Course Descriptions

50101 Concepts of Programming
An introduction to programming — no prior programming knowledge or experience is required. The goal is to familiarize students with the fundamental components of structured programming — memory management, algorithms, data structures, modular design, etc. The course emphasizes fundamental concepts and elementary program design.

50103 Math for Computer Science
This course is an introduction to ideas and techniques from discrete mathematics that are widely used in computer science. Topics include: propositional logic and quantifiers, proof methods (direct proof, proof by contradiction, case analysis, mathematical induction), basic number theory (GCD, Euclid’s algorithm, prime factorization, modular arithmetic), summations and closed forms, growth of functions and asymptotic notation, basics of counting (permutations, combinations, binomial theorem, pigeonhole principle), recurrences and methods of solving linear recurrences, basic graph theory (connectedness, trees, bipartite graphs), discrete probability, conditional probability and independence, random variables, expected value, and variance.

51025 Practicum in Trading System Development
This course will focus on the operation of exchange trading systems at a high level, real world computational issues in trading systems and their communication with exchanges, and the implementation details in actually developing an exchange trading system in computer software. The class will have lectures that focus on the technological strategies in implementation, along with lectures that focus more on relevant system functionality, as well as guest lectures from industry leaders who will offer a greater breadth in the overall landscape of trading systems and exchange systems operations. Specifically, this course is designed to give students hands-on implementation experience in designing and building a functioning trading system in C and C++ using state-of-the-art tools and environments. Students will work collaboratively in developing an exchange platform that implements the fundamental lifecycle of client-to-exchange communication, namely: Order Receipt, Order Matching, Market Data Broadcast, Order Book Management Strategy, and Trade Notification.

51030 iOS Application Development
This course will instruct students on the fundamentals of mobile application development using the iOS SDK. An introduction to object-oriented design using the model-view-controller pattern, memory management, Objective-C programming language will also be taught. Using iOS APIs and tools, including Xcode, Interface Builder and Instruments, students will complete weekly programming assignments that will culminate in the development of a fully functioning iOS application. The course will also explore interface and application design considerations specific to mobile technologies. As a final project, each student will design and implement an application an application of their choice.

51031 Android Application Development
After a quick introduction to mobile computing, competing platforms, Android architecture, market projections, and social and economic implications, we will dive directly into developing several reference implementations. Alternating between theory and practice, and progressing cumulatively, we will cover every major feature of the Android platform, including: audio, graphics, internet connectivity, wifi, mapping/geo-positioning, notifications, sms, structured feeds, persistence, threads, states, and inter-process communication, among others. Students will choose a final project, then envision, design, develop, test, and deploy an application to the Android marketplace.

51032 Advanced iOS
Advances in mobile technologies are changing the way that individuals and businesses use computing devices. This course will explore real-world issues with developing robust, high-performance iOS applications for iPhone, iPod Touch and iPad. The course will consist of lectures, hands-on coding exercises and discussion. Weekly programming assignments will be used to create a portfolio of applications using advanced iOS APIs and tools, such as Xcode, Interface Builder and Instruments.
51036 Java Programming
A fast-paced first course in Java for students with some prior programming experience, though not necessarily in an object-oriented language. A strong emphasis will be placed on understanding basic fundamentals of OO design — inheritance, polymorphism, composition, etc., and more generally on applying sound principles of modern software engineering to real-world problems. In the latter half of the course more advanced OO design patterns will be studied in the context of certain Java libraries (e.g., Swing). However, the main focus will be on using the “core language” to write good software rather than a detailed study of particular high-level libraries.

51037 Advanced Java Programming
This is an advanced course designed for students with a good foundation in Java programming. Basic familiarity with C is also assumed. The course focuses on designing distributed, multithreaded applications with the Java platform. It is an application programming course. Emphasis is placed on applying technology rather than studying API design and implementation. Topics proceed (roughly) from “low-level” to high level network programming concepts: socket byte streams, object serialization, Remote Method Invocation, Java/CORBA (minimal), Web Services, and (briefly) Enterprise Java Beans. While any of these topics alone could form the basis for an entire course, the emphasis is on providing students with an adequate foundation for pursuing individual topics in greater depth. Along the same lines, a major focus of the course is to help students determine when to best apply a given Java technology in a real world, multi-tier application.

