Katz
Katz School
of Science and Health

M.S. in Computer Science

Course Descriptions

Core Course (Agile Program Only)

COM 5000 Introduction to Programming

Learning to write concise, effective, and well-documented computer programs is a prerequisite for any substantive study of computer science. This course introduces students to structured and object-oriented programming constructs, including data types, mathematical and logical operators, control flow constructs, basic data structures, functions, data input and output, objects, classes, methods, inheritance, and algorithmic problem solving. Students are expected to learn to effectively design, execute, and debug algorithms using an object-orientated approach.

Industry Application: Structured and object-oriented programming is a foundational skill for nearly every technical role. In software engineering (e.g., Google, Microsoft, Amazon), developers use OOP principles to design maintainable modules, classes, and interfaces, and rely on debugging and testing to ensure reliable delivery. In data and business analytics (e.g., Airbnb, Uber, Shopify), analysts and data scientists frequently write Python scripts for data cleaning, automation, and API-based data collection, reducing repetitive manual work and minimizing human error. In finance and risk (e.g., JPMorgan, Goldman Sachs, BlackRock), teams build data pipelines and calculation services that require clean, well-documented code and careful handling of edge cases, while in healthcare and research (e.g., Mayo Clinic, Mass General Brigham, NIH), programming enables reproducible workflows for preprocessing, experimentation, and analysis. Increasingly, this work is augmented by AI coding tools such as Claude (especially Claude Code), Cursor, and other LLM-powered pair programmers, supporting applications from rapid prototyping and refactoring to building end-to-end systems that integrate LLM-based services, agentic AI workflows, autonomous coding agents, and API-driven tools into larger software architectures.

COM 5001 Computer Science Math I

Many concepts and theories in modern computer science are built on discrete mathematics, linear algebra, and calculus. This course introduces students to fundamental mathematics for computer science, including topics such as number theory, combinatorics, graph theory, differential, integral and vector calculus for functions of more than one variable as well as basics of linear algebra.

Industry Application: This course provides the mathematical foundation for many areas of modern computing. Discrete mathematics (number theory, combinatorics, and graph theory) supports security, networking, and large-scale systems through rigorous modeling and proof-based reasoning. Linear algebra and multivariable calculus form the core of machine learning, computer graphics, scientific computing, and optimization-based engineering. These skills enable professionals to analyze complex problems, design efficient algorithms, and build reliable, scalable computational solutions across industry.

COM 5002 Algorithms and Data Structures

Most accomplished software designers recognize that a thorough knowledge of data structures and algorithmic analysis can significantly improve application design and performance. This course introduces students to a variety of data structures and algorithmic design paradigms. Students are expected to learn to assess the effectiveness of these structures and algorithms, considering factors such as computational complexity, storage space requirements, ease of use, and maintainability. They also design, develop, implement, and analyze a variety of software applications using data structures and algorithmic design paradigms.

Industry Application: Data structures and algorithms determine whether systems can scale efficiently and respond with low latency. In e-commerce and large online platforms (e.g., Amazon, Walmart, Alibaba), search, recommendation, and advertising systems rely on efficient structures (hash tables, heaps, trees, graphs) and algorithms (sorting, indexing, shortest paths, approximate nearest neighbor search) to handle high traffic. In cloud and distributed systems (e.g., AWS, Google Cloud, Azure), engineers use complexity analysis to optimize caching, scheduling, load balancing, and rate limiting. In social and content platforms (e.g., Meta, TikTok, X), graph algorithms support friend/follow recommendations, community detection, fraud detection, and information diffusion analysis. In robotics and autonomous systems (e.g., Tesla, Waymo, Boston Dynamics), real-time planning and decision-making depend heavily on graph search, dynamic programming, and efficient queue/heap-based implementations. Strong algorithmic thinking also helps professionals justify performance trade-offs and make maintainable engineering decisions.

COM 5010 Computer Systems

Understanding the internal behavior of computing devices enables the design of more efficient and scalable software systems. This course introduces students to the fundamental principles and components of computer system architecture. Students are expected to learn to describe, compare, and contrast different computing architectures and gain a deeper understanding of the relationship between hardware and software. Topics include the evolution of computing systems, digital logic, processor design, instruction sets, x86, x64, ARM, and RISC-V architectures, embedded systems, processor virtualization, graphics processing units (GPUs), and smartphone architecture.

