoffs between different data representations.
- (Data engineering) Identify data cleaning, inspection, and other pre-processing steps for noisy structured and semi-structured datasets, and apply these steps within a relational database system. Demonstrate bulk data loading and extract-transform-load within a relational database system.

CSCI404. ARTIFICIAL INTELLIGENCE. 3.0 Semester Hrs.
General investigation of the Artificial Intelligence field. Several methods used in artificial intelligence such as search strategies, knowledge representation, logic and probabilistic reasoning are developed and applied to practical problems. Fundamental artificial intelligence techniques are presented, including neural networks, genetic algorithms, and fuzzy sets. Selected application areas, such as robotics, natural language processing and games, are discussed. Prerequisite: (CSCI220 with a grade of C- or higher or CSCI262 with a grade of C- or higher) and (MATH201 or MATH334).
Course Learning Outcomes

- Identify problems where artificial intelligence techniques are applicable.
- Apply selected basic AI techniques, judge applicability of more advanced techniques.

CSCI406. ALGORITHMS. 3.0 Semester Hrs.
Equivalent with MATH406,
Reasoning about algorithm correctness (proofs, counterexamples). Analysis of algorithms: asymptotic and practical complexity. Review of dictionary data structures (including balanced search trees). Priority queues. Advanced sorting algorithms (heapsort, radix sort). Advanced algorithmic concepts illustrated through sorting (randomized algorithms,

lower bounds, divide and conquer). Dynamic programming. Backtracking. Algorithms on unweighted graphs (traversals) and weighted graphs (minimum spanning trees, shortest paths, network flows and bipartite matching); NP-completeness and its consequences. Prerequisite: CSCI220 with a grade of C- or higher or CSCI262 with a grade of C- or higher, MATH213 or MATH223, MATH300 or MATH358 or CSCI358.
Course Learning Outcomes

- Reasoning about algorithm correctness (proofs, counterexamples)
- Analysis of algorithms

CSCI410. ELEMENTS OF COMPUTING SYSTEMS. 3.0 Semester Hrs.
This comprehensive course will help students consolidate their understanding of all fundamental computer science concepts. Topics include symbolic communication, Boolean logic, binary systems, logic gates, computer architecture, assembly language, assembler construction, virtual machines, object-oriented programming languages, software engineering, compilers, language design, and operating systems. Using a hardware simulator and a programming language of their choice, students construct an entire modern computer from the ground up, resulting in an intimate understanding of how each component works. Prerequisites: CSCI341 or EENG383. 3 lecture hours, 3 credit hours.

CSCI413. DATA SCIENCE FOR COMPUTER SCIENCE. 3.0 Semester Hrs.
This course will teach students the core skills needed for gathering, cleaning, organizing, analyzing, interpreting, and visualizing data. Students will use the python programming language and related toolkits for data manipulation and the use and application of statistical and machine learning for data analysis. The course will be primarily focused on applications, with an emphasis on working with real (non-synthetic) datasets. Students will propose and design a semester project using a dataset from their domain of interest, leveraging the concepts and skills acquired from this course (e.g., data analysis, ethical considerations, evaluation and synthesis of results, storytelling and visualization). Prerequisites: CSCI 200 with a grade of C- or higher and (MATH 201 OR MATH 334).
Course Learning Outcomes

- 1. Conduct data acquisition using a varied set of techniques with structured and unstructured datasets; including raw data files, SQL databases, online repositories, and programmatically through web scraping and APIs.
- 2. Apply preprocessing strategies to complex and dynamic datasets using industry-standard toolkits and machine learning algorithms to extract features, reduce dimensionality, remove errors, inconsistencies, and missing values.
- 3. Differentiate between machine learning approaches such as classification, regression, clustering, and neural networks for predictive analytics and pattern recognition.
- 4. Evaluate the predictive power of the different statistical and machine learning methods to solve real-world data science problems.
- 5. Develop storytelling and visualization techniques to effectively communicate (exploratory) or persuade (explanatory) findings to a specific audience.
- 6. Critically assess ethical considerations and challenges related to data collection and analysis.
- 7. Construct a comprehensive data science project from inception to presentation, integrating the various techniques and tools learned throughout the course.