51040 C Programming
This is an accelerated introduction to the C Programming Language designed for students with prior programming experience. C is in many ways the lingua franca of computing, and a broad range of programming languages and related technologies derive from the basic principles of C memory management, control flow, and abstraction. Though there are many subtleties, C is not a big language, and it is expected that students will leave the course with a relatively in-depth understanding of the key concepts, which will then form a solid foundation for studying higher-level technologies. At the same time, C itself remains a very practical language, particularly so in areas such as scientific programming, high-performance computing, application level library design, systems programming, network programming, multithreaded programming, etc. Students who successfully complete the course will be well prepared for subsequent MPCS courses in these areas. The course studies both fundamental and advanced C language constructs in the abstract and reinforces them through a range of exercises in the design of basic and advanced data structures, rudimentary algorithms, and API design.

51044 C++ for Advanced Programmers
This course begins with a rapid, disciplined overview of C++. Students will learn the effective use of the standard template language, and will pass from there to cover object-oriented features of the language. After mastering language basics, we will go over the language specification in greater detail, covering advanced language features and programming techniques.

51045 Advanced C++
This course is a continuation of MPCS 51044, in which we dive deeper into advanced C++ techniques and constructs, including functors, binders, type traits, and cache-aware programming. The second half of the course develops an extended look at using template metaprograms to implement common Design Patterns, allowing programs to achieve dramatically higher performance while at the same time increasing abstraction and maintainability.

51050 OO Architecture: Patterns, Technologies, and Implementations
This course gives hands-on experience in object oriented architecture and design and the understanding of such designs in the form of patterns. The course is designed to give students a solid introduction to software patterns as they are implemented in large-scale system architectures currently in industry, as well as the opportunity to play with various languages in implementing these patterns. Students will be encouraged to explore various implementation possibilities afforded by these patterns. Trade-offs in performance, development time and maintenance impact, to name a few, will be discussed.
51081 Unix System Programming  
This is a hands-on lab-based course and examines the development of software, from a systems perspective, on a Unix system. It focuses on the tools used to develop systems software on Unix, with specific emphasis on the systems programming API.

51083 Cloud Computing  
This course provides an introduction to cloud computing with specific consideration for application development in two contexts: highly scalable (or so-called “web-scale”) web applications, and enterprise applications in a hybrid environment comprising both on-premises and cloud infrastructure. We will focus primarily on infrastructure and platform services, and will introduce software-as-a-service from the perspective of a consuming application. The course will emphasize practical applications of cloud computing technologies, with sufficient exploration of their theoretical underpinnings to inform architectural, design, and implementation decisions.

51087 High Performance Computing  
This is a fast-paced applied programming course aimed at students with significant development experience in either C, C++, or FORTRAN (Java is also possible, but not ideal). A brief overview of parallel computing will be presented at the outset, but the course will be less on overview of HPC architectures and much more a focus on algorithmic implementation and performance tuning. The goal of the course it to give students experience in developing efficient, scalable (distributed memory) parallels algorithms appropriate for any system running an implementation of the Message Passing Interface (MPI). Assignments will be designed with some flexibility to allow students to explore applying parallel techniques to applications in their own field of interest. After an introduction to MPI, some of the topics covered will include: parallel solutions of linear equations, sorting, interpolation, and integration of ODE’s and PDE’s.

51100 Advanced Programming  
This class is intended for students who are joining the program with an existing degree in Computer Science, or with substantial experience in programming. This course will be taught primarily in C, including an accelerated introduction to the C language for students who have not used C before. The course will cover advanced data structures and topics in concurrent and multicore programming not covered in the Java Programming or C Programming courses.

51200 Introduction to Software Engineering  
Writing first-class software requires top-notch architecture, design and coding skills, but successful software project execution—from identifying the need to providing support—depends on many factors besides technical prowess. This course surveys the key practices and processes that help ensure successful projects. It provides an introduction to central activities of software engineering other than just coding, such as planning, requirements, testing and management. It balances this discussion of typical engineering activities against the development process models in which they take place – specifically, it addresses the tension between traditional plan-driven approaches and adaptive agile techniques. By examining the underlying principles of major development models, it shows how those principles address (or fail to address) the various problems encountered by project teams. Students who complete this course will gain a solid understanding of both plan-driven and agile software development principles and how to negotiate between them in different contexts.

51210 Advanced Topics in Software Architecture and Design  
This course provides hands-on experience in the architecture and design of object-oriented software systems and a review of best practices for the communication of that design. Issues in the landscape of software design, including complexity, constraints, progressive discovery, and limitations in communication will be explored. Specific topics and projects will modulate year by year, but generally, our discussion of software design will be understood to operate within the object-oriented paradigm.

