CU Boulder Applied Computer Science Program

Program Overview

The following are courses that are apart of the CU Boulder Applied Computer Science program. The full curriculum can be found **here**. Core course codes and classes are colored with yellow, elective course codes and classes are colored with gray, credit hours are colored with cyan, completed courses are colored with green, in progress courses are colored with orange, and scheduled courses are colored with magenta.

Course Name	Course Code	Credit Hours	Completion Status
Computer Science 1: Starting Computing	**CSPB 1300**	4	Completed - TA
Computer Science 2: Data Structures	**CSPB 2270**	4	In Progress - SM23
Computer Systems	**CSPB 2400**	4	Scheduled
Discrete Structures	**CSPB 2824**	3	Scheduled
Algorithms	**CSPB 3104**	4	Scheduled
Principles of Programming Languages	**CSPB 3155**	4	Scheduled
Software Development Methods & Tools	**CSPB 3308**	3	Scheduled
Linear Algebra W/ Computer Science Applications	**CSPB 2820**	3	To Be Determined
Introduction to Data Science W/ Probability & Statistics	**CSPB 3022**	3	To Be Determined
Introduction to Artificial Intelligence	**CSPB 3202**	3	To Be Determined
Design & Analysis of Database Systems	**CSPB 3287**	3	To Be Determined
Cognitive Science	**CSPB 3702**	3	To Be Determined
Design & Analysis of Operating Systems	**CSPB 3753**	4	To Be Determined
Information Visualization	**CSPB 4122**	3	To Be Determined
Data Mining	**CSPB 4502**	3	To Be Determined

Clicking on the **Course Name** description will take you to the course description found on CU's website. Clicking on the **Course Code** link will take you to the course description found in this document.

Program Schedule

The following schedule for the program is as follows.

Course	Course Code	Course Hours	Schedule
Computer Science 1: Starting Computing	**CSPB 1300**	4	Completed - TA
Computer Science 2: Data Structures	**CSPB 2270**	4	In Progress - SM23
Discrete Structures	**CSPB 2824**	3	Scheduled - F23

CSPB 1300: Computer Science 1: Starting Computing - Prerequisites: **N/A** - Credits: **4**

Brief Description of Course Content - The course covers techniques for writing computer programs in high level programming languages to solve problems of interest in a range of application domains. This class is intended for students with little to no experience with programming.

Specific Goals - By the end of this course, students should be well positioned to learn any mainstream programming language, and have a foundation for learning more advanced concepts for software engineering and computer science.

Specific Outcomes of Instruction

- Understand how to break down hard problems into a series of sub-problems.
- Be able to use fundamental programming constructs (such as variables, conditional and iterative control structures) in Python and C++.
- Understand and be able to implement simple input and output (I/O) (e.g. interactive input from the user, or using disk storage).
- Design functions and reason about their role in programs, including an understanding of passing arguments and returning values.
- Learn the properties of data types, including primitive types like numbers and booleans, as well as complex data types like lists and dictionaries.
- Use an Integrated Development Environment (IDE) to write code. Begin to understand the art of debugging as part of software development.
- Design and create code using the fundamentals of object-oriented design methods.
- Develop an understanding of software development as a dynamic, social process, and that learning how to seek out information is a necessary skill for success.
- Leverage two different programming languages to understand programming concepts in general rather than just in the particular.
- Understand type systems (dynamic vs static).
- Know the differences between interpreted and compiled languages.

Brief List of Topics to be Covered

- Python Basics
- Debugging
- Modules and Functions
- Selection
- Iterable Data Structures
- Classes and Objects
- Intro to C++ & C++ program composition

Mathematic Concepts Used

- Basic Algebra
- Modulo

CSPB 2270: Computer Science 2: Data Structures - Prerequisites: **CSPB 1300** - Credits: **4**

Brief Description of Course Content - Studies data abstractions (e.g., stacks, queues, lists, trees) and their representation techniques (e.g., linking, arrays). Introduces concepts used in algorithm design and analysis including criteria for selecting data structures to fit their applications.

Topics include data and program representations, computer organization effect on performance and mechanisms used for program isolation and memory management.