CSCI421. INTRODUCTION TO HUMAN COMPUTER INTERACTION. 3.0 Semester Hrs.
This course provides an introduction to human-computer interaction and the design process, framed through interface design. Students will learn methods and skills for designing and prototyping interactive systems. The course covers the design process from the initial formulation of a design problem to creation of digital prototypes. The course structure is a mix of lectures, classroom activities, and design critiques by peers and the instructor. The course is overwhelmingly organized around a group project. There are no programming assignments in the course. Prior programming experience is optional, but willingness to socialize and work as a group is necessary. Prerequisite: CSCI220.
Course Learning Outcomes

- 1. Ideate and propose interaction design problems
- 2. Explain and apply core theories and models from the field of HCI
- 3. Create task-focused scenarios to guide design efforts, and use sketches and storyboards to communicate those scenarios
- 4. Improve existing designs through rapid prototyping and iteration
- 5. Communicate a design, and justify design decisions, through writing and spoken presentations
- 6. Read, discuss and critique research in the field of HCI

CSCI422. USER INTERFACES. 3.0 Semester Hrs.
User Interface Design is a course for programmers who want to learn how to create more effective software. This objective will be achieved by studying principles and patterns of interaction design, critiquing existing software using criteria presented in the textbooks, and applying criteria to the design and implementation of one larger product. Students will also learn a variety of techniques to guide the software design process, including Cognitive Walkthrough, Talk-aloud and others. Prerequisite: CSCI262. 3 hours lecture; 3 semester hours.

CSCI423. COMPUTER SIMULATION. 3.0 Semester Hrs.
A first course in computer simulation using formal learning groups and emphasizing the rigorous development of simulation applications. Topics will include random number generation, Monte Carlo simulation, discrete event simulation, and the mathematics behind their proper implementation and analysis (random variates, arrival time modeling, infinite horizon statistics, batch means and sampling techniques). The course uses learning group assignments, quizzes, programming projects (using Linux) and exams for assessment. Prerequisite: (CSCI210 or CSCI274) AND CSCI306 AND (MATH201 or MATH334).
Course Learning Outcomes

- Simulation software design and implementation.
- Individual "Student Bounties" in simulation, results of journaled communications.
- Design and develop software applications, applying effective software engineering practices and algorithmic techniques to maximize efficiency.
- Select or develop appropriate Abstract Data Types (ADTs) to solve realistic problems. Analyze the complexity of algorithms used in problem solutions.
- Display effective professional and interpersonal skills such as communication, teamwork, and knowledge of ethical issues specific to computing.

CSCI425. COMPILER DESIGN. 3.0 Semester Hrs.
An introductory course to the design and construction of compilers. Topics include scanning (lexical analysis), context free grammars,

recursive descent (top-down) parsing, LR (bottom up) parsing, syntax directed translation, syntax trees, expression trees, parse trees, intermediate representation, register allocation and target code generation. Students will construct their own tool chain for compiling a simple language, tracking the relevant course topics as they are covered. Prerequisite: CSCI306. Corequisite: CSCI341.

Course Learning Outcomes

- 1. Understand the steps taken by most compilers to translate a program listing to either another language or target code for a machine.
- 2. Understand and be able to apply the theory of lexical analysis, regular expressions, deterministic finite automata (DFAs), non-deterministic finite automata (NFAs), and how to use scanner generators (eg flex).
- 3. Know the fundamental principles of context free grammars as well as their associated attributes and properties.
- 4. Understand the constructs for expressing operator precedence and associativity in context free grammars, as well as how language ambiguities are detected and worked around by modern compiler tools.
- 5. Know the theory and implementation of top-down and bottom up parsing, the generation of parse trees, and their translation to abstract syntax trees (ASTs).
- 6. Know program statements are translated to machine language at the CPU op-code and register level.

CSCI432. ROBOT ETHICS. 3.0 Semester Hrs.