51300 Compilers  
This class teaches the theory and practice of how to write a compiler, including lexical analysis, grammars, lexers and parsers, type checking, and code generation. For decades, compilers have been the most dynamic and challenging branch in computer science. The main part of this class will focus on providing the basics of the different phases of compilation. Through the course, students will develop appreciation for the implementation
strategies behind making an efficient and robust compiler.

52010 Computer Architecture
This class will introduce students to the architectural knowledge they need to write high performance software for modern systems. It will start with a review of modern computer architecture and give the programmer and system architect a thorough grounding in modern computer architecture. The second part of the class will focus on creating a cloud driven distributed microcontroller project. Students will use the wi-fi enabled Electric Imp microcontroller and write cloud server software and microcontroller software, which will interact to create a distributed system for the management of local devices over wi-fi and the Internet. The class will also discuss the special security issues associated with such distributed systems.

52011 Intro to Computer Systems
This course covers machine organization including details of individual units and their interaction with each other. The course includes machine and assembly language concepts, data representation and management of data and control flow in a computer, from a programmer’s perspective.

52030 Operating Systems
This is an introductory course on operating systems. Students will learn the fundamentals of how modern operating systems are built, from the interface with hardware up through the kernel-userspace boundary. Important topics include the relationship between processes and threads, synchronization, inter-process communication, memory management, file systems, scheduling, and I/O.

52553 Web Development
This course provides students with an introduction to modern web development, with an emphasis on the pragmatic skills needed to build live, functioning web applications. Students will learn fundamental domain modeling skills, HTML and CSS frameworks, agile software techniques and best practices, Javascript and AJAX, and both server-side and client-side debugging techniques. We will use the Ruby language and the Rails framework to immerse students into the challenge of building a live, database-backed web application deployed at a public web address.

52554 Advanced Web Development
This course builds upon MPCS 52553 to enable students to gain mastery over modern web architectures and services. Today’s consumer-facing and business applications must consume external services and publish services of their own. Students will build interconnected chains of services, with a particular emphasis on efficiency, security, and sustainability using modern web frameworks such as Rails, React, Node, and more.

53001 Databases
The objective of this course is to introduce you to the principles and techniques of relational databases. The course focuses on three broad topics: modeling, implementation, and operation environment.

53003 Advanced Databases
The objective of this course will be to (i) expand the knowledge by covering new topics that represent the state-of-the-art in database management systems and distributed systems, and (ii) to build upon foundations developed in MPCS 53001 - Databases by covering topics in greater depth. Topics covered include: distributed databases, distributed transactions, NoSQL databases, data warehousing, alternative data storage & models, graph databases, object-relational databases and cloud databases, geospatial databases, GIST indexes in PostGIS, space filling curves, and geohashing.

53013 Big Data
In this course, we will cover both the theory and practice of Big Data. To support practical experience with genuinely big data, we have arranged that all students will receive a substantial credit on the Google Cloud Platform and App Engine courtesy of generous support from Google. We will use this to implement a much extended version of the tutorial at http://goo.gl/P9DB8. To develop a sound understanding of the theory of Big Data, we will use Marz and Warren’s Big Data textbook providing a conceptual architecture for Big Data systems. We will also cover important additional topics that invariably arise in real world applications of Big Data, such as like cleaning scraped data meant for human consumption to meet the needs Big Data systems.
53110 Foundations of Computational Data Analysis
Introduces the fundamental concepts and techniques in computer science and statistics used in modern data analysis. The course covers statistical methods for summarizing and exploring data sets, for making inferences about a population from a given sample, and for testing hypotheses. It examines in detail techniques from machine learning used for solving fundamental problems in data mining: classification, prediction, clustering, association analysis, and data reduction. Students use Python for implementing algorithms and Python libraries such as NumPy, SciPy, matplotlib, and pandas for analyzing real-world data sets.

53111 Machine Learning
This course introduces the fundamental concepts and techniques in data mining, machine learning, and statistical modeling, and the practical know-how to apply them to real-world data through Python-based software. The course examines in detail topics in both supervised and unsupervised learning. These include linear and logistic regression and regularization; classification using decision trees, nearest neighbors, naive Bayes, boosting, random trees, and artificial neural networks; clustering using k-means, expectation-maximization, hierarchical approaches, and density-based techniques; and dimensionality reduction through PCA and SVD. Students use Python and Python libraries such as NumPy, SciPy, matplotlib, and pandas for for implementing algorithms and analyzing data.