Industry Application: Understanding computer systems and architecture directly improves performance tuning, reliability, and cross-platform development. In AI engineering and high-performance computing (e.g., NVIDIA and major cloud providers), GPU architecture, parallelism, memory hierarchy (cache/DRAM), and vectorization strongly influence training and inference speed—many “algorithmic” bottlenecks are actually bandwidth, caching, or scheduling problems. In mobile and embedded systems (e.g., Apple, Qualcomm, Samsung, ARM ecosystems), knowledge of ARM/RISC-V, power constraints, and hardware resources is essential for system optimization, firmware/driver work, and edge AI deployment. In cybersecurity (e.g., CrowdStrike, Palo Alto Networks), understanding instruction execution, process/memory models, and virtualization helps with vulnerability analysis, malware behavior tracing, and sandboxing. In cloud virtualization (e.g., VMware, AWS, Google), hardware-assisted virtualization, isolation, and I/O paths are critical for diagnosing production issues and performing capacity and performance planning.

COM 5003 Systems Analysis and Design

Effective analysis and design of computing systems and applications requires a variety of business, technological, and methodological considerations. In this course, students explore analytical approaches to the definition of business problems, requirements gathering, process modeling, data modeling, system design, system testing/quality assurance, and system implementation using full system development life cycle. They also explore traditional and emerging project management and application design and development paradigms.

Industry Application: Systems analysis and design bridges business needs and implementable software by covering the full lifecycle: requirements, modeling, design, testing, deployment, and iteration. In enterprise IT and consulting (e.g., Accenture, Deloitte, PwC), business analysts and solution architects conduct stakeholder interviews, model processes (e.g., BPMN), build data models (ERDs), create prototypes, and define acceptance criteria to ensure systems solve real operational problems. In finance, insurance, and regulated industries (e.g., Citi, AIG, healthcare insurers), systems must satisfy compliance, auditing, role-based access control, and traceability—making documentation, test planning, and release governance especially important. In product-driven tech teams (e.g., Google, Meta, Spotify), cross-functional collaboration depends on user stories, use cases, API contracts, and non-functional requirements (latency, availability, scalability) to reduce rework and ambiguity. In healthcare and government projects, where privacy and standards are strict (e.g., HIPAA-related environments), solid analysis and design methods significantly reduce project risk and improve maintainability and long-term adoption.

Advanced Courses and Electives

COM 5100 Advanced Algorithms

Designing efficient algorithms is one of the most important tools computer scientists use to solve difficult problems. This course covers techniques for designing efficient algorithms, as well as advanced topics such as selfadjusting search trees, network flows, linear programming, approximation algorithms, and randomization algorithms. Students apply these tools and techniques to real-world problems, such as airline scheduling, image segmentation, social networking, genomic sequencing, and survey design.

Industry Application: Advanced algorithm design is essential for building systems that scale and perform under real-world constraints. Techniques such as network flows, linear programming, and approximation algorithms are widely used in logistics, transportation, and operations (e.g., airline scheduling, routing, and capacity planning) at companies like UPS, FedEx, and major airlines. Randomized and approximation methods support large-scale data processing and machine learning pipelines where exact solutions are too costly, including ranking, clustering, and segmentation tasks in tech and media platforms. Self-adjusting data structures and efficient search/optimization techniques also appear in high-throughput databases, storage systems, and performance-critical services across cloud providers and fintech.

COM 5101 Theoretical Computer Science and its Applications

The course provides students with a comprehensive understanding of the mathematical aspects of computer science as well as their application. Throughout the course, students will learn the theoretical foundations of computer science and gain knowledge with various topics including algorithms, computational models, and the fundamental principles underlying computation.