This course explores ethical issues arising in robotics and human-robot interaction through philosophical analysis, behavioral and psychological analysis, research ethics education, and the integration of social and ethical concerns in scientific experimentation and algorithm design. Topics include case studies in lethal autonomous weapon systems, autonomous cars, and social robots, as well as higher-level concerns including economics, law, policy, and discrimination. Prerequisite: CSCI200 and MATH201.

Course Learning Outcomes

- 1. Understand the basic ethical theories, concepts, tools, and frameworks for analyzing the social and ethical ramifications of robotics
- 2. Be able to critically examine the ethical significance of the use of robotics in daily and technical fields including human-robot interaction, medicine, relationship, military, etc.
- 3. Develop a critical attitude toward the role of robotics in shaping human society including human perceptions and behaviors
- 4. Be able to use the theories, concepts, tools, and frameworks learned from this class to critically examine emerging robot ethics issues in the society.
- 4. Understand the tradeoffs underlying the design of autonomous moral agents

CSCI436. HUMAN-ROBOT INTERACTION. 0-3 Semester Hr.

Human-Robot Interaction is an interdisciplinary field at the intersection of Computer Science, Robotics, Psychology, and Human Factors, that seeks to answer a broad set of questions about robots designed to interact with humans (e.g., assistive robots, educational robots, and service robots), such as: (1) How does human interaction with robots differ from interaction with other people? (2) How does the appearance and behavior of a robot change how humans perceive, trust, and interact with that robot? And (3) How can we design and program robots that are natural, trustworthy, and effective? Accordingly, In this course, students will learn (1) how to design interactive robots, (2) the algorithmic foundations of interactive robots; and (3) how to evaluate interactive robots. To achieve these learning objectives, students will read and present key papers from the HRI literature, and complete a final project in which they will design, pilot, and evaluate novel HRI experiments in small groups, with in-class time expected to be split between lecture by the instructor, presentations by students, and either collaborative active learning activities or discussions with researchers in the field. Prerequisite: CSCI200 or CSCI262 and MATH201 or MATH334.

Course Learning Outcomes

- 1. Understand the theoretical foundations and critical application domains of human-robot interaction.
- 2. Employ design techniques to design interactive robots.
- 3. Design human-subject experiments to evaluate interactive robots.
- 4. Perform qualitative and quantitative analysis on the results of human-robot interaction experiments.

CSCI437. INTRODUCTION TO COMPUTER VISION. 3.0 Semester Hrs.

Equivalent with CSCI512,EENG507,EENG512,

Computer vision is the process of using computers to acquire images, transform images, and extract symbolic descriptions from images. This course provides an introduction to this field, covering topics in image formation, feature extraction, location estimation, and object recognition. Design ability and hands-on projects will be emphasized, using popular software tools. The course will be of interest both to those who want to learn more about the subject and to those who just want to use computer imaging techniques. Must be Senior level standing. 3 hours lecture; 3 semester hours. Prerequisite: (MATH201 or MATH334 or EENG311) and MATH332 and (CSCI200 or CSCI261).

Course Learning Outcomes

- Be able to analyze and predict the behavior of image formation, transformation, and recognition algorithms.
- Be able to design, develop, and evaluate algorithms for specific computer vision applications.
- Be able to use software tools (such as OpenCV) to implement computer vision algorithms.
- Develop an understanding of classical (hand-engineered) and modern (deep-learning-based) computer vision algorithms.
- Learn key algorithms in 3 major areas of image filtering, camera geometry, and machine learning.

CSCI440. PARALLEL COMPUTING FOR SCIENTISTS AND ENGINEERS. 3.0 Semester Hrs.

Equivalent with MATH440,

This course is designed to introduce the field of parallel computing to all scientists and engineers. The students will be taught how to solve scientific problems using parallel computing technologies. They will be introduced to basic terminologies and concepts of parallel computing, learn how to use MPI to develop parallel programs, and study how to design and analyze parallel algorithms. Prerequisite: CSCI220 with a grade of C- or higher or CSCI262 with a grade of C- or higher, CSCI341.