53112 Advanced Data Analytics
This course explores selected advanced themes in data mining and analytics. These include the recent “model-free” techniques for mining massive datasets, foundations of natural language processing, and time series analysis. Topics include frameworks such as MapReduce; algorithmic ideas such as locality-sensitive hashing, Bloom filters, random walks, and competitive analysis; and applications such as link analysis, social-network analysis, recommendation systems, streaming data, and advertising on the web. In natural language processing, the course introduces fundamentals of language models, text classification, and information retrieval and extraction. In time series analysis, the course examines stationary processes and the ARIMA and GARCH models.

54001 Networks
Broadly, this course will focus on the history, theory and implementation of computer networks. We will discuss the low-level technologies that move bits around (such as Ethernet and WiFi), the high-level applications that are part of our everyday 21st-century lives (such as email, the Web, and mobile phones), and everything in between (security, TCP/IP). At the completion of this quarter, students should be able to explain, in detail, how data makes its way around the Internet when you click on a web link, how you can drive around at 80 MPH talking on a cell phone without the call dropping, and how you can make a streaming video call over a lousy wireless link without frame dropping or jitter. In short, we’ll pull back the curtain on what can be a somewhat mysterious and magical part of working with computers.

55001 Algorithms
This course is an introduction to the design and analysis of efficient algorithms. Topics include basic algorithm design techniques such as divide-and-conquer methods, dynamic programming, greedy choice, and graph searching. Sorting and searching algorithms are studied, as well as graph algorithms such as graph search, minimal spanning trees, and shortest paths.

55005 Advanced Algorithms
This course introduces students to advanced techniques for designing and analyzing algorithms, and explores their use in a variety of problem areas. Emphasis is placed on fundamental algorithms and techniques of algorithm design.

56420 Bioinformatics for Computer Scientists
This course aims to introduce computer scientists to the field of bioinformatics. The vast amounts of data produced in genomics related research has significantly transformed the role of biological research. High-throughput automated biological experiments require advanced algorithms, implemented in high-performance computing systems, to interpret their results. This course will focus on analyzing complex data sets in the context of biological problems. Students will design and implement systems that are reliable, capable of handling huge amounts of data, and utilize best practices in interface and usability design.
to accomplish common bioinformatics related problems. While this course should be of interest for students interested in biological sciences and biotechnology, techniques and approaches taught will be applicable to other fields.

56513 Digital Forensics
In this course we will cover processes for investigations and evidence handling, types of evidence available, tools used in forensic investigations, recovery and preservation of data, and other forensic processes used in system incident response. We will use hands-on approaches with a number of tools and document results.

56515 Computer and Network Security
The objective of this course is to provide a basic understanding of Information Technology security, and to build an understanding of the elements that should be in place for an IT environment to achieve an adequate security level. We will begin with a general overview of IT security and introduce a framework for addressing security needs across an enterprise. Major security objectives and mechanisms for attaining these objectives will be discussed, including cryptography, authentication systems, Public Key Infrastructure, and platform and network security mechanisms. This course will give an overview of the technical details involved in the platform and network levels of security. We will look at common TCP/IP applications and discuss their security vulnerabilities. The course material will be presented in a framework of understanding business risks and how to address them.

58001 Numerical Methods
This is a practical programming course focused on the basic theory and efficient implementation of a broad sampling of common numerical methods. Each class of numerical methods will be introduced conceptually followed by detailed exercises focused on both prototyping them (using matlab) and programming them efficiently on modern (serial) architectures. The ideal student in this course would have a strong interest in the use of computer modeling as predictive tool in a range of disciplines — for example risk management, optimized engineering design, safety analysis, etc. The numerical methods studied in this course underlie the modeling and simulation of a huge range of physical and social phenomena, and are being put to increasing use in a range of industrial applications. After successfully completing this course, a student should have the necessary foundation to quickly gain expertise in any application-specific area of computer modeling.

58020 Time Series Analysis and Stochastic Processes
Stochastic processes are driven by random events. They can be used to model phenomena in a broad range of disciplines, including science/engineering (e.g. computational physics, chemistry, and biology), business/finance (e.g. investment models and operations research), and computer systems (e.g. client/server workloads and resilience modeling). In many cases relatively simple stochastic simulations can provide estimates for problems that are difficult or impossible to model with closed-form equations. In this class we focus on the rudimentary ideas and techniques that underlie stochastic time series analysis, discrete events modeling, and Monte Carlo simulations. Course lectures will focus on the basic principles of probability theory, their efficient implementation on modern computers, and examples of their application to real world problems. Upon completion of the course, students should have an adequate background to quickly learn in depth specific Monte Carlo approaches in their chosen field of interest.