Specific Goals

Specific Outcomes of Instruction

- Document code including precondition/postcondition contracts for functions and invariants for classes.
- Determine quadratic, linear and logarithmic running time behavior in simple algorithms, write big-O expressions to describe this behavior, and state the running time behaviors for all basic operations on the data structures presented in the course.
- Create and recognize appropriate test data for simple problems, including testing boundary conditions and creating/running test cases, and writing simple interactive test programs to test any newly implemented class.
- Define basic data types (vector, stack, queue, priority queue, map, list).
- Specify, design and test new classes using the principle of information hiding for the following data structures: array-based collections (including dynamic arrays), list-based collections (singly-linked lists, doubly-linked lists, circular-linked lists), stacks, queues, priority queues, binary search trees, heaps, hash tables, graphs (e.g. for depth-first and breadth-first search), and at least one balanced search tree.
- Be able to describe how basic data types are stored in memory (sequential or distributed), predict what may happen when they exceed those bounds.
- Correctly use and manipulate pointer variables to change variables and build dynamic data structures.
- Determine an appropriate data structure for given problems.
- Follow, explain, trace, and be able to implement standard computer science algorithms using standard data types, such as a stack-based evaluation of arithmetic expressions or a traversal of a graph.
- Recognize situations in which a subtask is nothing more than a simpler version of the larger problem and design recursive solutions for these problems.
- Follow, explain, trace, and be able to implement binary search and a variety of quadratic sorting algorithms including mergesort, quicksort and heapsort.

Brief List of Topics to be Covered

- Cost of algorithms and Big O notation.
- Memory and pointers, structs, and dynamic memory allocation.
- Linked lists, stacks and queues.
- Trees: Binary trees, binary search trees, tree traversal, recursion.
- Tree balancing: red-black trees.
- Graphs: graph traversal algorithms, depth-first and breadth-first search.
- Hash tables, hash functions, collision resolution algorithms.
- Algorithms for sorting, such as insertion sort, bubble sort, quick sort, and merge sort.

Mathematic Concepts Used

- Logarithms

- Big O
- Recursion
- Trees
- Graphs

CSPB 2400: Computer Systems - Prerequisites: **CSPB 2270** - Credits: **4**

Brief Description of Course Content - Covers how programs are represented and executed by modern computers, including low level machine representations of programs and data, an understanding of how computer components and the memory hierarchy influence performance.

Topics include data and program representations, computer organization effect on performance and mechanisms used for program isolation and memory management.

Specific Goals

Specific Outcomes of Instruction

- Explain and perform common logical operations (and, or, negation, conversion) on binary variables and binary vectors and identify and apply common boolean algebraic laws such as DeMorgan's laws, idempotence, etc.
- An ability to translate between integer binary and decimal data, detect and identify the outcome of operations due to limited data representations (e.g. overflow), distinguish between the data representations and ranges for signed and unsigned data types.
- Translate IEEE floating point representation to and from binary and real numbers and identify the limitations of fixed-precision floating point representation.
- An ability to related compiler-generated assembly programs to the corresponding higher level language structures with sufficient ability to enable debugging high level programs. Given a machine language representation of a program compiled in a higher level language, students should be identify and describe the operation of conditional statements, loops, function calls, switch statements.
- The ability to explain how higher level language functions are implemented using the stack of an underlying machine, including how local variables are allocated, trace the execution due to recursion and identify and trace the effect of buffer-overflow of the stack.
- An ability to explain how high level program structures can be restructured to facilitate optimization for pipelined architectures and cache memory hierarchies.
- An ability to explain how computer memory is organized and represented both to the programmer and to the computer architecture by the operating system through the use of virtual memory mapping.
- An understanding of how to use asynchronous signals, concurrent programs and the programming issues that arise with such programs, such as race conditions.
- Identify and construct processes on a common computer platform, identify and perform basic synchronization between processes and understand the costs and benefits of using processes.
- An ability to explain how global memory, function-local and dynamic memory allocation is performed and the performance benefits of each form of memory allocation.
- An ability to explain how programming errors may affect program correctness, including errors in function calls, memory allocation, integer and floating point data representations.
- An ability to measure program performance and use that measured information to determine how to improve program performance.
- An ability to use a machine-level debugger and inspect the memory and register state of programs.

