COMPUTER SCIENCE

The part-time Computer Science program balances theory with practice, offers an extensive set of traditional and cutting-edge courses, and provides the necessary flexibility to accommodate working professionals with various backgrounds. The program appeals to those with undergraduate degrees in computer science seeking to broaden or deepen their understanding, as well as scientists and engineers who wish to gain deeper insights into the field. The program is also a good option for those with practical computer science experience wishing to formalize their computer science background.

Courses are offered at the Applied Physics Laboratory and online.

Bioinformatics Joint Program

This program is offered jointly by the Zanvyl Krieger School of Arts and Sciences and the Whiting School of Engineering. However, the administration is handled by the Zanvyl Krieger School of Arts and Sciences, and applications for admission to the Master of Science in Bioinformatics program must be submitted directly to Zanvyl Krieger School of Arts and Sciences (bioinformatics.jhu.edu). In addition to supplying official transcripts, applicants must provide a résumé or curriculum vitae and a 500-word statement of purpose. The admissions committee reserves the right to request additional information from applicants, such as GRE scores or letters of recommendation, if needed to assess their candidacy for admission.

Program Committee

Lanier Watkins, Program Chair
Principal Professional Staff
JHU Applied Physics Laboratory

Robert S. Grossman, Vice Program Chair Emeritus
Principal Professional Staff (retired)
JHU Applied Physics Laboratory

Anthony N. Johnson, Program Manager
Senior Professional Staff
JHU Applied Physics Laboratory

Eleanor Boyle Chlan
Senior Professional Staff (retired)
JHU Applied Physics Laboratory

Theodore Colbert, III
Executive Vice President, The Boeing Company
President and Chief Executive Officer, Boeing Global Services

Anton Dahbura
Co-Director, Institute for Assured Autonomy
Johns Hopkins University

Mary Galvin
Alumni
JHU Engineering for Professionals

John Hurley
Professor, Cyberspace Strategies and Data Analytics
National Defense University

Tom Longstaff
CTO, Software Engineering Institute
Carnegie Mellon University

William Robinson
Vice Provost for Academic Advancement
Executive Director of the Provost's Office for Inclusive Excellence
Vanderbilt University

Ralph Semmel
Director
JHU Applied Physics Laboratory

J. Miller Whisnant
Principal Professional Staff
JHU Applied Physics Laboratory

Programs

- Computer Science, Graduate Certificate (https://e-catalogue.jhu.edu/engineering/engineering-professionals/computer-science/computer-science-graduate-certificate/)
- Computer Science, Master of Science (https://e-catalogue.jhu.edu/engineering/engineering-professionals/computer-science/computer-science-master/)
- Computer Science, Post-Master's Certificate (https://e-catalogue.jhu.edu/engineering/engineering-professionals/computer-science/computer-science-post-masters-certificate/)

Courses

EN.605.101. Introduction to Python.
Not for a letter grade. Offered pass/fail only. This is a six-week course. The withdrawal deadline is the end of the fourth week. Students must pass each module to pass the course. Course Note(s): Not for graduate credit. This course does not satisfy any admission requirements. Instructor(s): Non-facilitated

EN.605.156. Calculus for Engineers. 4 Credits.
This one-semester Calculus course is designed to equip students with a comprehensive foundation in differential and integral calculus. This one semester accelerated course covers key topics from the semester-based Calculus I and Calculus II courses. The course begins with limits, continuity, and the fundamentals of derivatives, with applications such as optimization and L'Hospital's rule. Students then progress to integrals, learning techniques of integration, applications of the definite integral, and concepts like area, volume, and work. The latter part of the course covers sequences, series, and convergence tests, along with a focused study of Taylor and Power Series. Emphasis is placed on problem-solving and modeling real-world problems using calculus. This accelerated course requires a strong grasp of algebra, precalculus, and trigonometry, as it moves at a rapid pace and assumes prior familiarity with foundational mathematical concepts. Students are expected to engage actively with the material through lectures, discussions, and extensive practice. By the end of the course, students will have developed both the conceptual understanding and computational skills necessary for further study in engineering.

EN.605.201. Introduction to Programming Using Java. 3 Credits.
This course enables students without a background in software development to become proficient programmers who are prepared for a follow-on course in data structures. The Java language will be used to introduce foundations of structured, procedural, and object-oriented programming. Topics include input/output, data types, operators, program control flow structures, arrays, strings, and methods. Students will also be introduced to classes, objects, inheritance, polymorphism, encapsulation, abstraction, exception handling, processing streams and files, collections, wrappers, and generics, and graphical user interfaces. Students will complete several programming assignments and projects to develop their problem-solving skills and to gain experience in detecting and correcting software errors. Prerequisite(s): One year of college mathematics. Course Note(s): Not for graduate credit. A programming methodology course is needed for admission to the Computer Science, Cybersecurity, Data Science, or Information Systems Engineering program. Students who lack this prerequisite can fulfill admission requirements by completing this course with a grade of B– or better.

EN.605.202. Data Structures. 3 Credits.
This course investigates abstract data types (ADTs), recursion, algorithms for searching and sorting, and basic algorithm analysis. ADTs to be covered include lists, stacks, queues, priority queues, trees, sets, and dictionaries. The emphasis is on the trade-offs associated with implementing alternative data structures for these ADTs. There will be four substantial programming assignments. This course will be taught in a language agnostic fashion. Students may choose to develop their work in Java, C++, or Python Prerequisite(s): One year of college mathematics. EN.605.201 Introduction to Programming Using Java or EN.605.206 Introduction to Programming in Python or equivalent. Course Note(s): Not for graduate credit. A course in data structures is needed for admission to the Computer Science and Cybersecurity program. Students who lack this prerequisite can fulfill admission requirements by completing this course with a grade of B– or better. A course in data structures is conditionally required for admission to the Information Systems Engineering program. Students who lack this prerequisite can satisfy it by completing this course with a grade of B– or better before taking any course that requires it. A second course in programming is required for admission to the Artificial Intelligence program. Students who lack this prerequisite can satisfy it by completing this course with a grade of B– or better before taking any course that requires it. Students in the Artificial Intelligence program who plan to take the EN.605.621 Foundations of Algorithms and EN.605.649 Introduction to Machine Learning Sequence are required to take EN.605.202 or equivalent.

EN.605.203. Discrete Mathematics. 3 Credits.
This course emphasizes the relationships between certain mathematical structures and various topics in computer science. Topics include set theory, graphs and trees, algorithms, propositional calculus, logic and induction, functions, relational algebra, and matrix algebra. Prerequisite(s): Calculus is recommended. Course Note(s): Not for graduate credit. A mathematics course beyond one year of calculus is needed for admission to the Computer Science, Cybersecurity, or Data Science program. A course in either calculus or discrete mathematics is needed for admission to the Information Systems Engineering program. Students who lack this prerequisite can fulfill admission requirements by completing this course with a grade of B– or better.

EN.605.204. Computer Organization. 3 Credits.
This course examines how a computer operates at the machine level. Students will develop an understanding of the hardware/ software interface by studying the design and operation of computing system components. In addition, students will program at the assembly language level to understand internal system functionality. Finally, students will become familiar with the machine representations of programs and data, as well as the influence of the underlying hardware system on the design of systems software such as operating systems, compilers, assemblers, and linkers and loaders. Prerequisite(s): EN.605.202 - Data Structures is recommended. Course Note(s): Not for graduate credit. A course in computer organization is needed for admission to the Computer Science or Cybersecurity program. Students who lack this prerequisite can fulfill admission requirements by completing this course with a grade of B– or better.

EN.605.205. Molecular Biology for Computer Scientists. 3 Credits.
This course is designed for students who seek to take bioinformatics courses but lack prerequisites in the biological sciences. The course covers essential aspects of biochemistry, cell biology, and molecular biology. Topics include the chemical foundations of life; cell organization and function; the structure and function of macromolecules; gene expression— transcription, translation, and regulation; biomembranes and transmembrane transport; metabolism and cellular energetics; and signal transduction. The application of foundational concepts in developmental biology, neurobiology, immunology, and cancer biology is also introduced. Course Note(s): Not for graduate credit. Several courses in the Bioinformatics track of Computer Science require background in Molecular Biology. Students can fulfill this requirement by completing this course with a grade of B- or better.

EN.605.206. Introduction to Programming Using Python. 3 Credits.
This course is a practical introduction for those interested in learning Python for a wide variety of applications and use cases. The material has been designed to expose you to common techniques and tools you'll be able to exercise immediately. This course assumes no prior development experience and ranges from beginning to intermediate Python concepts including: creating a Python environment, data types, operators/expressions, data and control structures, conditional statements, classes/objects, functions, multi-threaded applications, testing and deployment tools, REST API's, machine learning, and more. You'll also gain valuable experience with tools like PyCharm/ VSCode, Jupyter Notebooks, Git, PyLint, PyDocs/Doxygen, and many more. Each concept is accompanied by real code samples that will be explained in-detail and the assignments will present you with interesting scientific problems to enable you to practice your Python skills for the purpose of solving real, complex problems. The course is textbook-free and provides a number of hand-chosen readings to supplement the lecture materials. Upon completion of the course you will be equipped with knowledge of the skills and tools to begin tackling problems the Pythonic way.Prerequisite(s): One year of college mathematics.Course Note(s): Not for graduate credit. A programming methodology course is needed for admission to the Artificial Intelligence or Data Science programs. Students who lack this prerequisite can fulfill admission requirements by completing this course with a grade of B– or better.

EN.605.207. Introduction to Programming Using C++. 3 Credits.
This course provides introductory and foundational coverage of object-oriented programming principles and techniques using C++. Programming techniques covered by this course include modularity, abstraction, top-down design, specifications documentation, debugging and testing. The core material for this course includes control statements, operators, functions, lists, strings, abstract data types, file I/O, exceptions, pointers, overloading, and recursion. Topics include: Abstract Data Types (ADTs), an introduction to the C++ programming language including string and vectors, encapsulation and information hiding, inheritance and polymorphism, file processing, and templates. Improved programming techniques including adherence to programming standards is also an important part of this course. Course Note(s): Not for graduate credit. A programming methodology course is needed for admission to some programs. Students who lack this prerequisite can fulfill admission requirements by completing this course with a grade of B– or better.

EN.605.256. Modern Software Concepts in Python. 3 Credits.
This course will introduce you to a wide range of advanced topics in Python; the skills most developers don't learn until they've been in the workforce. The material has been designed to expose you to common techniques and tools you'll be able to exercise immediately. It is intended for students who have complete 605.206, Intro to Python who are looking to advance their skills by solving problems across a wide variety of topics: web programming, databases, concurrent programming, cloud computing, machine learning, cyber security, and even quantum computing. You'll be introduced to the most current and most popular technologies including: BeautifulSoup, PyCharm/VSCode, Linux CLI, Docker, Git, AWS, and Flask, as well as a range of Python libraries including: Dask, Dash, Pylint, Pytest, Sphinx, Psychopg3, Sqlalchemy, SciPy, SciKit-Learn, regex, Networkx,¬ numpy, Pytorch, Spacy, Streamlit, Qiskit, and more! Each concept is accompanied by real code samples that will be explained in-detail and the assignments will present you with interesting scientific problems to enable you to practice your Python skills for the purpose of solving real, complex problems. The course is textbook-free and provides several hand-chosen readings to supplement the lecture materials. Upon completion of the course, you should have the knowledge, skills, and tools to take a given problem and comfortably execute a solution using Python and a supporting ecosystem of technologies. Not for graduate credit.
Prerequisite(s): EN.605.206 Introduction to Programming using Python or equivalent course.