Industry Application: Theoretical computer science provides the rigor needed to reason about what can be computed efficiently, and how to build correct and secure systems. Concepts such as computational models, complexity, and correctness help engineers select appropriate algorithms, prove guarantees, and avoid hidden performance pitfalls in large-scale software. Formal reasoning and proof techniques are especially valuable in security, cryptography, and safety-critical domains (e.g., finance, aviation, healthcare), where correctness and robustness are non-negotiable. These foundations also inform modern areas such as programming language design, verification, and trustworthy AI by clarifying limits, trade-offs, and reliability requirements.

COM 5102 Emerging Paradigms in Programming

This advanced-level course explores innovative and cutting-edge programming concepts, languages, and methodologies. The course is designed for students who already possess a strong foundation in traditional programming paradigms and are eager to explore the latest trends and advancements in the field. Throughout the course, students will be exposed to various programming languages, frameworks, and tools to understand how they enable developers to tackle modern-day challenges effectively.

Industry Application: Modern software development increasingly relies on new languages, frameworks, and paradigms to improve scalability, reliability, and developer productivity. Emerging approaches—such as functional and reactive programming, concurrency models, and modern type systems—support robust services in cloud-native and distributed environments (e.g., microservices, event-driven systems, and streaming data). New tooling and frameworks accelerate delivery for web, mobile, and AI applications while improving testing, observability, and maintainability. Understanding these paradigms helps developers evaluate technology choices, adopt modern stacks efficiently, and build systems that meet today's performance, security, and rapid-iteration demands. In parallel, AI software engineering practices are expanding through LLM-based coding agents and agentic AI workflows, where modern coding copilots such as Claude Code and the Cursor editor use large language models to understand entire codebases, generate and refactor code from natural-language instructions, automate tests, scaffold services and APIs, and help design and integrate AI-powered components—such as LLM services and autonomous agents—into larger software systems.

AIM 5001 Data Acquisition & Management

Data Acquisition and Management focuses on the data structures, data design patterns, algorithms, methods, and best practices for the pre-modeling phases of data science workflows, including problem formulation, gather, analyze, explore, model, and communicate. This comprises the “data wrangling” work which is where most data scientists spend most of their time. Because data science is iterative, this preparatory work informs the modeling phase. Often, the creation and validation of new models requires going back for additional data, different data transformations, and exploration of data distributions. In short, every effective data scientist should master analytics programming. Course topics include basics of Python programming and required tools for data management and reading from or writing to databases, text files, and the web; shaping data into “tidy” data frames, exploratory data analysis, data imputations, feature engineering, and feature scaling.

Industry Application: Data acquisition and management skills are essential for building reliable data products. Professionals use these techniques to collect data from databases, files, and APIs; clean and reshape it into analysis-ready formats; and engineer features that improve model performance. This work supports end-to-end analytics workflows in areas such as business intelligence, finance, healthcare, and product analytics.

AIM 5002 Computational Statistics and Probability

Arguably, most of data science is statistical learning, which requires strong foundational knowledge in probability and statistics. And applying computational methods such as direct simulation, shuffling, bootstrapping, and cross validation to statistical problems is often more intuitive and can provide solutions where analytical methods would prove computationally intractable. This course introduces students to the statistical analysis of data using modern computational methods and software. Probability, descriptive statistics, inferential statistics, and computation methods such as simulations sample distributions, shuffling, bootstrapping, and cross-validation will be covered.

Industry Application: Computational statistics enables data scientists to quantify uncertainty and validate conclusions when closed-form analysis is impractical. Simulation, bootstrapping, and cross-validation are widely used for experiment evaluation, forecasting, risk estimation, and model selection. These methods are standard in data-driven decision-making across tech, finance, and healthcare.

AIM 5005 Machine Learning

In classical programming, answers are obtained from rules and data. In machine learning, rules are obtained from data and answers. The widespread availability and sharing of data, and improvements in computing capacity, processing methods, and algorithms have given machine learning the power to deliver game-changing systems and technologies to organizations that compete on predictive, prescriptive, and/or autonomous analytics. In this course, we’ll look at methods for using, tuning, and comparing machine learning algorithms, based on measures of performance, accuracy, and explainability. We’ll also look at recent advances and trends in automated machine learning.