Course Learning Outcomes

- To follow

CSCI441. COMPUTER GRAPHICS. 3.0 Semester Hrs.

Equivalent with MATH441,

This class focuses on the basic 3D rendering and modeling techniques. In particular, it covers the graphics pipeline, elements of global

illumination, modeling techniques based on polynomial curves and patches, and shader programming using the GPU. Prerequisite: CSCI220 with a grade of C- or higher or CSCI262 with a grade of C- or higher, MATH332.

Course Learning Outcomes

- Describe, replicate, & modify the graphical pipeline
- Create interactive, real-time graphics applications

CSCI442. OPERATING SYSTEMS. 3.0 Semester Hrs.
Introduces the essential concepts in the design and implementation of operating systems: what they can do, what they contain, and how they are implemented. Despite rapid OS growth and development, the fundamental concepts learned in this course will endure. We will cover the following high-level OS topics, roughly in this order: computer systems, processes, processor scheduling, memory management, virtual memory, threads, and process/thread synchronization. This course provides insight into the internal structure of operating systems; emphasis is on concepts and techniques that are valid for all computers. Prerequisite: (CSCI 220 with a grade of C- or higher OR CSCI 262 with a grade of C- or higher) AND CSCI341.

Course Learning Outcomes

- Learn the essential concepts in the design and implementation of operating systems: what they can do, what they contain, and how they are implemented.
- Sharpen C/C++ programming and object-oriented development skills and practice on simulation software design and implementation and get familiar with using Linux system calls.
- Understand fundamental resource management and scheduling techniques that will help you establish the necessary base-knowledge required in many CS-related fields of industry and research: performance optimization, security considerations, resource sharing, etc.

CSCI443. ADVANCED PROGRAMMING CONCEPTS USING JAVA. 3.0 Semester Hrs.
This course will quickly review programming constructs using the syntax and semantics of the Java programming language. It will compare the constructs of Java with other languages and discuss program design and implementation. Object oriented programming concepts will be reviewed and applications, applets, servlets, graphical user interfaces, threading, exception handling, JDBC, and network - ing as implemented in Java will be discussed. The basics of the Java Virtual Machine will be presented. 3 hours lecture, 3 semester hours Prerequisite: CSCI306.

CSCI444. ADVANCED COMPUTER GRAPHICS. 3.0 Semester Hrs.
Equivalent with MATH444,
This is an advanced computer graphics course, focusing on modern rendering and geometric modeling techniques. Students will learn a variety of mathematical and algorithmic techniques that can be used to develop high-quality computer graphic software. Runtime performance will be evaluated to create optimized real-time graphics applications. In particular, the course will cover global illumination, GPU programming, and virtual and augmented reality. Prerequisites: CSCI441. 3 hours lecture; 3 semester hours.

Course Learning Outcomes

- Describe, replicate, and modify the modern advanced graphics pipeline
- Create a real-time graphics application using an advanced graphics technique

CSCI445. WEB PROGRAMMING. 3.0 Semester Hrs.
Web Programming is a course for programmers who want to develop web-based applications. It covers basic website design extended by client-side and server-side programming. Students should acquire an understanding of the role and application of web standards to website development. Topics include Cascading Style Sheets (CSS), JavaScript, PHP and database connectivity. At the conclusion of the course students should feel confident that they can design and develop dynamic Web applications on their own. Prerequisite: CSCI306. Co-requisite: CSCI403.

Course Learning Outcomes

- Design and develop a modern website using HTML5 + CSS3 while applying software engineering principles.
- Discuss and apply proper design methodologies to create visually appealing and accessible websites.
- Describe the basic structure of the Internet and World Wide Web. Discuss how a URL determines what webpage is returned to a browser from this structure.
- Explain and compare the differences between (1) client-side and server-side processing (2) front- end development and back-end development.
- Create a responsive, dynamic, and interactive website using JavaScript.
- Create a responsive, dynamic, and database backed website using Go.
- Create a responsive, dynamic, and interactive website using AJAX and HTMX.
- Locate information and construct a solution to learn new technologies/languages/tools/libraries on your own.