EN.605.601. Foundations of Software Engineering. 3 Credits.
Fundamental software engineering techniques and methodologies commonly used during software development are studied. Topics include various life cycle models, project planning and estimation, requirements analysis, program design, construction, testing, maintenance and implementation, software measurement, and software quality. Emphasized are structured and object-oriented analysis and design techniques, use of process and data models, modular principles of software design, and a systematic approach to testing and debugging. The importance of problem specification, programming style, periodic reviews, documentation, thorough testing, and ease of maintenance are covered. Course Note(s): The required foundation courses may be taken in any order but must be taken before other courses in the degree.

EN.605.603. Object-Oriented and Functional Programming in Kotlin. 3 Credits.
This course introduces object-oriented and functional programming in the new programming language Kotlin. Kotlin runs on multiple platforms and virtually anywhere, compiling to native code, JavaScript, the Android runtime, and the Java Virtual Machine. It easily interacts with other Java code. Through this course, you'll become adept at Kotlin programming, an easier-to-use, safer and more productive language than Java. We'll cover the basics of the language, including data types, functions and collections, object-oriented features such as classes, encapsulation, inheritance, composition, delegation and generics, and functional features such as immutability, higher-order functions and functional chaining. You'll learn how to create multi-threaded applications using coroutines and builders that will simplify the use of your libraries using simple Domain-Specific languages. Students will build several projects in Kotlin. Pre-requisites: Competence in a procedural language (such as C, Pascal, or Visual Basic) or object-oriented language (such as Java or C++). Note that this is not an "introduction to programming" class and cannot substitute for EN.605.201; we assume familiarity with programming in general.

EN.605.604. Object-Oriented Programming with C++. 3 Credits.
This course provides in-depth coverage of object-oriented programming principles and techniques using C++. Topics include classes, overloading, data abstraction, information hiding, encapsulation, inheritance, polymorphism, file processing, templates, exceptions, container classes, and low-level language features. The course briefly covers the mapping of UML design to C++ implementation and object-oriented considerations for software design and reuse. The course also relates C++ to GUI, databases, and real-time programming. The course material embraces the C++11 language standard with numerous examples demonstrating the benefits of C++11. Prerequisite(s): Knowledge of a high level block structures language.

EN.605.606. Programming with Domain-Specific Languages. 3 Credits.
Domain-specific languages (DSLs) are little languages you write that look and feel like a spoken way to specify data or write code. You can use them for input and output, incorporating the jargon and nomenclature of your subject-matter experts (SMEs), as well as inside your own code to make it more expressive and fluent, and often simpler. You can use them as part of your build process to generate hundreds of classes full of otherwise tedious and error-prone boilerplate code from a small specification in a consistent manner. In this course, we'll design and implement several types of DSLs. We'll write code to edit and import data, allowing SMEs more natural-feeling access to your software. We'll create APIs in multiple programming languages to make it easier and more secure for others to use your libraries. We'll generate code to improve productivity and reliability in your own software. Course Note(s): Examples and assignments in this class will be done in several programming languages. We assume a high comfort level with Java and the ability to adapt to new languages quickly.
Prerequisite(s): EN.605.601 Intro to Software Engineering

EN.605.607. Agile Software Development Methods. 3 Credits.
This course emphasizes the quick realization of system value through disciplined, iterative, and incremental software development techniques and the elimination of wasteful practices. Students will study the full spectrum of agile methods, including Scrum, Extreme Programming, Lean, Kanban, Dynamic Systems Development Method, and Feature-Driven Development. These methods promote teamwork, rich concise communication, and the frequent delivery of running, tested systems containing the highest-priority stakeholder features. Agile methods are contrasted with common workplace practices and traditional, Waterfall-based methods. Examples of agile adoption in industry are covered, along with scaling methods for large-scale development. Assignments and projects are designed to help students apply agile principles and practices in their own professional or academic context. Additional subthemes in the course include enterprise agility, team dynamics, collaboration, software quality, and metrics for reporting progress. **Prerequisite(s):** EN.605.601 Foundations of Software Engineering

EN.605.608. Software Project Management. 3 Credits.
This course describes the key aspects of a software project. It begins with the job description of a software manager and then addresses those topics germane to successful software development management, including organizing the software development team; interfacing with other engineering organizations (systems engineering, quality assurance, configuration management, and test engineering); assessing development standards; selecting the best approach and tailoring the process model; estimating software cost and schedule; planning and documenting the plan; staffing the effort; managing software cost and schedule during development; risk engineering; and continuous process improvement. Personnel management topics, including performance evaluations, merit planning, skills building, and team building, are also covered. This course introduces software engineers aspiring to become technical team leaders or software project managers to the responsibilities of these roles. For those engineers who have advanced to a software development leadership position, this course offers formal training in software project management. Prerequisite(s): Three to five years technical work experience is recommended.

EN.605.609. DevOps and Secure Software Development. 3 Credits.
This course focuses on three key concepts: Agile Software Development, Infrastructure as Code, and Secure Software Delivery. Throughout this course students will learn how to build modern software systems through version control, automated deployment techniques, and improved documentation. This course gathers the latest publications to instruct students on: source code control, virtualization and containerization (Docker) techniques, build automation tools, software composition management/analysis, cloud security, and application security testing (SAST/DAST/IAST/RASP). The course concludes with a team project where students code a functioning DevSecOps pipeline to automate the assessment of software for security.Prerequisite(s): Prior experience in software development in any language is required. Familiarity with software design, cloud development, and architecture techniques is recommended.

EN.605.611. Foundations of Computer Architecture. 3 Credits.
This course provides a detailed examination of the internal structure and operation of modern computer systems. Each of the major system components is investigated, including the following topics: the design and operation of the ALU, FPU, and CPU; microprogrammed vs. hardwired control, pipelining, and RISC vs. CISC machines; the memory system including caches and virtual memory; parallel and vector processing, multiprocessor systems and interconnection networks; superscalar and super-pipelined designs; and bus structures and the details of low-level I/O operation using interrupt mechanisms, device controllers, and DMA. The impact of each of these topics on system performance is also discussed. The instruction set architectures and hardware system architectures of different machines are examined and compared. The classical Von Neumann architecture is also compared and contrasted with alternative approaches such as data flow machines and neural networks. Course Note(s): The required foundation courses may be taken in any order but must be taken before other courses in the degree.

EN.605.612. Operating Systems. 3 Credits.
The theory and concepts related to operating system design are presented from both developer and user perspectives. Core concepts covered include process management, memory management, file systems, I/O system management including device drivers, distributed systems, and multi-user concepts including protection and security. Process management discussions focus on threads, scheduling, and synchronization. Memory management topics include paging, segmentation, and virtual memory. Students will examine how these concepts are realized in several current open-source operating systems, including Linux. Students will complete several assignments that require the design and implementation of operating system programs using a high-level language.

EN.605.613. Introduction to Robotics. 3 Credits.
This course introduces the fundamentals of robot design and development with an emphasis on autonomy. Robot design, navigation, obstacle avoidance, and artificial intelligence will be discussed. Topics covered in robot design include robot structure, kinematics and dynamics, the mathematics of robot control (multiple coordinate systems and transformations), and designing for autonomy. Navigation topics include path planning, position estimation, sensors (e.g., vision, ultrasonics, and lasers), and sensor fusion. Obstacle avoidance topics include obstacle characterization, object detection, sensors and sensor fusion. Topics to be discussed in artificial intelligence include learning, reasoning, and decision making. Students will deepen their understanding through several assignments and the term-long robot development project.

EN.605.614. System Development in the UNIX Environment. 3 Credits.
This course describes how to implement software systems in a UNIX (POSIX-compliant) operating system environment. Students will discuss and learn the complexities, methodologies, and tools in the development of large systems that contain multiple programs. Topics include an overview of the UNIX system and its general-purpose tools, advanced makefile usage, UNIX system calls, UNIX process management, threads, and basic and advanced interprocess communication. Additional topics include source code configuration control, Perl, and debugging techniques.Prerequisite(s): Familiarity with UNIX, experience with C++ or C.

EN.605.615. Compiler Design with LLVM. 3 Credits.
The components of a compiler appear in every software application that handles input from an external source. This course shows how the components of a compiler are built and how they fit together to extract meaning from the input and how the data flows through the compiler's components to become useful to applications. Students will get practical experience in how to use the LLVM tools to build a complete compiler for a subset of the C++ programming language that can target almost any platform. Students will also get experience in developing a "Just In Time" component for an application that will accept code as input into the application while it is running, to be compiled and linked into the application so the application can execute it. Prerequisites: This course has no formal prerequisites, but experience with C++ is highly recommended because LLVM is written in C++, and therefore, all homework will be in C++, and this course is software homework intensive.

EN.605.616. Multiprocessor Architecture & Programming. 3 Credits.
This course addresses how to utilize the increasing hardware capabilities of multiprocessor computer architecture's highperformance computing platforms for software development. The famous Moore's Law is still alive, although it is now realized from increasing the number of CPU cores instead of increasing CPU clock speed. This course describes the differences between single-core and multi-core systems and addresses the impact of these differences in multiprocessor computer architectures and operating systems. Parallel programming techniques to increase program performance by leveraging the multiprocessor system, including multi-core architectures, will be introduced. Additional topics include program performance analysis and tuning, task parallelism, synchronization strategies, shared memory access and data structures, and task partition techniques. The course encourages hands-on experience with projects selected by the student.

EN.605.617. Introduction to GPU Programming. 3 Credits.
This course will teach the fundamentals needed to utilize the ever-increasing power of the GPUs housed in the video cards attached to our computers. For years, this capability was limited to the processing of graphics data for presentation to the user. With the CUDA and OpenCL frameworks, programmers can develop applications that harness this power directly to search, modify, and quickly analyze large amounts of various types of data. Students will be introduced to core concurrent programming principles, along with the specific hardware and software considerations of these frameworks. In addition, students will learn canonical algorithms used to perform high-precision mathematics and data transformations. Class time will be split between lectures and hands-on exercises. There will be two individual projects in both CUDA and OpenCL programming, which will allow students to independently choose demonstrable goals, develop software to achieve those goals, and present the results of their efforts.

EN.605.618. Introduction to High Performance Computing. 3 Credits.
This course provides an introduction to architectures, programming models, and optimization strategies for parallel and high performance computing systems. This course begins by defining what constitutes parallel computing and a high performance system. We will address the rudimental architectural characteristics of distributed and shared memory systems and their respective programing paradigms. We will discuss how to leverage HPC systems to solve large scale problems in a variety of applications such as image processing, deep learning, and molecular dynamics. The course will conclude with a discussion of evaluating deep neural networks on high-performance computing resources. Students will complete a number of assignments along with a project demonstrating particular applications on parallel processing systems.