Industry Application: Machine learning (ML) techniques drive predictive and prescriptive analytics across industries by leveraging statistical patterns in data to automate complex decisions. Financial institutions use ML for credit scoring, underwriting, and fraud detection; retailers and logistics firms apply forecasting and optimization for demand and inventory management; and healthcare organizations employ clustering and survival analysis for patient risk and resource planning. Emerging applications include generative and agentic AI systems such as Claude Code, ChatGPT, and Gemini, embodied AI in robots, drones, humanoids, and autonomous vehicles, and vision-language models and coding agents that integrate perception, reasoning, and adaptive decision-making sets to produce new content (such as text, images, or code) in response to a prompt.

AIM 5006 Artificial Intelligence

Artificial Intelligence (AI) is an interdisciplinary field, integrating knowledge and methods from computer science, mathematics, philosophy, psychology, economics, neuroscience, linguistics, and biology. Intelligent agents mimic cognitive functions to implement intelligent behaviors such as perception, reasoning, communication, and acting in symbolic and computational models. AI is used in a wide range of narrow applications, from medical diagnosis to speech recognition to bot control. The autonomous single, multiple, and adversarial agents that students build in this course will support fully observable and partially observable decisions in both deterministic and stochastic environments. Topics covered include search, constraint satisfaction, Markov decision processes, planning, knowledge representation, reasoning under uncertainty, graphical models, and reinforcement learning. The techniques and technologies mastered here will provide the foundational knowledge for the ongoing study and application of AI in other applications across practice areas.

Industry Application: Professional practice in artificial intelligence focuses on the design and deployment of AI systems. Applications span large-scale optimization at companies like Google and Microsoft, generative and reasoning models such as ChatGPT, Claude Code, and Gemini, and embodied AI in robots, drones, humanoids, and autonomous vehicles. In aerospace, defense, and healthcare, organizations like Boeing, Lockheed Martin, and Mayo Clinic apply planning, constraint satisfaction, and probabilistic reasoning for mission planning and clinical decision support. Professionals collaborate closely with domain experts, systems engineers, and risk or compliance teams to ensure reliable and ethical AI deployment. Advanced applications include text and image generators, and Agentic AI systems that are goal-oriented and proactive—capable of planning and executing multi-step tasks autonomously in domains like software development agents, research copilots, autonomous vehicles, robotic systems, and enterprise workflow automation.

AIM 5007 Neural Network and Deep Learning

Data scientists have been able to leverage better algorithms on faster hardware optimized with graphical processing units to deliver improved performance and accuracy in whole classes of applications that had been previously commercially unviable. The biggest beneficiaries are applications that require unstructured data, such as audio and/or video processing. Deep neural networks have also provided gains for other complex applications, from recommendation systems to natural language processing. This course builds on the concepts in machine learning to train multi-layered neural networks. Main topics covered in this course are generalization, convolutional neural network, recurrent neural network, long short-term memory, and autoencoder.

***Industry Application*:** Deep learning powers high-dimensional pattern recognition and autonomous perception across sectors by leveraging GPU-accelerated architectures to derive insights from complex data. In healthcare, convolutional neural networks automate anomaly detection in MRI, CT, and X-ray imaging. Transformers and large language models such as ChatGPT, Claude Code, and Gemini enable advanced NLP, code generation, multimodal reasoning, and conversational systems. Emerging embodied and agentic AI applications include vision-language models, coding agents, and autonomous robots, drones, humanoids, and vehicles. Deep learning also drives retrieval-augmented generation (RAG) with fine-tuned code assistants, next-generation search, marketing, and creative content production using generative and diffusion models.

COM 5014 Special Topics in Computer Science

This course provides the opportunity to offer special interest courses on emerging theory, phenomena, and technologies in computer science, in areas such as systems, human-computer interaction, machine learning, and artificial intelligence. This will be an advanced class, whether seminar style or project based. Students are required to complete an appropriate project or other deliverable in line with the number of credits awarded for the course.

Industry Application: Special-topics courses reflect fast-changing industry needs by focusing on emerging technologies and research directions. Project-based work mirrors real-world R&D and product prototyping, requiring students to evaluate new tools, read current literature, and deliver working implementations. This course is especially relevant for roles in innovation, applied research, and advanced engineering.