CSCI446. WEB APPLICATIONS. 3.0 Semester Hrs.
In Web Applications students will learn how to build effective web-based applications. At the completion of this course, students should know HTTP, Hypertext Markup Language (HTML), Cascading Style Sheets (CSS), JavaScript, Ajax, and RESTful architectures. Additionally students should have considered a variety of issues related to web application architecture, including but not limited to security, performance, web frameworks and cloud-based deployment environments. 3 hours lecture; 3 semester hours. Prerequisite: CSCI220 or CSCI262. Co-requisite: CSCI403.

Course Learning Outcomes

- n/a

CSCI448. MOBILE APPLICATION DEVELOPMENT. 3.0 Semester Hrs.
This course covers basic and advanced topics in mobile application development. Topics include the mobile application lifecycle, user interface components and layouts, storing persistent data, accessing network resources, using location and sensor APIs including GPS and accelerometer, starting and stopping system services, and threading. This is a project-based course where students will design and develop complete applications. Prerequisite: CSCI306 with a grade of C- or higher. 3 hours lecture; 3 semester hours.

Course Learning Outcomes

- unchanged

CSCI455. GAME THEORY AND NETWORKS. 3.0 Semester Hrs.
Equivalent with CSCI555,
An introduction to fundamental concepts of game theory with a focus on the applications in networks. Game theory is the study that analyzes the strategic interactions among autonomous decision-makers. Originated

from economics. Influenced many areas in Computer Science, including artificial intelligence, e-commerce, theory, and security and privacy. Provides tools and knowledge for modeling and analyzing real-world problems. Prerequisites: CSCI358, CSCI406. 3 hours lecture; 3 semester hours.

CSCI456. NETWORK SCIENCE. 3.0 Semester Hrs.
Network Science is the study of interconnected systems modeled as nodes joined by links. Networked systems arise in a wide range of applications including social networks, transportation networks, the electrical grid, the Web, and the brain. This course introduces theoretical concepts, analytical techniques and algorithms to interpret, analyze and model these networks. Core topics include small-world phenomena, hubs and centrality, Google's PageRank algorithm, signed networks, several random network models, the detection of communities from the underlying network structure, and network dynamics (e.g., the spread of epidemics). Theoretical concepts will be supplemented with hands-on Python-based programming exercises. Prerequisite: CSCI220.
Course Learning Outcomes

- Describe fundamental concepts of network science.
- Analyze and interpret real-world systems modeled as networks.
- Apply tools to measure and evaluate network properties.
- Compare network models and assess their strengths and limitations.
- Explain the properties of small-world networks and their implications for connectivity.
- Calculate and interpret centrality measures to identify important nodes.
- Detect and evaluate community structure in networks using computational methods.
- Design and explain network-based studies of processes such as diffusion.

CSCI460. SOFTWARE STARTUPS: FROM IDEA TO LAUNCH. 3.0 Semester Hrs.
This course is intended for students who wish to first-hand experience what it takes to get a software idea off the ground. You may already have an idea, or along with classmates, can choose from one of the course's prefabricated ideas. You will learn how to test your idea with real market feedback – you will interview potential users, develop user stories, design wireframes (skeletal frameworks that help visualize one's idea), develop and launch a working beta product, and subsequently iterate on your product with user feedback to continue to refine and drive user adoption. You will learn different go-to-market strategies and gain a deeper understanding of your market. In addition, this course places a real emphasis on interpersonal dynamics and driving to founder-level productivity. Through interactive role-plays, perspective-taking drills, and frameworks for how to give and receive feedback without being hurtful, students will learn how to effectively work together as part of a team. This course is experiential and requires a technical background and strong software development skills to take an idea and implement a hands-on solution over the course of the term. Prerequisite: CSCI306.
Course Learning Outcomes

- In the context of a problem that the student sets out to solve, synthesize the pain of target users through conducting user interviews.
- Create wireframes based on insights from user interactions.
- Develop and launch a working beta product based on user needs.
- Engage in a feedback loop with users to refine their product and drive user adoption.
- Foster high performance in teams by developing proficiency in feedback mechanisms.
- Develop mastery of storytelling in the context of software startup ventures.