EN.605.620. Algorithms for Bioinformatics. 3 Credits.
This follow-on course to data structures (e.g., EN.605.202 Data Structures) provides a survey of computer algorithms, examines fundamental techniques in algorithm design and analysis, and develops problem-solving skills required in all programs of study involving computer science. Topics include advanced data structures (red-black and 2-3-4 trees, union-find), recursion and mathematical induction, algorithm analysis and computational complexity (recurrence relations, big-O notation, introduction to NP-completeness), sorting and searching, design paradigms (divide and conquer, greedy heuristic, dynamic programming), and graph algorithms (depth-first and breadth-first search, minimum spanning trees). Advanced topics are selected from among the following: multithreaded algorithms, matrix operations, linear programming, string matching, computational geometry, and approximation algorithms. Students will form groups to work on difficult problems and also to present an advanced topic at the end of the term. The course will draw on applications from Bioinformatics. Prerequisite(s): EN.605.202 Data Structures or equivalent, and EN.605.201 Introduction to Programming Using Java or EN.605.206 Introduction to Programming in Python or equivalent. EN.605.203 Discrete Mathematics or equivalent is recommended. Course Note(s): This course does not satisfy the foundation course requirement for Computer Science, Data Science, or Cybersecurity. Students can only earn credit for one of EN.605.620, EN.605.621, EN.685.621 or EN.705.621

EN.605.621. Foundations of Algorithms. 3 Credits.
This follow-on course to data structures (e.g., EN.605.202) provides a survey of computer algorithms, examines fundamental techniques in algorithm design and analysis, and develops problem-solving skills required in all programs of study involving computer science. Topics include advanced data structures (red-black and 2-3-4 trees, union-find), recursion and mathematical induction, algorithm analysis and computational complexity (recurrence relations, big-O notation, NP-completeness), sorting and searching, design paradigms (divide and conquer, greedy heuristic, dynamic programming, amortized analysis), and graph algorithms (depth-first and breadth-first search, connectivity, minimum spanning trees, network flow). Advanced topics are selected from among the following: randomized algorithms, information retrieval, string and pattern matching, and computational geometry. Prerequisite(s): EN.605.202 Data Structures or equivalent. EN.605.203 Discrete Mathematics or equivalent is recommended. Course Note(s): The required foundation courses may be taken in any order but must be taken before other courses in the degree. Students can only earn credit for one of EN.605.620, EN.605.621, EN.685.621 or EN.705.621

EN.605.622. Computational Signal Processing. 3 Credits.
This course introduces algorithms and architectures for the analysis and processing of digital signals, taking the computer science perspective. It emphasizes computational complexity and efficiency and the design and implementation of computer algorithms for processing signals, designing digital filters, and effectively presenting and displaying information. Topics include signal analysis, discrete Fourier transform (definition, applications, and fast algorithms), convolution and correlation, spectral estimation and display, filter design, signal encoding/decoding, time-frequency analysis, Software Defined Radio (SDR), arithmetic computational complexity, and applications Background in signal processing and mathematics will be introduced as needed. **Prerequisite(s):** EN.605.621 Foundations of Algorithms or equivalent background, some knowledge of complex numbers and linear algebra (vectors and matrices).

EN.605.624. Logic: Systems, Semantics, and Models. 3 Credits.
Traditionally, logic is the study of correct reasoning. In the last few decades, logic has become increasingly important to knowledge representation -- a subfield of artificial intelligence concerned with developing representations of the world (often called ontologies) that aid computers in understanding and making sense of data. This course will promote both a theoretical and practical understanding of logic as a stepping stone for working in contemporary knowledge representation. We will begin with a review of categorical, propositional, and predicate logic. We will then survey modal logics, which include systems that represent necessity and probability, as well as other systems that represent time, and moral notions such as obligation and permissibility. The second half of the course will then introduce the semantic web and ontology engineering. Students will explore the top-level ontology Basic Formal Ontology (BFO) and gain familiarity using mereological and temporal relations. In addition, students will create ontologies in the web ontology language (OWL2) and use the language SPARQL to query knowledge graphs. Students will have the option of writing either a research paper or creating an ontology in OWL with slides as part of a final project.

EN.605.625. Probabilistic Graphical Models. 3 Credits.
This course introduces the fundamentals behind the mathematical and logical framework of graphical models. These models are used in many areas of machine learning and arise in numerous challenging and intriguing problems in data analysis, mathematics, and computer science. For example, the "big data" world frequently uses graphical models to solve problems. While the framework introduced in this course will be largely mathematical, we will also present algorithms and connections to problem domains. The course will begin with the fundamentals of probability theory and will then move into Bayesian networks, undirected graphical models, template based models, and Gaussian networks. The nature of inference and learning on the graphical structures will be covered, with explorations of complexity, conditioning, clique trees, and optimization. The course will use weekly problem sets and a term project to encourage mastery of the fundamentals of this emerging area. Prerequisite(s): Graduate course in probability and statistics (such as EN.625.603 Statistical Methods and Data Analysis). Course Note(s): This course is the same as EN.625.692 Probabilistic Graphical Models.

EN.605.626. Image Processing. 3 Credits.
Fundamentals of image processing are covered, with an emphasis on digital techniques. Topics include digitization, enhancement, segmentation, the Fourier transform, filtering, restoration, reconstruction from projections, and image analysis including computer vision. Concepts are illustrated by laboratory sessions in which these techniques are applied to practical situations, including examples from biomedical image processing. Prerequisite(s): Familiarity with Fourier transforms.

EN.605.629. Programming Languages. 3 Credits.
A programming language provides instructions to a computing device to perform tasks. A language has a vocabulary and a set of grammatical rules to shape and frame the communication between the user and the computing device. The Programming Languages course compares and contrasts a wide variety of features of numerous old and new programming languages, including programming language history; formal methods of describing syntax and semantics; names, binding, type checking, and scopes; data types; expressions and assignment statements; statement-level control structures; design and implementation of subprograms; lambda calculus; exception handling; support for object-oriented programming; and concurrency. Our course will also introduce logic programming and theorem proving through Prolog, Objective Caml, and Coq framework. The course will survey the fundamental concepts underlying modern programming languages with the goal of understanding paradigms, but not vocational training in any given language. Several examples will be drawn from C, C++, Java, Python, ML, JavaScript, Scheme, Prolog, and Coq.

EN.605.630. Theory of Computation. 3 Credits.
This is a graduate-level course studying the theoretical foundations of computer science. Topics covered will be models of computation from automata to Turing machines, computability, time and space complexity theory, Boolean circuits, and interactive proof systems.

EN.605.631. Statistical Methods for Computer Science. 3 Credits.
Statistical methods are the foundation for data science, artificial intelligence, and much of the field of computer science. Topics include probability, random variables, regression, gradient search, Bayesian methods, graphical methods, and exponential random graph models. Student will have the foundation to excel in future courses in machine learning, data science, algorithms, and more. Practice exercises will develop proficiency in the R programming language.

EN.605.632. Graph Analytics. 3 Credits.
Graphs are a flexible data structure that facilitates fusion of disparate data sets. Applications of graphs have shown steady growth with the development of Internet, cyber, and social networks, presenting large graphs for which analysis remains a challenging problem. This course introduces algorithms and techniques to address large-scale graph analytics. It will blend graph analytics theory, hands-on development of graph analytics algorithms, as well as processing approaches that support the analytics. We will start by introducing graphs, their properties, and example applications, including necessary background on probability and linear algebra. Statistical properties of random and scale-free graphs will be introduced. Graph analytic methods, including centrality measures, graph clustering, partitioning, link inference, and dynamic graph processes such as diffusion, contagion, and opinion formation will be covered. Application of graph analytics to high-dimensional data analysis and data clustering will be discussed. Students will use standard graph interfaces as well as linear algebra-based methods to analyze graphs. Parallelization of analytics to handle larger-scale graphs will be discussed. Students will identify and apply suitable algorithms and analysis techniques to a variety of graph analytics problems, as well as gain experience setting up and solving these problems. There will be hands-on programming assignments.

EN.605.633. Social Media Analytics. 3 Credits.
Today an immense social media landscape is being fueled by new applications, growth of devices (e.g., Smartphones and devices), and human appetite for online engagement. With a myriad of applications and users, significant interest exists in the obvious question, "How does one better understand human behavior in these communities to improve the design and monitoring of these communities?" To address this question a multidisciplinary approach that combines social network analysis (SNA), natural language processing, and data analytics is required. This course combines all these topics to address contemporary topics such as marketing, population influence, etc. There will be several small projects. Prerequisite(s): Knowledge of Python or R; matrix algebra.
Prerequisite(s): Computer Science majors need to complete foundation requirement first.;Foundation Prerequisites for Cybersecurity Majors:EN.605.621 AND EN.695.601 AND EN.695.641

EN.605.634. Crowdsourcing and Human Computation. 3 Credits.
Crowdsourcing and human computation reverses the typical approach to computing. Rather than using computers to conduct computation that is too difficult for a human, many humans are used to conduct computation that is too difficult for a computer. This course explores computer science topics that lie at the intersection of data science and social psychology. Topics include crowdsourcing, social media, social network analysis, games, gamification, ubiquitous computing, and computersupported cooperative work. Laboratory exercises will involve hands-on data collection and analysis to include Mechanical Turk and require programming in R or Python depending on student preference/ proficiency.

EN.605.635. Cloud Computing. 3 Credits.
Cloud computing helps organizations realize cost savings and efficiencies without spending capital resources up front, while modernizing and expanding their IT capabilities. Cloud-based infrastructure is rapidly scalable, secure, and accessible over the Internet —you pay only for what you use. So, enterprises worldwide, big and small, are moving toward cloud-computing solutions for meeting their computing needs, including the use of Infrastructure as a Service (IaaS) and Platform as a Service (PaaS). We have also seen a fundamental shift from shrinkwrapped software to Software as a Service (SaaS) in data centers across the globe. Moreover, providers such as Amazon, Google, and Microsoft have opened their datacenters to third parties by providing low-level services such as storage, computation, and bandwidth. This trend is creating the need for a new kind of enterprise architect, developer, QA, and operational professional—someone who understands and can effectively use cloud-computing technologies and solutions. In this course, we discuss critical cloud topics such as cloud service models (IaaS, PaaS, SaaS); virtualization and how it relates to cloud; elastic computing; cloud storage; cloud networking; cloud databases; cloud security; and architecting, developing, and deploying apps in the cloud. The format of this course will be a mix of lectures, and hands-on demos. Upon completing this course, students will have a deeper understanding of what cloud computing is and the various technologies that make up cloud computing, along with hands-on experience working with a major cloud provider. Prerequisite(s): EN.605.202 Data Structures.

EN.605.636. Autonomic Computing. 3 Credits.
This course provides an introduction to autonomic and self-aware computing. It concentrates on the self-managing and self-awareness properties of computing systems, their architecture, adaptation, and decision making needed for system resiliency by continuingly adapting to changing environments. The vision for autonomic computing is described, how autonomic computing differs from automated and autonomous systems, as well the self-awareness properties and biological inspiration for autonomic systems. Architectures of autonomic systems are covered, which includes autonomic managers that provide the self-management for autonomic systems. Adaptive technology is also covered as well as what makes an autonomic system self-aware. Applications of autonomic computing are discussed, including security and resiliency applications, and how autonomic computing is used and is applied to cloud computing. Hands-on programming assignments as well as a project that provides autonomic capabilities to an IoT device, industrial control system, or other system of the student's choosing will be ongoing throughout the course that provides application of the theories and concepts from the lectures. There will be weekly readings and discussions, with bi-weekly assignments that go into depth on selected topics that also contribute to the projects. With the help of the instructors, students will be encouraged to submit the project write-ups to a conference.

EN.605.641. Principles of Database Systems. 3 Credits.
This course examines the underlying concepts and theory of database management systems. Topics include database system architectures, transaction management, data models, query languages, conceptual and logical database design, and physical organization. The entity-relationship (ER) model, using ER diagram (ERD) and Enhanced ERD, as well as relational models, are investigated in detail. Object-oriented databases are introduced along with legacy systems based on the network. Hierarchical models as well as big data and NoSQL are also briefly described. Mappings from the conceptual level to the logical level, integrity constraints, dependencies, and normalization are studied as a basis for formal design. Theoretical languages such as the relational algebra and the relational calculus are described, and high-level languages such as SQL, triggers and Stored Procedures are discussed. An overview of file organization and access methods is provided as a basis for discussion of query optimization and execution. The course also covers the causes of performance problems and how to improve database application performance during database design and implementation. Course prerequisite(s): EN.605.202 Data Structures.