COM 5120 Human-Computer Interaction

The rapid expansion of ubiquitous computing means that humans interact with computer technologies in all aspects of their lives. This presents numerous opportunities—and pitfalls—with regards to computer design. This course introduces students to the quantitative and qualitative study of Human Computer Interaction (HCI). We survey various approaches to studying HCI, including Interaction Design, Graphical Design, Educational Design, Human Robot Interaction, and Games. We also consider how the study of HCI influences the design of effective computer technologies.

Industry Application: HCI methods improve product usability, accessibility, and user satisfaction through systematic study and design. Organizations apply qualitative and quantitative research, prototyping, and evaluation to build user-centered interfaces for web, mobile, education, games, and human-robot interaction. These skills are core to UX design, product design, and user research roles.

COM 5210 Mobile Computing and Apps Development

Rapid developments in mobile technologies and systems—like low-cost and energy-efficient CPUs, new applications, increased internet speed, and advances in human-computer interfaces—have made mobile computing an indispensable part of human life. This course provides a broad introduction to the field of mobile computing and mobile application development. Topics include networking, operating systems, database, mobile security, and app development. Students also gain hands-on experience using mobile simulators and apps.

Industry Application: Mobile development skills support building secure, high-performance applications used in daily consumer and enterprise workflows. Knowledge of networking, OS concepts, databases, and mobile security is critical for developing scalable apps and services. Hands-on app development experience aligns with industry practices in testing, deployment, and performance optimization.

COM 5323 Computer Graphics

The course provides students with a comprehensive understanding of the fundamental principles and techniques used in computer graphics. It explores the creation, manipulation, and rendering of digital images and visual content using software and hardware technologies. Throughout the course, students are expected to learn the theoretical foundations of computer graphics and gain hands-on experience with various tools and software used in the field. The course covers both 2D and 3D graphics, enabling students to develop skills in both domains.

Industry Application: Computer graphics techniques enable the creation and rendering of 2D/3D visual content for interactive and immersive experiences. These skills are used in game development, AR/VR, simulation, film and animation, scientific visualization, and UI/UX rendering pipelines. Practical tooling experience supports production work in real-time and offline rendering environments.

COM 5421 DevOps

The course is designed to provide participants with a comprehensive understanding of DevOps principles, practices, and tools. During this course, students will explore the fundamental concepts, methodologies, and technologies that drive the DevOps culture. They are expected learn how to bridge the gap between development and operations teams, enabling faster and more reliable software releases. The course will cover various aspects of the DevOps lifecycle, including continuous integration, continuous delivery, infrastructure automation, and monitoring. Through the course, the students are expected to have a solid understanding of DevOps concepts, tools, and practices, allowing them to contribute effectively to DevOps initiatives within their organizations.

Industry Application: DevOps practices accelerate software delivery while improving reliability and operational quality. Continuous integration and delivery, infrastructure automation, and monitoring are standard in cloud-native engineering and platform teams, preparing students to support scalable deployments, incident response, and high-availability services. Generative AI and large language models are now embedded in DevOps and AI software engineering as coding and operations copilots, using LLM-based agents to generate and optimize CI/CD pipelines, infrastructure-as-code, and tests; analyze logs and metrics for faster incident response; automate runbooks and remediation; and assist with configuration management and environment provisioning.

COM 5440 Software System Security

The course provides students with a deep understanding of the principles, techniques, and processes involved in software system security. The course covers the measures and practices put in place to protect software applications and systems from potential threats and vulnerabilities. It emphasizes multiple layers of protection, each addressing different aspects of security, as software systems are often targeted by malicious actors seeking to exploit weaknesses and gain unauthorized access or control.

Industry Application: Software security practices protect systems against vulnerabilities, attacks, and unauthorized access. Modern AI and CS applications increasingly rely on large language models and AI security tools to support secure design, threat modeling, secure coding, testing, and layered defenses (e.g., AI-assisted code review, dependency scanning, policy checking, and anomaly detection in logs). This knowledge is essential for building trustworthy, resilient products in regulated and high-stakes environments where AI-enabled systems, services, and assistants must be protected against prompt injection, data leakage, model abuse, and traditional software exploits.