CSCI470. INTRODUCTION TO MACHINE LEARNING. 3.0 Semester Hrs.
The goal of machine learning is to build computer systems that improve automatically with experience, which has been successfully applied to a variety of application areas, including, for example, gene discovery, financial forecasting, and credit card fraud detection. This introductory course will study both the theoretical properties of machine learning algorithms and their practical applications. Students will have an opportunity to experiment with machine learning techniques and apply them to a selected problem in the context of term projects. Prerequisite: (CSCI101 or CSCI102 or CSCI128) and (MATH201 or MATH334) and MATH332.
Course Learning Outcomes

- 1. Understand the basic ideas of supervised learning, unsupervised learning and reinforcement learning, as well as their applications to real-world problems.
- 2. Use the linear models for regression and classification and write programs to implement these models using off-the-shelf packages.
- 3. Understand the kernel methods to deal with nonlinear problems and use support vector machines and neural networks to solve real-world problems.
- 4. Understand unsupervised learning and solve clustering and dimensionality reduction problems.
- 5. Understand reinforcement learning and its applications in robotics.

CSCI471. COMPUTER NETWORKS I. 3.0 Semester Hrs.
This introduction to computer networks covers the fundamentals of computer communications, using TCP/IP standardized protocols as the main case study. The application layer and transport layer of communication protocols will be covered in depth. Detailed topics include application layer protocols (HTTP, FTP, SMTP, and DNS), transport layer protocols (reliable data transfer, connection management, and congestion control), network layer protocols, and link layer protocols. In addition, students will program client/server network applications. Prerequisite: (CSCI220 or CSCI262) AND (CSCI210 or CSCI274).
Course Learning Outcomes

- using TCP/IP standardized protocols

CSCI473. ROBOT PROGRAMMING AND PERCEPTION. 3.0 Semester Hrs.
Equivalent with CSCI573,
In this class students will learn the basics of integrated robot system programming and the design and use of algorithms for robot perception. Students will learn how to use the ROS robot middleware for the design of robot systems for perceiving and navigating the world; develop reinforcement learning based models for perception-informed autonomous navigation; and develop computational models for 3D robot perception and perceptual representation of human data. Prerequisite: (CSCI220 or CSCI262),and,(MATH201 or MATH334).
Course Learning Outcomes

- 1. Explain the basic concepts in human-centered robotics
- 2. Model and analyze human behaviors for human-robot interaction applications

- 3. Recognize the cutting-edge human-centered robotics research and applications
- 4. Apply the learned knowledge to other fields

CSCI474. INTRODUCTION TO CRYPTOGRAPHY. 3.0 Semester Hrs.
Equivalent with MATH474,
This course is primarily oriented towards the mathematical aspects of cryptography, but is also closely related to practical and theoretical issues of computer security. The course provides mathematical background required for cryptography, including relevant aspects of number theory and mathematical statistics. The following aspects of cryptography will be covered: symmetric and asymmetric encryption, computational number theory, quantum encryption, RSA and discrete log systems, SHA, steganography, chaotic and pseudo-random sequences, message authentication, digital signatures, key distribution and key management, and block ciphers. Many practical approaches and most commonly used techniques will be considered and illustrated with real-life examples. Prerequisite: CSCI220 or CSCI262, CSCI358, MATH334 or MATH335 or MATH201.
Course Learning Outcomes

- Explain the basic terms and concepts in cryptography.
- Correlate number theory with cryptography.
- Analyze the limitations of classical cryptographic algorithms.
- Compare the similarities and differences between symmetric and asymmetric encryption algorithms.
- Assess the security strength of cryptographic techniques.
- Choose appropriate cryptographic techniques such as hashing, message authentication, and digital signature for addressing real-world security and privacy problems.
- Critique research work in applied cryptography.