EN.605.643. Linked Data and the Semantic Web. 3 Credits.
The World Wide Web Consortium (W3C) is endeavoring to create standards and technology that support a distributed "Web of data." Collectively, these advances allow the systems we develop to work and interact more effectively, through the use of XML-based languages, and information on how various tags relate to real-world objects and concepts. This course covers a range of Semantic Web technologies, including RDF (Resource Description Framework - a model for data interchange) and OWL (Web Ontology Language), as well as domain-specific standards and ontologies (formal specifications of how to represent objects and concepts). Representative applications of RDF, OWL, and ontologies to various problems will be discussed. Students will apply course concepts to an in-depth project in an area of personal or professional interest. Prerequisite(s): EN.605.202 Data Structures. Course Note(s): This course may be counted toward a three course track in Bioinformatics.

EN.605.644. XML Design Paradigms. 3 Credits.
The course explores understanding the tradeoffs among XML grammars and XML techniques to solve different classes of problems. Topics include optimization of XML grammars for different XML technologies; benefits of using different XML schema languages; tradeoffs in using different parsing approaches; benefits of parsing technology vs. XML query; the role of Web 2.0 to deliver functionality through various web services approaches; exploiting XML to drive audio, visual, and tactile displays; the role of XML in multiplying the power of standard web browser technologies; and the role of Web 3.0 to deliver Semantic Web functionality. XML technologies that will be covered include XML Schema, XPath, XSLT, SAX, DOM, XQuery, SOAP, WSDL, JAX-B, JAX-WS, REST, RDF, and OWL.
Prerequisite(s): EN.605.681 Principles of Enterprise Web Development or equivalent Java experience.

EN.605.645. Artificial Intelligence. 3 Credits.
This is a foundational course in Artificial Intelligence. Although we hear a lot about machine learning, artificial intelligence is a much broader field with many different aspects. In this course, we focus on three of those aspects: reasoning, optimization, and pattern recognition. Traditionally, the first was covered under "Symbolic AI" or "Good Old Fashioned AI" and the latter two were covered under "Numeric AI" (or more specifically, "Connectionist AI" or "Machine Learning"). However, despite the many successes of machine learning algorithms, practitioners are increasingly realizing that complicated AI systems need algorithms from all three aspects. This approach falls under the ironic heading "Hybrid AI".
In this course, the foundational algorithms of AI are presented in an integrated fashion emphasizing Hybrid AI. The topics covered include state space search, local search, example based learning, model evaluation, adversarial search, constraint satisfaction problems, logic and reasoning, expert systems, rule based ML, Bayesian networks, planning, reinforcement learning, regression, logistic regression, and artificial neural networks (multi-layer perceptrons). The assignments weigh conceptual (assessments) and practical (implementations) understanding equally.
Prerequisite(s): EN.605.621 Foundations of Algorithms. A working knowledge of Python programming is assumed as all assignments are completed in Python.

EN.605.646. Natural Language Processing. 3 Credits.
This course surveys the principal difficulties of working with written language data, the fundamental techniques that are used in processing natural language, and the core applications of NLP technology. Topics covered in the course include language modeling, text classification, labeling sequential data (tagging), parsing, information extraction, question answering, machine translation, and semantics. The dominant paradigm in contemporary NLP uses supervised machine learning to train models based on either probability theory or deep neural networks. Both formalisms will be covered. A practical approach is emphasized in the course, and students will write programs and use open source toolkits to solve a variety of problems. Course prerequisite(s): There are no formal prerequisite courses, although having taken any of EN.605.649 Introduction to Machine Learning, EN.605.744 Information Retrieval, or EN.605.645 Artificial Intelligence is helpful. Course note(s): A working knowledge of Python is assumed. While some of the assigned exercises can be done in any programming language, we will sometimes provide example code in Python, and many of the labs are best solved in Python.Course note(s): A working knowledge of Python is assumed. While some of the assigned exercises can be done in any programming language, we will sometimes provide example code in Python, and many of the labs are best solved in Python.

EN.605.647. Neural Networks. 3 Credits.
This course provides an introduction to concepts in neural networks and connectionist models. Topics include parallel distributed processing, learning algorithms, and applications. Specific networks discussed include Hopfield networks, bidirectional associative memories, perceptrons, feedforward networks with back propagation, and competitive learning networks, including self-organizing and Grossberg networks. Software for some networks is provided. Prerequisite(s): Multivariate calculus and linear algebra.

EN.605.649. Principles and Methods in Machine Learning. 3 Credits.
Analyzing large data sets ("Big Data"), is an increasingly important skill set. One of the disciplines being relied upon for such analysis is machine learning. In this course, we will approach machine learning from a practitioner's perspective. We will examine the issues that impact our ability to learn good models (e.g., inductive bias, the curse of dimensionality, the bias-variance dilemma, and no free lunch). We will then examine a variety of approaches to learning models, covering the spectrum from unsupervised to supervised learning, as well as parametric versus non-parametric methods. Students will explore and implement several learning algorithms, including logistic regression, nearest neighbor, decision trees, and feed-forward neural networks, and will incorporate strategies for addressing the issues impacting performance (e.g., regularization, clustering, and dimensionality reduction). In addition, students will engage in online discussions, focusing on the key questions in developing learning systems. At the end of this course, students will be able to implement and apply a variety of machine learning methods to real-world problems, as well as be able to assess the performance of these algorithms on different types of data sets. Prerequisite(s): EN.605.202 – Data Structures or equivalent.
Prerequisite(s): EN.605.202 – Data Structures or equivalent, EN.605.621 – Foundations of Algorithms or EN.685.621 – Algorithms for Data Science or 705.621 – Introduction to Algorithms

EN.605.651. Principles of Bioinformatics. 3 Credits.
This course is an interdisciplinary introduction to computational methods used to solve important problems in DNA and protein sequence analysis. The course focuses on algorithms but includes material to provide the necessary biological background for science and engineering students. Algorithms to be covered include dynamic programming for sequence alignment, such as Smith-Waterman, FASTA, and BLAST; hidden Markov models, such as the forward, Viterbi, and expectation maximization algorithms; a range of gene-finding algorithms; phylogenic tree construction; and clustering algorithms. Prerequisite(s): Familiarity with probability and statistics; working knowledge of Java, C++, C, Perl, MATLAB or Python; EN.605.205 Molecular Biology for Computer Scientists or a course in molecular biology; and a course in either cell biology or biochemistry.

EN.605.652. Biological Databases and Database Tools. 3 Credits.
The sequencing of thousands of genomes, including those related to disease states, interest in proteomics, epigenetics, and variation resulted in an explosive growth in the number of biological databases, as well as the need to develop new databases to handle the diverse content being generated. This course focuses on the design of biological databases and examines issues such as those related to data modeling, heterogeneity, interoperability, evidence, and tool integration. It also surveys a wide range of biological databases and their access tools and enables students to develop proficiency in their use. Databases introduced include genome and sequence databases such as GenBank and Ensembl, as well as protein databases such as PDB and UniProt. Databases related to RNA, sequence variation, pathways and interactions, metagenomics, and epigenomics are also presented. Tools for accessing and manipulating data from databases such as BLAST, genome browsers, multiple sequence alignment, gene finding, and protein tools are reviewed. The programming language Perl is introduced, along with the use of Perl in obtaining data via web services and in storing data in an SQLite database. Students will use Perl (Python may be used for some homework), biological databases, and database tools to complete homework assignments. Students will also design a database and will write code in the language of their choice to create their own database as a course project. Note for AAP students: this is a programming class. Students new to programming may find this course challenging and time consuming.
Prerequisite(s): (For JHEP Students) EN.605.205 Molecular Biology for Computer Scientists or AS.410.634 Practical Computer Concepts for Bioinformatics or equivalent; EN.605.641 Principles of Database Systems or equivalent; EN.605.202 Data Structures and EN.605.201 Introduction to Programming Using Java.

EN.605.653. Computational Genomics. 3 Credits.
This course focuses on current problems of computational genomics. Students will explore bioinformatics software, discuss bioinformatics research, and learn the principles underlying a variety of bioinformatics algorithms. The emphasis is on algorithms that use probabilistic and statistical approaches. Topics include analyzing eukaryotic, bacterial, and viral genes and genomes, genome sequencing and assembling, finding genes in genomes and identifying their biological functions, predicting regulatory sites, and assessing gene and genome evolution. Prerequisite(s): EN.605.205 Molecular Biology for Computer Scientists or equivalent and familiarity with probability and statistics.

EN.605.656. Computational Drug Discovery,Dev. 3 Credits.
Recent advances in bioinformatics and drug discovery platforms have brought us significantly closer to the realization of rational drug design and development. Across the pharmaceutical industry, considerable effort is being invested in developing experimental and translational medicine, and it is starting to make a significant impact on the drug discovery process itself. This course examines the major steps of the evolving modern drug discovery platforms, the computational techniques and tools used during each step of rational drug discovery, and how these techniques facilitate the integration of experimental and translation medicine with the discovery/development platforms. The course will build on concepts from a number of areas including bioinformatics, computational genomic/ proteomics, in-silico system biology, computational medicinal chemistry, and pharmaceutical biotechnology. Topics covered in the course include comparative pharmacogenomics, protein/ antibody modeling, interaction and regulatory networks, QSAR/ pharmacophores, ADME/toxicology and clinical biomarkers. Relevant mathematical concepts are developed as needed in the course. Prerequisite(s): EN.605.205 Molecular Biology for Computer Scientists or equivalent.

EN.605.657. Statistics for Bioinformatics. 3 Credits.
This course provides an introduction to the statistical methods commonly used in bioinformatics and biological research. The course briefly reviews basic probability and statistics including events, conditional probabilities, Bayes theorem, random variables, probability distributions, and hypothesis testing and then proceeds to topics more specific to bioinformatics research, including Markov chains, hidden Markov models, Bayesian statistics, and Bayesian networks. Students will learn the principles behind these statistical methods and how they can be applied to analyze biological sequences and data. Prerequisite(s): EN.605.205 Molecular Biology for Computer Scientists or equivalent, and AS.410.645 Biostatistics or another statistics course.

EN.605.661. Computer Vision. 3 Credits.
This course provides an overview of fundamental methods in computer vision from a computational perspective. Methods studied include: camera systems and their modeling, computation of 3-D geometry from binocular stereo, motion, and photometric stereo, and object recognition, image segmentation, and activity analysis. Elements of machine learning and deep learning are also included. Practical application of these concepts is emphasized through written and programming homework assignments. Students will also have an opportunity to further explore concepts through a semester long project. Prerequisite(s): Intro to Programming, Linear Algebra & Probability/Statistics

EN.605.662. Data Visualization. 3 Credits.
This course explores the underlying theory and practical concepts in creating visual representations of large amounts of data. It covers the core topics in data visualization: data representation, visualization toolkits, scientific visualization, medical visualization, information visualization, flow visualization, and volume rendering techniques. The related topics of applied human perception and advanced display devices are also introduced. Prerequisite(s): Experience with data collection/ analysis in data-intensive fields or background in computer graphics (e.g., EN.605.667 Computer Graphics) is recommended.

EN.605.667. Computer Graphics. 3 Credits.
This course examines the principles of computer graphics, with a focus on the mathematics and theory behind 2D and 3D graphics rendering. Topics include graphics display devices, graphics primitives, 2D and 3D transformations, viewing and projection, color theory, visible surface detection and hidden surface removal, lighting and shading, and object definition and storage methods. Practical application of these concepts is emphasized through laboratory exercises and code examples. Laboratory exercises use the C++ programming language and OpenGL on a PC. Prerequisite(s): Familiarity with linear algebra.

EN.605.668. Computer Gaming Engines. 3 Credits.
This course examines the fundamentals of computer game software designed to familiarize the students with a broad understanding of many aspects of computer gaming. The course prioritizes broad coverage over deep coverage. Topics include 2D/3D graphics, input/output, real-time simulations, resource management, vector mathematics, sound, concurrency, and so forth, with an emphasis on cross-platform development. Practical applications of these topics are covered in programming assignments throughout the semester with the goal of developing a simple game of the student's choice. Programming assignments are done in C or C++ on PC, MacOS, or Linux.