Brief List of Topics to be Covered

- Vectors
- Linear functions
- Number representation in computers
- Program representation

- Computer security: stack overflows and code injection
- Computer organization and its impact on computer performance
- Memory hierarchy and its impact on computer performance and security
- Cache organization
- Processes, exceptions and signals
- Virtual and dynamic memory management
- Linking and loading programs

Mathematical Concepts Used

- Boolean Logic
- Binary

CSPB 2824: Discrete Structures - Co-Requisites: **CSPB 1300** - Credits: **3**

Brief Description of Course Content - The course covers fundamental ideas from discrete mathematics, especially for computer science students. It focuses on topics that will be foundational for future courses including algorithms, artificial intelligence, programming languages, theoretical computer science, computer systems, cryptography, networks, computer/network security, databases, and compilers.

Specific Goals

Specific Outcomes of Instruction
We will build on the following 6 primary learning goals throughout the term:

- Understand and construct logical arguments and proofs using formal logic, truth tables, and proof techniques.
- Understand and use the basics structures of sets, functions, sums and matrices.
- Use and understand algorithms, number theory and cryptography.
- Demonstrate and make arguments using counting, and probability.
- Use, develop, and analyze formal relations, and graph theory.
- Develop the skills of "Mathematical Maturity" including:
  - The capacity to generalize from a specific example to broad concept.
  - The capacity to handle increasingly abstract ideas.
  - A significant shift from learning by memorization to learning through understanding.
  - The ability to recognize mathematical patterns and think abstractly.
  - Read, write and critique formal proofs.
  - Teach yourself and fill in missing details.

Brief List of Topics to be Covered

- Logic
- Proof techniques
- Algorithms
- Modular Arithmetic
- Number theory
- Cryptography
- Induction
- Combinatorics
- Probability
- Bayes Thm
- Relations
- Graphs

Mathematical Concepts Used

- Basic Algebra
- Program Entry Requirements

CSPB 3104: Algorithms - Prerequisites: **CSPB 2270** & **CSPB 2824** - Credits: **4**

Brief Description of Course Content - Covers the fundamentals of algorithms and various algorithmic strategies, including time and space complexity, sorting algorithms, recurrence relations, divide and conquer algorithms, greedy algorithms, dynamic programming, linear programming, graph algorithms, problems in P and NP, and approximation algorithms.

Specific Goals

Specific Outcomes of Instruction

- Understanding properties of algorithms.
- Proving these properties mathematically.
- Proving rigorous time and space complexity bounds on the performance.
- Understand the relative merits or demerits of an algorithm, in practice.
- Use algorithms to solve core problems that may arise inside applications.
- Learn key tricks (motifs) underlying the design of new algorithms for emerging applications.

Brief List of Topics to be Covered

- Introduction to Algorithms: Complexity analysis
- Divide and Conquer Algorithms
- Sorting and Order Statistics
- Advanced Data Structures: heaps, balanced trees and hash-functions
- Dynamic Programming
- Greedy Algorithms
- Graph Algorithms: Search, Minimum Spanning Trees, Shortest Paths, Network Flows
- Introduction to Linear and Integer Programming
- Basics of Computational Complexity: P, NP, reductions and open problems

Mathematical Concepts Used

- Recursion
- Logarithmic
- Exponential Functions
- Induction
- Graph Theory

CSPB 3155: Principles of Programming Languages - Prerequisites: **CSPB 2270** & **CSPB 2824** - Credits: **4**

Brief Description of Course Content - Study fundamental concepts on which programming of languages are based, and execution models supporting them. Topics include values, variables, bindings, type systems, control structures, exceptions, concurrency, and modularity. Learn how to select a language and to adapt to a new language.