CSCI475. INFORMATION SECURITY AND PRIVACY. 3.0 Semester Hrs.
Information Security and Privacy provides a hands-on introduction to the principles and best practices in information and computer security. Lecture topics will include basic components of information security including threat assessment and mitigation, policy development, forensics investigation, and the legal and political dimensions of information security. Completion of CSCI274 recommended. Prerequisite: (CSCI220 or CSCI262) and (CSCI 341) and (CSCI 210 or CSCI274).
Course Learning Outcomes

- Explain the fundamental security and privacy protection principles, techniques, and practices.
- Describe risk mitigation, legal requirements, and ethical considerations in cybersecurity.
- Analyze real-world security and privacy problems in modern computing environments.
- Practice security and privacy protection tools and techniques.
- Critique existing security and privacy protection techniques or studies.

CSCI476. DEEP LEARNING. 3.0 Semester Hrs.
Machine Learning is a key component of Artificial Intelligence that allows computers to generate accurate outcomes without being explicitly programmed. Over the past decade, machine learning has powered speech recognition, augmented reality, self-driving vehicles, and biometric authentication. The purpose of this course is to bridge general machine learning concepts to real-world applications. This course provides a broad introduction to machine learning, including supervised learning, unsupervised learning, and feature learning. During the course, students will design a project that applies concepts from class to several different real-world problems and challenges and create applications to solve them. Prerequisite: CSCI470.
Course Learning Outcomes

- Design deep learning models to address supervised and unsupervised learning tasks implemented in Python that meet evaluation criteria such as accuracy, F1 score, and mean-square error.
- Apply state-of-the-art deep learning architectures for computer vision tasks involving object detection, image generation, and self-driving vehicles.
- Develop deep learning models for natural language processing tasks using pre- trained transformers.
- Utilize Markov Decision Process, Monte Carlo Method, and deep reinforcement learning techniques to solve sequential decision-making problems.
- Evaluate ethical challenges in deep learning models, including bias, fairness, transparency, and interpretability.
- Demonstrate clear communication of deep learning results and insights to both technical and non-technical audiences.

CSCI477. ELEMENTS OF GAMES AND GAME DEVELOPMENT. 3.0 Semester Hrs.
This course provides an overview of computer and video game development along with practical game projects designed to introduce the student to the computer entertainment industry. Topics will include the nature of games, the game player, game play, game design, game mechanics, story and character, game worlds, interface and the game development process. Students will be required to develop code both in C++ and with the use of a game engine. Prerequisite: CSCI220 or CSCI262.
Course Learning Outcomes

- Define the process of game software design and development from start to finish.
- Develop a game design document and project plan to be utilized in the development and implementation of a game.
- Develop a working game product using an existing game engine.
- Write game programming code and scripting.
- Identify the core types of AI behavior and their uses, such as pathfinding, fuzzy logic, cooperative behavior, decision trees, neural nets, adaptive and heuristics.
- Discuss the importance of story, character, actions, and rewards as part of a successful game design.
- Implement story, character, gameplay, challenges, and mechanics in the design of a game.

CSCI478. INTRODUCTION TO BIOINFORMATICS. 3.0 Semester Hrs.
Bioinformatics is the theory, application and development of computing tools to solve problems and create hypotheses in all areas of biological sciences, which has contributed to advances in biology by providing tools that handle datasets too large and/or complex for manual analysis. This course focuses on an introduction to computational analysis of genetic variation and computational interdisciplinary research in genetics. The topics of this course include introduction to genetics, identification of genes involved in disease, inferring human population history, technologies for obtaining genetic information, and genetic sequencing, with an emphasis on formulating interdisciplinary problems

as computational problems and then solving those problems using computational techniques from statistics and computer science. Prerequisite: CSCI101 or CSCI102 CSCI128.
Course Learning Outcomes

- Construct a systems level abstraction of the dynamics of a cell by relating components of the cell such as genes, proteins, and metabolic reactions.
- Design and implement algorithms for solving pairwise, multiple sequence alignments, and search sequence databases using sequences from NCBI GenBank or Uniprot.
- Design and implement algorithms for constructing phylogenetic trees for a set of biological sequences and apply them on real datasets on sequence families, such as SARS-CoV-2 variants.
- Apply existing tools to predict protein structures from sequence and compare multiple protein structures using sequences from Uniprot and structures from the Protein Data Bank.
- Interpret and analyze transcriptome data such as microarray data or RNA-Seg data to reveal clusters of genes/samples and identify differentially expressed genes.
- Design and implement algorithms for analyzing biological networks such as protein-protein interaction networks, gene-regulatory networks, and metabolic networks.