EN.605.671. Principles of Data Communications Networks. 3 Credits.
This course provides an introduction to the field of data communications and computer networks. It covers the principles of data communications, the fundamentals of signaling, basic transmission concepts, transmission media, circuit control, line sharing techniques, physical and data link layer protocols, error detection and correction, data compression, network security techniques and protocols, common carrier services and data networks, the mathematical techniques used for network design and performance analysis, Ethernet and Wi-Fi local area networks, and the TCP/IP-based Internet. Potential topics include analog and digital signaling; data encoding and modulation; Shannon channel capacity; synchronous and asynchronously transmission; RS232 physical layer interface standards; FDM, TDM, and STDM multiplexing techniques; inverse multiplexing; analog and digital transmission; V series modem standards; PCM encoding and T1 transmission circuits; LRC, VRC, and CRC error detection techniques; Hamming and Viterbi forward error correction techniques; character and bit-oriented protocols, information transparency, and BSC and HDLC data link layer protocols; Huffman, MNP5, and Lempel-Ziv-Welch data compression algorithms; circuit, message, packet, and cell switching techniques; public key and symmetric encryption algorithms, authentication, digital signature, and message digest techniques, secure e-mail, PGP, TLS/SSL, Kerberos, and IPsec security algorithms; reliability, availability, and queuing analysis performance techniques; Ethernet, Fast Ethernet, Gigabit and 10 Gigabit Ethernet LANs; Wi-Fi 4, Wi-Fi 5, Wi-Fi 6, and Wi-Fi 7 LANs; IPv4 and IPv6 network layer protocols, RIP, OSPF, and BGP4 routing protocols; and TCP and UDP transport layer protocols.

EN.605.674. Network Programming. 3 Credits.
Emphasis is placed on the theory and practice associated with the implementation and use of the most common processto-process communications associated with UNIX. The interprocess communications comprise both local and distributed architectures. The distributed communications protocols include those most widely implemented and used: the worldwide Internet protocol suite [the Transmission Control Protocol/ Internet Protocol (TCP/IP), and the U.S. government-mandated International Organization for Standardization (ISO) protocol suite]. Practical skills are developed, including the ability to implement and configure protocol servers (daemons) and their clients. Students are expected to have working knowledge of UNIX.
Prerequisite(s): EN.605.671 Principles of Data Communications Networks, or EN.605.614 System Development in the UNIX Environment .

EN.605.675. Protocol Design. 3 Credits.
This course covers the formal design, specification, and validation of computer and network protocols. Design, implementation, and verification of protocols will be illustrated using the latest simulation tools, such as OPNET and NS2. Protocol examples include the latest wired and wireless networks, such as the IEEE 802.X family, as well as protocols in VoIP, Web 2.0, and network security. The course focuses on protocol specification, structured protocol design, protocol models, and protocol validation. Students will gain hands-on experience using simulation tools to design, validate, and assess protocols.

EN.605.677. Internetworking with TCP/IP I. 3 Credits.
This course investigates the underlying technology of the Internet and culminates with a team-based research project. The presentation begins with a survey of distributed applications operating over the Internet, including the Web, electronic mail, VoIP, instant messaging, file transfers and peer-to-peer file sharing. The course investigates the details of the Internet architecture and the TCP/IP protocol suite, covering the protocols that provide communications services to end systems and the management and control protocols that create the Internet from disparate underlying networks and technologies. Communications-related protocols analyzed in detail include the foundational Internet Protocol (IP), the connection-oriented reliable Transmission Control Protocol (TCP), the connectionless User Datagram Protocol (UDP) and the Real-Time Protocol (RTP) for streaming media. To allow the student to understand the control and management of the Internet, the course analyzes protocols that support naming (DNS), addressing and configuration (DHCP), management (SNMP) and the dynamic IP routing protocols RIP, OSPF and BGP.
Prerequisite(s): EN.605.671 Principles of Data Communications Networks.

EN.605.681. Principles of Enterprise Web Development. 3 Credits.
This course examines fundamental aspects of Enterprise Web Development including client, middleware and databases as a foundation for follow on courses. It introduces the student to client side development using HTML 5, CSS and JavaScript. After a brief review of Object Oriented Programming in Java, Swing is used to introduce common user interface design patterns. Network protocols and multithreading concepts using Java transition into server-side technologies like Servlets, JavaserverPages and ReST. Java database development with JDBC and web security are also introduced during the semester. While the class covers development using build tools (Maven), basic IDEs are utilized to facilitate the teaching of concepts and demonstration through examples. Prerequisite(s): EN.605.202 Data Structures.

EN.605.682. Web Application Development with Java. 3 Credits.
This project-oriented course will enable students to use various techniques for building browser-based applications for dynamically generated websites, e-commerce, web-enabled enterprise computing, and other applications that require web access to server-based resources. Particular attention will be paid to methods for making web-based applications efficient, maintainable, and flexible. The course will use at least two sets of tools: servlets/JSP and a higher-level Java-based framework such as JSF 2.0. Major topics will include handling HTTP request information, generating HTTP response data, tracking sessions, designing custom tag libraries or components, page templating, asynchronously updating pages with Ajax, and separating content from presentation through use of the MVC architecture. Additional topics may include HTML5, database access techniques for web apps, web app security, and dependency injection in web apps (e.g., with the Spring framework). Course Note(s): Formerly 605.682 Web Application Development with Servlets and JavaServer Pages (JSP).
Prerequisite(s): EN.605.681 Principles of Enterprise Web Development or equivalent Java experience.

EN.605.683. Java Enterprise Development: Processes, Tools and Infrastructure. 3 Credits.
The focus of this course is to get the student acclimated to the process and tools used in the design to delivery cycle of an Enterprise Application using Java. It will introduce students to the use of build tools and repositories for creating and maintaining software in a team environment. It will then then cover tools and techniques for improving the quality of Enterprise Software like unit and integration testing, code optimization and profiling. It will also cover techniques for automation of processes in testing and deployment of software; like Continuous Integration and the use and orchestration with virtual containers. The course will also look at some modern integrated development environments and demonstrate how they integrate with the aspects of the class. A sample of tools covered in the class will include Maven, Gradle, JMeter, Postman, Jenkins, Git, JProfiler, Docker, Docker Compose, Eclipse and IntelliJ.
Prerequisite(s): EN.605.681 Principles of Enterprise Web Development with Java

EN.605.686. Mobile Application Development for the Android Platform. 3 Credits.
This project-oriented course will investigate application development for the Android mobile platform. We will explore techniques for building well-structured applications, from local and remote data access using databases and REST APIs, through view models that synchronously and asynchronously manage and expose that data, to a Jetpack Compose user interface layer for a simple specification and testing. Assigned projects include demonstrations of full data flow from database to user interface, use of graphics and user-screen interaction, Google Maps, REST API communication and testing. Prerequisites: Strong comfort with Java and its basic APIs. Comfort with concepts such as callbacks, threads, lists, and maps. EN.605.603 Object-Oriented and Functional Programming in Kotlin is recommended but not required. Course Notes: This course is taught using Kotlin, the primary language for Android development (and required for Jetpack Compose). Kotlin knowledge is not required for this course, and its basics will be covered from the assumption that students are very comfortable with Java. Tools for developing and testing Android apps are available free of charge. Note that Android emulators may run slowly on some machines; physical Android devices are strongly recommended, but not required, for this course.

EN.605.687. Mobile Application Development for the iOS Platform. 3 Credits.
This project-oriented course will investigate application development on iOS platforms. First, we will cover the main language for iOS, Swift, Apple's in-house, open sourced language for iOS, macOS and watchOS development, along with tools such as Xcode, Instruments, and Swift Playgrounds. Second, we will discuss the aspects of creating an application: the application life cycle, user experience and data presentation, user interface elements (including how to use the SwiftUI framework), and application performance. Then, we will discuss the application frameworks that the iOS SDK provides: CoreData, SpriteKit, MapKit, and Notifications, to name just a few. Finally, we will prepare your app for deployment, considering localization and internationalization, accessibility, and App Store Connect. By the end of the class, students will be able to use Xcode, implement the Model-View-Controller paradigm, use Protocols and Delegates, construct a user interface that operates on many different devices, store and retrieve data on the network, interact with the OS or other applications, distinguish between the various iOS frameworks, and explain the App publication process. Course prerequisite(s): EN.605.201 Introduction to Programming Using Java or equivalent Java or Objective C experience. Course note(s): Access to a Mac running the current version of macOS *IS REQUIRED* for this class. Development tools can be downloaded for free from the Mac App Store. Additional hardware (iPhones, iPods, iPads) is strongly suggested, as several class examples and some projects will work best (or only work) on devices. *THIS REQUIREMENT IS SUBJECT TO CHANGE*

EN.605.691. Entrepreneurship for Computer Scientists. 3 Credits.
Entrepreneurship is defined as the pursuit of opportunity beyond the resources currently controlled. A successful business venture needs more than just a good idea. This course focuses on providing the student with the knowledge and skills necessary to start, manage and grow a successful business venture. Classes will cover the basics of opportunity identification, prototyping, business models, marketing, business planning, financing, and management. The course is designed to provide students with both the practical and theoretical knowledge to understand and pursue entrepreneurial ideas. Lectures, case studies and readings as well as a practical understanding of real-world situations will help students to develop a business plan, conduct market research, create a marketing campaign, and the ability to pitch their ideas to others. Ultimately the goal of this course is to provide the student with the knowledge and confidence to turn ideas into successful business ventures.

EN.605.701. Software Systems Engineering. 3 Credits.
Software Systems Engineering applies engineering principles and the system view to the software development process. The course focuses on the engineering of complex systems that have a strong software component. This course is based on the philosophy that the key to engineering a good software system lies just as much in the process that is followed as in the purely technical regime. The course will show how good a software development process is and how to make a software process better by studying successful techniques that have been employed to produce correct software systems within budget. Topics are explored in a sequence designed to reflect the way one would choose to implement process improvements. These topics include steps to initiate process change, methods to establish control over the software process, ways to specify the development process, methods for quantitative process control, and how to focus on problem prevention. Students will prepare term projects. Prerequisite(s): EN.605.202 Data Structures; one software engineering course beyond EN.605.601 Foundations of Software Engineering.

EN.605.702. Cloud-native Architecture and Microservices. 3 Credits.
Cloud-native architecture is an approach to designing, developing, and deploying applications that leverage the advantages of the cloud computing model. Cloud-native applications are composed of loosely coupled microservices that run in containers, orchestrated by platforms like Kubernetes. This course will introduce the concepts, principles, and practices of cloud-native architecture, microservices, Kubernetes and serverless compute. Students will learn how to design, implement, and deploy cloud-native applications. The course will also cover topics such as observability, security, resilience, scalability, and DevOps.
Prerequisite(s): EN.605.635 Cloud Computing, EN.605.681 Principles of Enterprise Web Development or EN.605.682 Web Application Development with Java.

EN.605.704. Object-Oriented Analysis and Design. 3 Credits.
This course describes fundamental principles of object-oriented modeling, requirements development, analysis, and design. Topics include specification of software requirements; object-oriented static and dynamic analysis approaches using the Unified Modeling Language (UML); object-oriented design; object-oriented reuse and maintainability, including design patterns; software implementation concerns; state models; persistence; and the Object Constraint Language (OCL). Prerequisite(s): While there are no programming assignments in this course, experience in an object-oriented programming language such as C++ or Java is important.