Specific Goals

Specific Outcomes of Instruction

- Learn new programming languages quickly
- Choose the language for a programming task
- Write pure functional code
- Write new languages or APIs with clear semantics
- Read and write context-free grammars and parsers

Brief List of Topics to be Covered

- Scala
- Javascript
- Program Semantics
- Context-free grammars
- Recursion and higher-order functions
- Algebraic Data Types
- Expression Trees
- Type checking
- Mutable State
- Scope, bindings, and closures
- Currying
- Callbacks and Continuation-Passing Style

Mathematical Concepts Used

- Regular Expressions
- Context-Free Grammars
- Proofs About Program Properties
- Recursion and Induction

CSPB 3308: Software Development Methods & Tools - Prerequisites: **CSPB 2270 -** Credits: **3**

Brief Description of Course Content - Covers tools and practices for software development with a strong focus on best practices used in industry and professional development, such as agile methodologies, pair-programming and test-driven design. Students develop web services and applications while learning these methods and tools.

Specific Goals

Specific Outcomes of Instruction

- Learn and use new software development tools; understand technical documentation for software tools
- Work in small, distributed groups on software projects
- Lead Agile development teams
- Write functional web applications
- Use distributed version control fluently, including merging and branching
- Write unit tests and use test-driven design to build software
- Compose SQL queries to access data
- Write clear and helpful documentation

Brief List of Topics to be Covered

- Unix shell
- Shell Scripting
- Regular Expressions
- Agile Development Methods
- Makefiles and Build tools
- Unit Testing
- HTML, CSS, and Javascript
- SQL
- Cloud Computing
- Web Services
- Platform as a Service (PaaS)

Mathematical Concepts Used

- Regular Expressions

CSPB 2820: Linear Algebra W/ Computer Science Applications - Prerequisites: **CSPB 2824** - Credits: **3**

Brief Description of Course Content - Introduces the fundamentals of linear algebra in the context of computer science applications. Includes vector spaces, matrices, linear systems, and eigenvalues. Includes the basics of floating point computation and numerical linear algebra.

Specific Goals - By the end of this course, students should be well positioned to apply linear algebra skills in a computer science context.

Specific Outcomes of Instruction

- Use and reason about vectors, theoretically and in computer science applications
- Use and reason about matrices, theoretically and in computer science applications
- Understand and apply linear functions, and the relation between linear functions and matrices
- Solve systems of linear equations, and reason about the computational complexity of them

Brief List of Topics to be Covered

- Vectors
- Linear functions
- Norm and distance
- Writing linear algebra code
- Clustering
- Linear independence
- Matrices
- Matrix examples
- Linear equations
- Linear dynamical systems
- Matrix multiplication
- Matrix inverses
- Least squares
- Eigenvalues, eigenvectors, and singular values
- Least squares data fitting
- Least squares classification

Mathematical Concepts Used

- Algebra
- Proofs
- Real Numbers
- Inequalities
- Linear Algebra

CSPB 3022: Introduction to Data Science W/ Probability & Statistics - Prerequisites: **CSPB 1300** - Credits: **3**

Brief Description of Course Content - Introduces students to the tools methods and theory behind extracting insights from data. Covers algorithms of cleaning and munging data, probability theory and common distributions, statistical simulation, drawing inferences from data, and basic statistical modeling.

Specific Goals

Specific Outcomes of Instruction

- Recognize the importance of data collection, identify limitations in data collection methods and other sources of statistical bias, and determine their implications and how they affect the scope of inference.
- Use statistical software to summarize data numerically and visually, and to perform data analysis.
- Have a conceptual understanding of the unified nature of statistical inference.
- Apply estimation and testing methods to analyze single variables or the relationship between two variables in order to understand natural phenomena and make data-based decisions.
- Model numerical response variables using a single explanatory variable or multiple explanatory variables in order to investigate relationships between variables.
- Interpret results correctly, effectively, and in context without relying on statistical jargon.
- Critique data-based claims and evaluate data-based decisions.

Brief List of Topics to be Covered

- Data Exploration and Probability
- Conditional probability and Bayes rule
- Discrete/continuous random variables and computing with distributions
- Joint distributions, covariance, correlation and sums of random variables
- Using Jupyter python environment
- Python tools for data science – NumPy and Pandas
- Basic statistical estimation, random samples, bootstrap and resampling techniques, unbiased estimators and confidence intervals for measure data
- T-Test
- Linear Regression and classification
- Maximum likelihood estimation and analysis of variance

Mathematical Concepts Used

- Counting Theory
- Probabilities
- Integration

CSPB 3202: Introduction to Artificial Intelligence - Prerequisites: **CSPB 2270** & **CSPB 2820** & **CSPB 3022** - Credits: **3**

Brief Description of Course Content - Surveys artificial intelligence techniques of search, knowledge representation and reasoning, probabilistic inference, machine learning, and natural language.