CSCI480. COMPUTER SCIENCE HONORS THESIS. 3.0 Semester Hrs.
Prerequisite: CSCI306. 3 hours research; 3 semester hours. Repeatable for credit up to 6 semester hours.
Course Learning Outcomes

- TBA

CSCI490. PROGRAMMING CHALLENGES III. 1.0 Semester Hr.
This course is the third of three courses in the Programming Challenges sequence which covers problem solving patterns and paradigms found in technical interviews and programming competitions. The students will learn more advanced dynamic programming, graph theory, and algorithms while focusing on algorithmic complexity to solve problems in a finite amount of time. Prerequisite: CSCI390. Co-requisite: CSCI406.
Course Learning Outcomes

- Solve common programming problems and identify underlying patterns
- Argue and prove correctness of solution
- Determine and argue space and time complexity of solutions
- Combine common algorithms to solve problems
- Improve skills in technical interviews and programming competitions

CSCI498. SPECIAL TOPICS. 1-6 Semester Hr.
(I, II) Pilot course or special topics course. Topics chosen from special interests of instructor(s) and student(s). Usually the course is offered only once. Prerequisite: none. Variable credit; 1 to 6 credit hours. Repeatable for credit under different titles.

CSCI498. SPECIAL TOPICS. 1-6 Semester Hr.

CSCI498. SPECIAL TOPICS. 1-6 Semester Hr.

CSCI498. SPECIAL TOPICS. 1-6 Semester Hr.

CSCI498. SPECIAL TOPICS. 1-6 Semester Hr.

CSCI499. INDEPENDENT STUDY. 1-6 Semester Hr.
(I, II) Individual research or special problem projects supervised by a faculty member, when a student and instructor agree on a subject matter, content, and credit hours. Prerequisite: "Independent Study" form must be completed and submitted to the Registrar. Variable credit; 1 to 6 credit hours. Repeatable for credit.

CSCI499. INDEPENDENT STUDY. 1-6 Semester Hr.

CSCI499. INDEPENDENT STUDY. 1-6 Semester Hr.

CSCI499. INDEPENDENT STUDY. 1-6 Semester Hr.

CSCI499. INDEPENDENT STUDY. 1-6 Semester Hr.

CSCI499. INDEPENDENT STUDY. 1-6 Semester Hr.

CSCI499. INDEPENDENT STUDY. 1-6 Semester Hr.

CSCI499. INDEPENDENT STUDY. 1-6 Semester Hr.

CSCI499. INDEPENDENT STUDY. 1-6 Semester Hr.

CSCI499. INDEPENDENT STUDY. 0.5-6 Semester Hr.

CSCI499. INDEPENDENT STUDY. 1-6 Semester Hr.

Professor and Department Head

Iris Bahar

Professors

Qi Han

Dinesh Mehta

Chuan Ye

Associate professors

Mehmet Belviranli

Dong Chen

Neil Dantam

Thomas Williams

Bo Wu

Dejun Yang

Assistant professors

Micah Corah

Kaveh Fathian

Gabriel Fierro

Kelsey Fulton

Guannan Liu

C. Estelle Smith

Teaching Professors

Tolga Can

Wendy Fisher

Christopher Painter-Wakefield

Jeffrey Paone

Teaching Associate Professors

Aysu Betin Can

Keith Hellman

Phillip Romig III

Kathleen Kelly

Robert Thompson

Teaching Assistant Professor

Zibo Wang

Professor of Practice

Christine Liebe

Emeriti Professors

Tracy Camp, Emeritus Professor

William Hoff, Emeritus Associate Professor

Cyndi Rader, Emeritus Teaching Professor