EN.605.705. Software Safety. 3 Credits.
This course describes how to develop and use software that is free of imperfections that could cause unsafe conditions in safety-critical systems. Systems engineering and software engineering techniques are described for developing "safeware," and case studies are presented regarding catastrophic situations that resulted from software and system faults that could have been avoided. Specific techniques of risk analysis, hazard analysis, fault tolerance, and safety tradeoffs within the software engineering paradigm are discussed. Prerequisite(s): EN.605.202 Data Structures.

EN.605.707. Software Patterns. 3 Credits.
Software patterns encapsulate the knowledge of experienced software professionals in a manner that allows developers to apply that knowledge to similar problems. Patterns for software are analogous to the books of solutions that enable electrical engineers and civil engineers to avoid having to derive every new circuit or bridge design from first principles. This course will introduce the concept of software patterns, and explore the wide variety of patterns that may be applied to the production, analysis, design, implementation, and maintenance of software. The format of the course will emphasize the discussion of patterns and their application. Each student will be expected to lead a discussion and to actively participate in others. Students will also be expected to introduce new patterns or pattern languages through research or developed from their own experience. Programming exercises performed outside of class will be used enhance discussion and illustrate the application of patterns.
Prerequisite(s): EN.605.604 Object-Oriented Programming with C++ or permission of instructor.

EN.605.708. Tools and Techniques of Software Project Management. 3 Credits.
This course examines tools and techniques used to lead software-intensive programs. Techniques for RFP analysis and proposal development are explored, and techniques of size estimation (function points, feature points, and lines-ofcode estimation) and the use of models such as COCOMO to convert size to effort and schedule are described. In addition, conversion of estimated effort to dollars and the effects of fringe, overhead, skill mix profiles, and staffing profiles on total dollar cost are explained. Moreover, techniques for estimating effort and planning the COTS intensive development programs are described, and tools and techniques for measuring process maturity and process efficiency (e.g., CMMi, Lean, Six Sigma, and Kaizen) are addressed. The course also investigates the formation and management of virtual teams, as well as techniques that can be used to ensure success in this environment. Finally, the course addresses topics that require collaboration between the project manager and human resources, such as personnel retention strategies, managing unsatisfactory performance, and formal mentoring programs. Prerequisite(s): Three to five years technical work experience is recommended.

EN.605.713. Advanced Robotics. 3 Credits.
This course explores advanced topics in navigation, perception, mapping, and control for mobile robots. Mathematical models of commonly used sensors such as wheel tachometers, accelerometers, gyroscopes, radio frequency transceivers, monocular visual cameras, laser scanners, and LiDAR imagers will be developed. Planar and 3D vehicle motion models will be derived in both deterministic and stochastic settings. Navigation algorithms covered in the course include dead reckoning, beacon localization, beacon simultaneous localization and mapping (SLAM), monocular visual camera based SLAM, and monocular visual camera + inertial sensor based SLAM. A survey of relevant stability/control theory, estimation theory, projective geometry, and image processing algorithms will be provided. Students will be exposed to these topics through course lectures, weekly homework assignments, and a term-long robotics hardware project.
Prerequisite(s): EN.605.613 – Introduction to Robotics or equivalent introductory robotics course.

EN.605.715. Software Development for Real-Time Embedded Systems. 3 Credits.
This course examines the hardware and software technologies behind real-time, embedded computer systems. From smart kitchen appliances to sophisticated flight control for airliners, embedded computers play an important role in our everyday lives. Hardware topics include microcomputers and support devices (e.g., flash, ROM, DMA, timers, clocks, A/D, and D/A), as well as common applications (e.g., servo and stepper motor control, automotive sensors, and voice processing). Software topics focus on unique aspects of embedded programming and include interrupts, real-time control, communication, common design patterns, and special test considerations. The course also explores the unique tools that are used to develop and test embedded systems. Labs, beginning with using Bare Metal and Free RTOS on Arduino for simple devices and culminating with using Linux on Raspberry-Pi for Quad-Copter flight control, are developed.

EN.605.716. Modeling and Simulation of Complex Systems. 3 Credits.
This multi-disciplinary course focuses on the application of modeling and simulation principles to complex systems. A complex system is a large-scale nonlinear system consisting of interconnected or interwoven parts (such as a biological organism, an ecological system, the economy, fluids or strongly-coupled solids). The subject is interdisciplinary with foundations in mathematics, nonlinear science, numerical simulations and statistical physics. The course begins with an overview of complex systems, followed by modeling techniques based on nonlinear differential equations, networks, and stochastic models. Simulations are conducted via numerical calculus, analog circuits, Monte Carlo methods, and cellular automata. In the course we will model, program, and analyze a wide variety of complex systems, including dynamical and chaotic systems, cellular automata, and iterated functions. By defining and iterating an individual course project throughout the term, students will gain hands-on experience and understanding of complex systems that arise from combinations of elementary rules. Students will be able to define, solve, and plot systems of linear and non-linear systems of differential equations and model various complex systems important in applications of population biology, epidemiology, circuit theory, fluid mechanics, and statistical physics.Course prerequisite(s): Knowledge of elementary probability and statistics and previous exposure to differential equations. Students applying this course to the MS in Bioinformatics should also have completed at least one Bioinformatics course prior to enrollment.Course note(s): This course may be counted toward a three-course concentration in Bioinformatics.

EN.605.721. Design and Analysis of Algorithms. 3 Credits.
In this follow-on course to EN.605.621 Foundations of Algorithms, design paradigms are explored in greater depth, and more advanced techniques for solving computational problems are presented. Topics include randomized algorithms, adaptive algorithms (genetic, neural networks, simulated annealing), approximate algorithms, advanced data structures, online algorithms, computational complexity classes and intractability, formal proofs of correctness, sorting networks, and parallel algorithms. Students will read research papers in the field of algorithms and will investigate the practicality and implementation issues with state-of-the-art solutions to algorithmic problems. Grading is based on problem sets, programming projects, and in-class presentations.Prerequisite(s): EN.605.621 Foundations of Algorithms or equivalent; EN.605.203 Discrete Mathematics or equivalent.
Prerequisite(s): EN.605.621 Foundations of Algorithms or equivalent; EN.605.203 Discrete Mathematics or equivalent.

EN.605.724. Applied Game Theory. 3 Credits.
In many organizations in the private and the public sectors, there is a need to support complex decisions that include a game-theoretic aspect. These decisions impact activities ranging from tactical to strategic, and play out in a number of problems, including monitoring and management of ongoing operations, the dynamics of organizational relationships in the competitive environment, and military force planning. This course extends treatment of game theoretic concepts and constructs, and explores their implementation and application, highlighting key issues such as decision space exploration and analysis, visualization, and the creation and use of models for specific domains. Students will have the opportunity to design a course project based on their area of professional or personal interest.

EN.605.727. Computational Geometry. 3 Credits.
This course covers fundamental algorithms for efficiently solving geometric problems, especially ones involving 2D polygons and 3D polyhedrons. Topics include elementary geometric operations; polygon visibility, triangulation, and partitioning; computing convex hulls; proximity searching, Voronoi diagrams, and Delaunay triangulations with applications; special polygon and polyhedron algorithms such as point containment and extreme point determination; point location in a planar graph subdivision; dimension reduction in data; and robot motion planning around polygon obstacles. Applications to such areas as computer graphics, big data analytics and pattern recognition, geometric databases, numerical taxonomy, and robotics will be addressed. The course covers theory to the extent that it aids in understanding how the algorithms work. Emphasis is placed on algorithm design and implementation. Programming projects are an important part of the coursework. Prerequisite(s): Foundations of algorithms. Some familiarity with linear algebra.
Prerequisite(s): EN.605.621 Foundations of Algorithms. Some familiarity with linear algebra.

EN.605.728. Quantum Computation. 3 Credits.
Quantum computing is no longer a purely theoretical notion. The NSA and NIST are preparing to transition to quantum resistant cryptography. We have now entered the intermediate-scale quantum era, with near-term applications such as quantum machine learning being explored. Scalable quantum computers aren't here yet, but ongoing developments suggest they are on their way. This course provides an introduction to quantum computation for computer scientists: the focus is on algorithms rather than physical devices, and familiarity with quantum mechanics (or any physics at all) is not a prerequisite. Instead, pertinent aspects of the quantum mechanics formalism are developed as needed in class. The course begins with an introduction to the QM formalism. It then develops the abstract model of a quantum computer, and discusses how quantum algorithms enable us to achieve, for some problems, a significant speedup (in some cases an exponential speedup) over any known classical algorithm. This discussion provides the basis for a detailed examination of quantum integer factoring, quantum search, and other quantum algorithms. The course also explores quantum error correction, quantum teleportation, and quantum cryptography. It concludes with a glimpse at what the cryptographic landscape will look like in a world with quantum computers. Required work includes problem sets and a research project. Prerequisites: Some familiarity with linear algebra and with the design and analysis of algorithms or instructor permission.

EN.605.729. Formal Methods. 3 Credits.
All science requires mathematics. Formal methods used in developing computer systems are mathematically based techniques for describing system properties. These formal methods then can provide frameworks within which developers can specify, develop, verify, and prove systems in a systematic, rather than ad hoc manner. According to some researchers, the application of formal specification and verification methods could avoid disasters such as Heartbleed bug, Ariane 5 rocket explosion, and Therac-25 radiation therapy machine harms. This course is an introduction to the vast world of formal methods. The course starts with review of propositional logic, predicate logic, and covers set theoretic specification methods via Z, temporal specification via PTL, grammars, and logic based methods via Caml and Coq proof assistant. Each student will carry out an investigation of an existing formal verification system, applying it to a suitable problem of the student's choice. Among possible projects will be the formal verification of problem solutions such as designing a semaphore, designing a machine learning algorithm, a web interface, a test suite, a sophisticated data structure, or a theorem.

EN.605.731. Survey of Cloud Computing Security. 3 Credits.
The promise of significant cost savings and inherent flexibility of resources are an impetus for the adoption of cloud computing by many organizations. Cloud computing also introduces privacy and security risks that are not traditionally present in a siloed data center. This course focuses on these security concerns and countermeasures for a cloud environment. An overview of cloud computing and virtualization, the critical technology underpinning cloud computing, provides the necessary background for these threats. Additional topics vary but may include access control, identity management, denial of service, account and service hijacking, secure APIs, malware, forensics, regulatory compliance, trustworthy computing, and secure computing in the cloud. This course follows a seminar-style format where students are expected to lead class discussions and write a publication-quality paper as part of a course project.

EN.605.740. Machine Learning: Deep Learning. 3 Credits.
Deep learning (DL) has emerged as a powerful tool for solving data-intensive learning problems such as supervised learning for classification or regression, dimensionality reduction, and control. As such, it has a broad range of applications including language processing, computer vision, medical imaging, and perception-based robotics. The goal of this course is to directly apply fundamental concepts of DL to current research problems. Students will apply theoretical underpinnings of machine learning, commonly used architectures for DL, current challenges including ethics and fairness, and specialized applications with a particular focus on computer vision. Students will complete several DL projects using standardized data sets in addition to a small-team research project on topics of their own interest. Recommended prior classes: A neural network OR machine learning course: Examples: EN.605.647, EN.625.638, EN.525.670, EN.605.649, EN.705.601, EN.605.646, or others as approved by the instructor. A working knowledge of Python is assumed. Prior coding experience data munging, ML, and visualization libraries is highly recommended: Example: Python, Numpy, Pandas, ScikitLearn, Matplotlib, etc.

EN.605.741. Large-Scale Database Systems. 3 Credits.
This course investigates the theory and practice of modern large-scale database systems. Large-scale approaches include distributed relational databases; data warehouses; and non-relational databases including HDFS, Hadoop, Accumulo for query and graph algorithms, and Mahout bound to Spark for machine learning algorithms. Topics discussed include data design and architecture; database security, integrity, query processing, query optimization, transaction management, concurrency control, and fault tolerance; and query formulation, graph algorithms, and machine learning algorithms on large-scale distributed data systems. At the end of the course, students will understand the principles of several common large-scale data systems including their architectures, performance, and costs. Students will also gain a sense of which approach is recommended for different requirements and circumstances. **Prerequisite(s):** EN.605.202 Data Structures; EN.605.641 Principles of Database Systems or equivalent. Familiarity with "big-O" concepts and notation is recommended.