COM 5500 Internship

This course allows students to participate in an off-campus internship supervised by a staff person at the internship site and overseen by a faculty advisor. The internship site must be approved by the program director and the overall duration of student work must be no less than 150 hours (based on a 3-credit course). At the start of the internship, the student and faculty advisor will jointly develop specific learning objectives tailored to the nature of the internship. Over the course of the internship, students will be required to submit weekly reflections, and at the end of the internship, students write a final paper that represents the culmination of the work.

Industry Application: Internships provide direct experience with professional workflows, collaboration practices, and real deliverables. Students apply classroom skills to practical projects, strengthen communication through reflections and reporting, and develop job-ready competencies. This course supports career placement by building industry experience and professional networks.

COM 5999 Independent Study in Computer Science

The course provides the student with the flexibility to learn more about a topic of interest outside of the formal course setting. The subject should be chosen in consultation with a faculty advisor who acts as the student's supervisor, and with the permission of the program director. The student is required to submit a course contract describing the course of study and its specific learning objectives. Course credit is determined in advance by the instructor with the approval of the program director.

Industry Application: Independent study develops self-directed learning and specialized expertise aligned with a student's career goals. Students practice defining objectives, conducting technical investigations, and producing concrete deliverables under faculty supervision, including prototypes that integrate large language models, AI coding assistants such as Cursor and Claude Code, and agentic AI components into software systems. This mirrors industry R&D and advanced engineering work where teams explore new methods in AI-powered software engineering, experiment with LLM-based services and autonomous agents, and build proof-of-concept applications and tools.

MAN 5580 Project Management

Big-Tech development and management is project-based, and successful researchers and technologists are effective at managing projects and collaborating in cross-functional, geographically distributed project teams. This course teaches the methodologies and tools for large-sized (PMI) and small-scale (Agile) projects as well as how to adapt management methods to organizational culture and project team members' background and experience.

Industry Application: Project management is essential for delivering complex technical work on time and within scope. Agile and PMI methods are widely used to plan milestones, manage risk, coordinate cross-functional teams, and communicate progress to stakeholders. These skills are critical in software development, data initiatives, and research-to-product execution.

COM 6000 Capstone in Computer Science I

The Capstone in Computer Science integrates students' prior coursework, research, colloquia, and professional experiences. It offers a unique opportunity to synthesize computer science theory with real-world practice through an applied project, thesis, approved internship, or equivalent activity. Students will work with their supervising faculty to identify deliverables for both Part 1 and Part 2 of the Capstone.

Industry Application: This capstone mirrors how real-world engineering and research work is executed: defining a problem, scoping requirements, planning milestones, and producing an initial set of deliverables under supervision. Students practice translating ambiguous needs into clear specifications, selecting appropriate tools and architectures, and documenting decisions in a professional format. The work emphasizes early-stage project execution—proposal/thesis framing, literature or competitive review, data or system design, and a proof-of-concept—skills commonly expected in software engineering, data science, and applied research roles. It also strengthens collaboration and stakeholder communication through regular check-ins, progress reporting, and iterative refinement.

COM 6001 Capstone in Computer Science II

The Capstone in Computer Science integrates students' prior coursework, research, colloquia, and professional experiences. It offers a unique opportunity to synthesize computer science theory with real-world practice through an applied project, thesis, approved internship, or equivalent activity. Students will work with their supervising faculty to identify deliverables for both Part 1 and Part 2 of the Capstone.

Industry Application: Capstone II aligns with the later stages of industry projects: implementation at production quality, rigorous evaluation, and final delivery. Students integrate testing and validation, performance and reliability considerations, security and ethics where relevant, and clear documentation so others can reproduce or maintain the work. The final deliverable—whether a deployed application, research thesis, or internship-based outcome—demonstrates end-to-end capability from design through execution, similar to what employers assess in portfolios and technical interviews. By completing a substantial project with measurable outcomes and a polished presentation, students develop job-ready experience in shipping solutions, defending technical choices, and communicating results to both technical and non-technical audiences.

Updated: January 2026