Specific Goals

Specific Outcomes of Instruction

- An ability to explain what AI is about, what it can solve, its brief history and applications, and its social impact.
- An ability to explain key concepts such as agents, environment and how the type of the agent and the environment affect the choice of an algorithm.
- An ability to explain how each AI algorithm works and implement those in codes.
- An ability to explain the algorithm properties such as completeness, optimality, time and space complexity and can compare algorithm efficiencies.
- Determine suitable AI algorithms to apply to a specific problem.

Brief List of Topics to be Covered

- Search
  - Classical search: DFS, BPS, iterative deepening, UCS
  - Non-classical search: heuristics, greedy search, A*
  - CSP
  - Adversarial Search
- Probabilistic Search
  - MDP
  - Reinforcement Learning
  - BayesNets, HMM
- Machine learning in AI
  - Machine learning basics
  - Logistic regression, perceptron, ANN, other ML models (e.g. decision tree)
  - Deep learning, applications in computer vision, NLP, robotics

Mathematical Concepts Used

- Data Structures: Queue, Stack, Tree, Graph
- Probability Basics: Bayes Rule, Conditional Probability, Joint Probability
- Basic Math Functions: Logarithm, Exponent, Argmax/Argmin, Max/Min
- Basic Calculus: Concept of Partial Differentiation, Gradient, Chain Rule

CSPB 3287: Design & Analysis of Database Systems - Prerequisites: **CSPB 2270** - Credits: **3**

Brief Description of Course Content - Analyzes design of data systems, including data stored in file systems, database management systems and physical data organizations. Studies calculus of data models, query languages, concurrency and data privacy and security.

Specific Goals

Specific Outcomes of Instruction

- Understand the theoretical underpinnings of modern databases
- Be capable of choosing appropriate models and their related databases
- Gain familiarity with the underlying mechanisms of various database implementations
- Develop troubleshooting and optimization skills for Relational Database Management Systems

Brief List of Topics to be Covered

- Relational Algebra and Algebraic Query Languages
- Structured Query Language (SQL)
- Data Modelling and Normalization
- Entity Relationship Models and E/R Design Techniques
- Keys, Constraints and Indexes
- Transactions and Optimization
- Hardware and implementation issues
- NoSQL, Key-Value and Column Databases
- Introduction to the CAP Theorem

Mathematical Concepts Used

- Discrete Mathematics
- Algebra
- Set Theory
- Boolean Algebra
- Algorithmic Complexity (Big O Notation)

CSPB 3702: Cognitive Science - Prerequisites: **N/A** - Credits: **3**

Brief Description of Course Content - This course serves as an introduction to Cognitive Science, the study of the mind, as an interdisciplinary field with roots in Computer Science along with Psychology, Education, and a variety of other fields. Our survey of this area centers on how these ideas of mind both inform and are influenced by computer science ideas.

Specific Goals

Specific Outcomes of Instruction - By the end of the course, you should have developed an understanding of the human mind and be able to contribute to discussions, research, and technical work related to the course topics. Hopefully, many of you will be interested in one or more of the areas presented in the course and may use that to shape your projects in future courses or further efforts in your career path.

Brief List of Topics to be Covered - This course is divided into 5 sections:

- The Metaphor of Mind as Machine
- Problem Solving for Minds and Machines
- Vision for Minds and Machines
- Game Theory as Simplified Decision Making
- Developmental Processes and Evolution for Minds and Machines

Mathematical Concepts Used

- Basic Algebra
- Probability & Statistics
- Matrix Multiplication

Many of these concepts are briefly explained in the lessons as a refresher.

CSPB 3753: Design & Analysis of Operating Systems - Prerequisites: **CSPB 2270** & **CSPB 2824** & **CSPB 3308** - Credits: **4**

Brief Description of Course Content - Examines the structure and function of operating systems as an intermediary between applications and computer hardware.