EN.605.742. Deep Neural Networks. 3 Credits.
This hands-on course provides a practical introduction to deep neural networks (DNNs), designed to deepen students' understanding of advanced deep learning (DL) techniques. Modeled after the brain's architecture, DNNs drive powerful applications in natural language processing (NLP), computer vision (CV), and speech processing—especially with unstructured data like text, images, video, and audio.The course begins with a four-week refresher on machine learning (ML), focusing on model evaluation and feature engineering using Python and Scikit-Learn (SKL). Prior experience with Python and SKL-based ML models is expected. From there, we transition to TensorFlow and Keras, exploring key DNN architectures including convolutional neural networks (CNNs), recurrent neural networks (RNNs), long short-term memory (LSTM), attention mechanisms, generative adversarial networks (GANs), deep reinforcement learning (DRL), transfer learning, and more. Students will read seminal DL papers and collaborate in teams on weekly Kaggle challenges to apply their skills. Teamwork is central to the course, which concludes with a final presentation based on a public ML/DL competition. **Prerequisite(s):** A course in Machine Learning

EN.605.743. Advanced Artificial Intelligence. 3 Credits.
Many advanced artificial intelligence systems are using both Machine Learning and Symbolic AI to solve subproblems. This course builds on the foundations of EN.605.645 Artificial Intelligence by delving more deeply into those AI algorithms and approaches that go under the name of Good Old Fashioned AI or Symbolic AI. In this course, we will cover logic programming, expert systems and business rules, fuzzy logic, case based reasoning, and knowledge graphs. We will also explore more advanced versions of planning and reinforcement learning algorithms. The instructor may add additional topics as warranted. Prerequisite(s): EN.605.645 Artificial Intelligence or permission of instructor. **Prerequisite(s):** EN.605.645 Artificial Intelligence

EN.605.744. Information Retrieval. 3 Credits.
A multibillion-dollar industry has grown to address the problem of finding information. Commercial search engines are based on information retrieval: the efficient storage, organization, and retrieval of text. This course covers both the theory and practice of text retrieval technology. Topics include automatic index construction, formal models of retrieval, Internet search, text classification, multilingual retrieval, question answering, and related topics in NLP and computational linguistics. A practical approach is emphasized and students will complete several programming projects to implement components of a retrieval engine. Students will also give a class presentation based on an independent project or a research topic from the IR literature. Prerequisite(s): EN.605.202 Data Structures or permission of the instructor

EN.605.745. Reasoning Under Uncertainty. 3 Credits.
This course is concerned with the problems of inference and decision making under uncertainty. It develops the theoretical basis for a number of different approaches and explores sample applications. The course discusses foundational issues in probability and statistics, including the meaning of probability statement, and the necessity of a rational agent acting in accord with probability theory. We will look at possible generalizations of Bayesian probability, including Dempster-Shafer theory. Next, we will develop algorithms for Bayesian networks—graphical probabilistic models—for exact and approximate inference and consider several application areas. Finally, the course will examine the problem of making optimal decisions under uncertainty. We will explore the conceptual foundations of decision theory and then consider influence diagrams, which are graphical models extending Bayesian networks to the domain of decision analysis. As time permits, we will also look at Bayesian games and Markov decision processes. Pertinent background in probability and theoretical computer science is developed as needed in the course.

EN.605.746. Advanced Machine Learning. 3 Credits.
This course focuses on recent advances in machine learning and on developing skills for performing research to advance the state of knowledge in machine learning. The material integrates multiple ideas from basic machine learning and assumes familiarity with concepts such as inductive bias, the bias-variance trade-off, the curse of dimensionality, and no free lunch. Topics range from determining appropriate data representations and models for learning, understanding different algorithms for knowledge and model discovery, and using sound theoretical and experimental techniques in assessing learning performance. Specific approaches discussed cover nonparametric and parametric learning; supervised, unsupervised, and semi-supervised learning; graphical models; ensemble methods; and reinforcement learning. Topics will be discussed in the context of research reported in the literature within the previous two years. Students will participate in seminar discussions and will present the results of a semester-long research project of their own choosing.
Prerequisite(s): EN.605.649 Introduction to Machine Learning; multivariate calculus;Students cannot receive credit for both EN.605.746 and EN.625.742

EN.605.747. Evolutionary and Swarm Intelligence. 3 Credits.
Recently, principles from the biological sciences have motivated the study of alternative computational models and meta-heuristic approaches to problem solving. Proceeding from a machine learning perspective, this course explores how principles from theories of evolution, natural selection, and swarming behavior can be used to construct machines that exhibit nontrivial behavior. In particular, the course covers techniques from evolutionary computation and swarm intelligence for developing software agents capable of solving problems as members of a larger population of agents. Specific topics addressed include representation and schemata; selection, reproduction, and recombination; theoretical models of computational intelligence; optimal allocation of trials (i.e., bandit problems); search, optimization, and machine learning; evolution of programs; population and swarm dynamics; and emergent behavior. Students will participate in seminar discussions and will complete and present the results of an individual project.
Prerequisite(s): EN.605.649 Introduction to Machine Learning; multivariate calculus.

EN.605.751. Algorithms for Structural Bioinformatics. 3 Credits.
This course is an interdisciplinary approach to the concepts, principals, computational methods and algorithms used in structural bioinformatics. It focuses on the fundamental aspects of structural biology along with computational methods and algorithms for studying protein folding, structure prediction and analysis. Algorithms for the prediction and annotation of protein secondary and tertiary structure and for structurestructure comparison will be studied in depth. We will also show how such algorithms and methods can be adapted for use with nucleic acids structure prediction and analysis. Students will apply various software tools and structure-visualization software to protein structure prediction and structurestructure comparison.
Prerequisite(s): EN.605.205 Molecular Biology for Computer Scientists or equivalent. EN.605.651 Principles of Bioinformatics is recommended.

EN.605.755. Systems Biology. 3 Credits.
Systems biology is the study of complex biological systems using theoretical, mathematical, and computational tools and concepts. The advent of genomics, big data, and highpowered computing is allowing better understanding and elucidation of these systems. Central to systems biology is the development of computational models, based on sound statistics, which incorporate biological structures and networks, and can be informed and tested, with data on multiple scales. In this class, students will learn to develop and use different types of models of complex biological systems and how to test and perturb them. Students will learn basic biological system components and dynamics, as well as the data formats, sources, and modeling tools required to interrogate them. Tools will be used relating to functional genomics, evolution, biochemical systems, and cell biology. Students will utilize a model they have developed and available data from public repositories to investigate both a discovery-based project and a hypothesis based project. Prerequisite(s): Courses in molecular biology (EN.605.205 Molecular Biology for Computer Scientists or AS.410.602 Molecular Biology) and differential equations.Prerequisite(s): Courses in molecular biology (EN.605.205 Molecular Biology for Computer Scientists or AS.410.602 Molecular Biology) and differential equations.
Prerequisite(s): Courses in molecular biology (EN.605.205 Molecular Biology for Computer Scientists or AS.410.602 Molecular Biology) and differential equations.

EN.605.759. Independent Project in Bioinformatics. 3 Credits.
This course is for students who would like to carry out a significant project in bioinformatics as part of their graduate program. The course may be used to conduct minor research, an in-depth literature survey, or a software implementation related to recent developments in the field. Students who enroll in this course are encouraged to attend at least one industry conference in bioinformatics related to their area of study. To enroll in this course, the student must be within two courses of degree completion and must obtain the approval and support of a sponsoring faculty member. Course Note(s): A student may not receive credit for both EN.605.759 and EN.605.802 Independent Study in Computer Science II.

EN.605.767. Applied Computer Graphics. 3 Credits.
This course examines advanced rendering topics in computer graphics. The course focuses on the mathematics and theory behind 3D graphics rendering. Topics include 3D surface representations including fractal geometry methods; visible surface detection and hidden surface removal; and surface rendering methods with discussion of lighting models, color theory, texturing, and ray tracing. Laboratory exercises provide practical application of these concepts. The course also includes a survey of graphics rendering applications (animation, modeling and simulation, and realistic rendering) and software. Students perform laboratory exercises using the C++ programming language.
Prerequisite(s): EN.605.667 Computer Graphics or familiarity with three-dimensional viewing and modeling transformations.;Foundation Prerequisites for Cybersecurity Majors:EN.605.621 AND EN.695.601 AND EN.695.641

EN.605.768. Advanced Game Design and Development Engines. 3 Credits.
This course resides where Computer Science meets fundamental game, visualization, and design theories. The course develops the practice of digital games related to the latest engines for research and development. These systems have evolved into more large-scale, online multi-player interactive game environments. With advanced game engines that offer more rapid prototyping, the course focuses on a combination of theoretical research tied to applied design and engine development. Students will study topics that include: 1.) Game design concepts of goals, themes, storyboarding, and levels, 2.) Visualization methods with 3D objects and scenes, art concepts, and styles, and 3.) Game engine element subtopics of history, genres, mechanics, gameplay, types of development engines, and Artificial Intelligence (AI). Through lectures, example real-world games, case studies of game engines used for recreation, educational, and simulation systems, hands-on exercises, and independent research opportunities, students will be given the resources to understand design concepts and how to develop games with the latest engines such as Unreal and Unity. Working knowledge of C++ or C# is required.

EN.605.771. Wired and Wireless Local and Metropolitan Area Networks. 3 Credits.
This course provides a detailed examination of wired and wireless local and metropolitan area network technologies, protocols, and the methods used for implementing LAN- and MAN-based enterprise intranets. The structure and operation of the IEEE 802 media access control (MAC) and physical layer protocols are examined in detail. The 802.2 logical link control, 802.3/Ethernet, 802.4 token bus, and 802.5 token ring protocols are analyzed, and the construction of LAN-based enterprise intranets is examined through a detailed analysis of bridging, routing, and switching techniques. High-speed wired LAN and MAN protocols are discussed. The wireless 802.11 Wi-Fi LAN standards are analyzed. The wireless 802.15.1 and the 802.15.4 PAN standards are discussed in detail. Finally, the wireless 802.16 MAN and the 802.11s mesh standards are examined. Topics include Manchester and Differential Manchester encoding techniques; bus, star, and ring topologies; optical fiber, coaxial cable, and UTP media; baseband, broadband, and carrierband bus networks; hubs, switched LANs, and full duplex LANs; CSMA/CD and CSMA/CA media access techniques; transparent and source routing bridges, BPDUs, spanning tree protocol, RSTP, and MSTP protocols; VLANs, QinQ, and VXLANs; Fast Ethernet, 100VG AnyLAN, Gigabit Ethernet, 10-Gigabit Ethernet, 40-100 Gigabit Ethernet, and 200/400 Gigabit Ethernet; Link Aggregation; 802.6 DQDB and 802.17 Resilient Packet Ring MANs; Wi-Fi 4, Wi-Fi 5, Wi-Fi 6, and Wi-Fi 7; Bluetooth and ZigBee PANs, Wi-Fi and Bluetooth security; OFDMA and MIMO multi-access schemes; WiMAX and 802.11s networks; and mesh path discovery.
Prerequisite(s): EN.605.202 Data Structures; EN.605.671 Principles of Data Communications Networks or EN.635.611 Principles of Network Engineering.

EN.605.776. Fourth Generation Wireless Communications: WiMAX and LTE. 3 Credits.
This course compares the WiMAX and LTE fourth-generation (4G) technologies and their performance. An overview of the IEEE 802.16 standards (802.16d/e/j/m/n/p) and WiMAX Forum (Fixed WiMAX vs. Mobile WiMAX, Interoperability certification and Core network) is presented along with the 3GPP standards for LTE and LTE-Advanced as well as LTE network architecture. For WiMAX, the MAC, call flow, 2D resource map, QoS, and scheduling are presented. For LTE, both control plane and data plane protocols for Evolved UMTS Terrestrial Radio Access Network (E-UTRAN) and Evolved Packet Core (EPC) are presented. The topics include protocol architecture, bearer management, signaling, radio resource control (RRC), packet data convergence protocol (PDCP), radio link control (RLC), and MAC. In addition, the role of universal subscriber identity module (USIM), eNodeB, mobility management entity (MME), serving gateway (S-GW), packet data network gateway (P-GW), and home subscription server (HSS) as well as the call flow across these various nodes will be presented. The 2D resource grid along with QoS and scheduling will be explained in detail. The voice over LTE (VoLTE), self-organizing network (SON), LTE-direct, and LTE-Advanced [including coordinated multipoint (CoMP), carrier aggregation, and Intercell interference coordination (ICIC)] will be presented. Finally, spectrum considerations as well as the concept of white space and dynamic spectrum access (DSA) will be discussed. LTE security will be discussed in detail. The course will also highlight some of the Open Source LTE projects, and will discuss the experimental results from various testbeds.
Prerequisite(s): EN.605.202 Data Structures; EN.605.671 Principles of Data Communications Networks or EN.635.611 Principles of Network Engineering and another course in the Data Communications and Networking track.

EN.605.777. Internetworking with TCP/IP II. 3 Credits.
This course builds on the foundation established in 605.677: Internetworking with TCP/ IP I. The architecture and operation of the Internet is examined. The classful addressing concept are introduced and the mapping of Internet addresses to physical addresses is discussed along with the extensions to the addressing paradigm: subnet addressing, classless addressing, and network address translation. Performance enhancements that provide quality of service (QoS) over the Internet and faster routing through the use of IP switching techniques are discussed. Techniques for providing multicasting and mobility over the Internet are examined. The next generation IP protocol (IPv6) is introduced. The changes and enhancements to the IPv4 protocol and addressing architecture are examined. The techniques developed for transitioning from IPv4 to IPv6 are discussed in detail. Security considerations are addressed through an examination of Virtual Private Networks (VPNs) and the IP Security (IPsec) protocols. New transport layer protocols to overcome the limitations of TCP are examined. The convergence of circuit and packet switching is discussed. Finally, Voice Over IP (VoIP) protocols are examined in detail. Topics include subnet addressing, CIDR, DHCP, DNS, NAT, IntServ, DiffServ, RSVP, CIP, MPOA, IP Switching, Tag Switching, MPLS, IP Multicast, IGMP, Reliable Multicast, Multicast Routing Protocols, IP Mobility Home and Foreign Agents, Tunneling, Proxy and Gratuitous ARP, IPv6 Addressing, IPv6 protocol and extension headers, Neighbor Discovery, IPv6 Stateless Address Autoconfiguration, DHCPv6; dual stack, protocol translation, and tunneling transitioning techniques; 6to4, ISATAP, and Teredo protocols; VPN Tunneling; PPTP, L2F, L2TP and SOCKSv5 VPNs; ESP, AH, and IKEv3 security protocols; Head-of-line blocking, multihoming; SCTP, Multipath TCP, and QUIC transport protocols; VoIP, H.323 Gateways and Gatekeeper; and SIP, SDP, RTP, MGCP, and Megaco/H.248 protocols.
Prerequisite(s): EN.605.677 Internetworking with TCP/IP I or 605.671 Principles of Data Communications Networks or the equivalent.

EN.605.779. Network Design and Performance Analysis. 3 Credits.
Networking services are a staple of our daily life. Different types of networks surround us all day long. This ubiquitous networking, thanks to smartphones and tablet computers, gives us the convenience of information at our fingertips. The right network architecture provides the fundamental support for network services, such as the products from Facebook, Google, Apple, etc. This course covers the details of network design and the design process. Starting from requirement specifications, a detail flow analysis is introduced. Examples of different network architecture designs, both in wireline and wireless, will be discussed, including mobile Ad Hoc network (MANET), mesh network, 4G cellular networks, wide area network (WAN), cloud networks, and advanced software define networking (SDN). Performance analyses and network security aspects are considered at every step of the design. Secured architecture covers Virtual Private Network (VPN) and Transport Layer Security (TLS)-based systems, with details on firewall and intrusion detection configurations. The course encourages hands-on projects selected from real network system problems.

EN.605.784. Enterprise Computing with Java. 3 Credits.
This comprehensive course explores core application aspects for developing, configuring, securing, deploying, and testing a Java-based service using a layered set of modern frameworks and libraries that can be used to develop full services. Students will learn thru lecture, examples, and hands-on experience to build multi-tier enterprise services using a configurable set of server-side technologies.The course will specifically cover designing and building components, a data tier, synchronous and asynchronous server-side logic, and integration with the web.The student will also learn to secure the application and tackle various build, testing, and development issues. Specific framework and specification emphasis (e.g., Jakarta EE, Spring, Spring Boot) for designing and developing server-side components will vary per section.
Prerequisite(s): EN.605.202 Data Structures; EN.605.681 Principles of Enterprise Web Development or equivalent. Course Note(s): Students will be assumed to already have strong Java skills and to be comfortable with IDEs.

EN.605.786. Enterprise System Design and Implementation. 3 Credits.
This course explores enterprise architectures for the development of scalable distributed systems. Effective patterns for distributed data access, MVC-based web tiers, and business logic components are explored as students build complex applications. Factors such as caching and clustering that enable distributed systems to scale to handle potentially thousands of users are a primary focus. In addition, creating a reusable blueprint for an enterprise architecture will be discussed. Applications developed utilizing these concepts are selected from current research topics in information retrieval, data visualization, and machine learning.
Prerequisite(s): EN.605.202 Data Structures; EN.605.784 Enterprise Computing with Java, EN.605.707 Software Patterns, or equivalent experience is recommended.

EN.605.787. Front End Web App Development. 3 Credits.
In this course, we build a real web application for a real client. We will go on a field trip to a selected business with a poor website and you get to see the client interview with identifying components of what you need to get out of that interview. We then go through the mockup process and then code the whole site from scratch, utilizing all the concepts we learn in the course. You will learn to create your own responsive design framework, and write your own media queries. In terms of technologies, we cover CSS in depth, including style conflict resolution, selectors types and how they work together, etc.; Twitter Bootstrap Framework (possibly), including the grid system; dive deep into Javascript, including Javascript DOM manipulation; utilize “raw” Ajax without any libraries to help us; and possibly touch on jQuery. We then take a deep dive into a Javascript framework, concentrating on concepts that are prevalent in a lot of modern front end frameworks. Prerequisite(s): Strong/mature programming skills in any programming language.
Prerequisite(s): EN.605.202 Data Structures; EN.605.682 Web Application Development with Java or equivalent servlet and JSP experience.

EN.605.788. Big Data Processing Using Hadoop. 3 Credits.
Organizations today are generating massive amounts of data that are too large and too unwieldy to fit in relational databases. Therefore, organizations and enterprises are turning to massively parallel computing solutions such as Hadoop for help. The Apache Hadoop platform, with Hadoop Distributed File System (HDFS) and MapReduce (M/R) framework at its core, allows for distributed processing of large data sets across clusters of computers using the map and reduce programming model. It is designed to scale up from a single server to thousands of machines, offering local computation and storage. The Hadoop ecosystem is sizable in nature and includes many subprojects such as Hive and Pig for big data analytics, HBase for real-time access to big data, Zookeeper for distributed transaction process management, and Oozie for workflow. This course breaks down the walls of complexity of distributed processing of big data by providing a practical approach to developing applications on top of the Hadoop platform. By completing this course, students will gain an in-depth understanding of how MapReduce and Distributed File Systems work. In addition, they will be able to author Hadoop-based MapReduce applications in Java and also leverage Hadoop subprojects to build powerful data processing applications. Course Note(s): This course may be counted toward a threecourse track in Data Science and Cloud Computing.
Prerequisite(s): EN.605.202 Data Structures; EN.605.681 Principles of Enterprise Web Development or equivalent Java experience.

EN.605.789. Service API Design and Development. 3 Credits.
This comprehensive course explores core aspects for designing, developing, configuring, securing, deploying, and testing Java-based services and service APIs using modern Spring frameworks and libraries. The focus of this course is on APIs for RESTful services and microservices, and interoperation across application components using APIs. The course also introduces the data exchange mechanism and common data formats, as well as security measures and solutions. At the end of this course, students will be able to apply a variety of techniques and will be able to: Apply best design principles, practices and patterns for creating APIs for RESTful services;Document API using YAML and RAML according to OpenAPI/Swagger specification; Create an API management discipline; Implement API security, control API versioning and life cycle stages; Build RESTful services with Spring Framework; Consume RESTful services using JSON and XML data formats; Integrate RESTful API with different data sources through hands-on coding projects; Build, package and deploy RESTful services on cloud-based platform; Conduct API testing using a variety of tools and techniques; Implement security mechanisms for controlling access to deployed services by service consumers using the Spring Security framework. Students will learn through guided lectures and real-world examples. Students will work on assignments and projects where they will apply newly learned techniques and best practices using the iterative approach of enhancing requested capabilities. Course Note(s): Students will be expected to already have a strong foundation in Java programming and to be comfortable with IDEs tools.
Prerequisite(s): EN.605.644 XML Design Paradigms or equivalent XML design and XML processing experience. EN.605.681 Principles of Enterprise Web Development or equivalent.

EN.605.795. Capstone Project in Computer Science. 3 Credits.
This course permits graduate students in computer science to work with other students and a faculty mentor to explore a topic in depth and apply principles and skills learned in the formal computer science courses to a real world problem. Students will work in self-organized groups of two to five students on a topic selected from a published list. Since students will have selected different courses to meet degree requirements, students should consider the combined strengths of the group in constituting their team. Each team will prepare a proposal, interim reports, a final report, and an oral presentation. The goal is to produce a publication quality paper and substantial software tool. This course has no formal content; each team should meet with their faculty mentor at least once a week and is responsible for developing their own timeline and working to complete it within one semester. The total time required for this course is comparable to the combined class and study time for a formal course. Course prerequisite(s): Seven computer science graduate courses including two courses numbered 605.7xx, all CS foundation courses, and meeting the track requirement; or admission to the post-master's certificate program. Students must also have permission of a faculty mentor, the student's academic advisor, and the program chair. Course note(s): Students may not receive graduate credit for both EN.605.795 and EN.605.802 Independent Study in Computer Science II. This course is only offered in the spring.

EN.605.801. Independent Study in Computer Science I. 3 Credits.
This course permits graduate students in computer science to work with a faculty mentor to explore a topic in depth or conduct research in selected areas. Requirements for completion include submission of a significant paper or project. Prerequisite(s): Seven computer science graduate courses including the foundation courses, three track-focused courses, and two courses numbered 605.7xx, or admission to the post-master's certificate. Students must also have permission of a faculty mentor, the student's academic advisor, and the program chair.

EN.605.802. Independent Study in Computer Science II. 3 Credits.
Students wishing to take a second independent study in computer science should sign up for this course. Prerequisite(s): EN.605.801 Independent Study in Computer Science I and permission of a faculty mentor, the student's academic advisor, and the program chair.
Course Note(s): A student may not receive credit for both EN.605.759 Independent Project in Bioinformatics and EN.605.802.
Prerequisite(s): EN.605.801 Independent Study in Computer Science I and permission of a faculty mentor, the student's academic advisor, and the program chair.