Topics include OS design goals, hardware management, multitasking, process and thread abstractions, file and memory management, security, and networking. Upon completion, students should be able to perform operating systems functions at the support level in a single-user environment.

Microcontrollers are ubiquitous in the modern world, in everything from your toaster, microwave, and refrigerator, to the complex and sophisticated systems in satellites and self-driving vehicles. We have augmented our Operating Systems course to use a Raspberry Pi in hands-on assignments such as adding systems calls to the Linux operating system running on the Raspberry Pi. The goal is for you to learn to apply the theory of operating systems and gain experience physically working with and changing a real working computer.

Specific Goals

Specific Outcomes of Instruction - Upon completion of this course, students are able to:

- Explain basic concepts in the design and structure of operating systems, including kernel/user mode, system calls, preemptive multitasking, and monolithic/microkernel structure.
- Describe how interrupt-based processing achieves efficient management of device input/output communication.
- Define processes and threads, describe different ways to communicate between processes and threads, and apply mutual exclusion-based solutions to synchronize multi-threaded processes without deadlock occurring.
- Identify different scheduling algorithms and their suitability for different types of applications, including compute-bound, I/O-bound and real time.
- Explain the concept of virtual memory, the rationale for on-demand paging, and the role of working sets to avoid thrashing in a caching-based memory hierarchy.
- Demonstrate understanding of fundamental concepts in file system design, including linked and indexed file allocation, mounting, a virtual file system layer, memory mapping, journaling, and performance optimizations for storage media (magnetic and solid state).
- Describe basic concepts to secure and protect operating systems.
- Explain basic concepts in networked operating systems design, including layered network architecture and distributed file systems structure.
- Describe the basic concept of a virtual machine and different types of virtual machines.
- Successfully modify, add functionality to, and re-compile the kernel of an operating system.

Brief List of Topics to be Covered

- Operating System Design and Structure
- Device Input/Output Management
- Process and Thread Management and Scheduling
- Inter-Process Communication and Synchronization
- Memory Management
- File System and Storage Management
- Security

- Networking
- Virtual Machines

Mathematical Concepts Used

- Basic Arithmatic
- Averages
- Recursions
- Random Numbers and Distributions

CSPB 4122: Information Visualization - Prerequisites: **CSPB 1300** - Credits: **3**

Brief Description of Course Content - Studies interactive visualization techniques that help people analyze data. This course introduces design, development, and validation approaches for interactive visualizations with applications in various domains, including the analysis of text collections, software visualization, network analytics, and the biomedical sciences. It covers underlying principles, provides an overview of existing techniques, and teaches the background necessary to design innovative visualizations.

Specific Goals

Specific Outcomes of Instruction

- Define information visualization and describe its role in data analysis
- Illustrate underlying principles in information visualization
- Explain the four nested levels of visualization design
- Design an interactive data visualization
- Build a prototype implementation of an interactive visualization using existing tools/frameworks
- Critique visualization techniques and tools
- Evaluate the utility of visualization system

Brief List of Topics to be Covered

- Review existing design approaches
- Learn a framework for analyzing and critiquing interactive visualizations
- Develop your own visualizations using existing tools and frameworks

Mathematical Concepts Used

- No significant mathematical concepts needed for this course.

CSPB 4502: Data Mining - Prerequisites: **CSPB 2270** - Credits: **3**

Brief Description of Course Content - Introduces basic data mining concepts and techniques for discovering interesting patterns hidden in large-scale data sets, focusing on issues relating to effectiveness and efficiency. Topics covered include data preprocessing, data warehouse, association, classification, clustering, and mining specific data types such as time-series, social networks, multimedia, and Web data.

Specific Goals

Specific Outcomes of Instruction

- Apply the concepts and techniques of Data mining on data sets
- Preprocess and clean data for use in data mining
- Discover interesting patterns from large amounts of data

Brief List of Topics to be Covered

- Data preprocessing
- Data warehouse
- Association
- Classification
- Clustering
- Mining specific data types such as time-series, social networks, multimedia, and Web data

Mathematical Concepts Used

- Statistics