College of Engineering & Applied Sciences

CSPB

CSPB Encyclopedia

Curriculum Encyclopedia

University Of Colorado

2024

Sections

Document Overview

Introduction

This document is an ‘Encyclopedia’ of courses taken at CU Boulder in pursuit of an Applied Computer Science degree in the Post Baccalaureate program.

1.0.1 Courses

Below are the courses that were taken for this degree:

Course Name	Credit Hours	Completion Status
CSPB 1300 - Computer Science 1: Starting Computing	4	Completed - F17 (TA)
CSPB 2270 - Computer Science 2: Data Structures	4	Completed - SM23 (A)
CSPB 2400 - Computer Systems	4	Completed - SP24 (A)
CSPB 2824 - Discrete Structures	3	Completed - F23 (A)
CSPB 3104 - Algorithms	4	Completed - SP24 (A)
CSPB 3155 - Principles Of Programming Languages	4	Scheduled - SM24
CSPB 3308 - Software Development Methods And Tools	3	Scheduled - SM24
CSPB 2820 - Linear Algebra With Computer Science Applications	3	Completed - F23 (A)
CSPB 3022 - Introduction to Data Science With Probability & Statistics	3	Completed - SP24 (A)
CSPB 3202 - Introduction to Artificial Intelligence	3	Scheduled - SM24
CSPB 3287 - Design & Analysis of Database Systems	3	Not Scheduled
CSPB 3702 - Cognitive Science	3	Completed - F23 (A)
CSPB 3753 - Design & Analysis of Operating Systems	4	Scheduled - F24
CSPB 4122 - Information Visualization	3	Not Scheduled
CSPB 4502 - Data Mining	3	Not Scheduled
CSPB 4622 - Machine Learning	3	Scheduled - F24

CSPB 1300 - Computer Science 1: Starting Computing

CSPB 1300

2.0.1 Course Overview

The overview of this course can be seen below.

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

CSPB 1300 - Computer Science 1: Starting Computing

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

3.0.1 Course Overview

The overview of this course can be seen below.

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

CSPB 2270 - Computer Science 2: Data Structures

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 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

3.0.2 Course Notes

The notes for this course can be found below:

CSPB 2270 - Data Structures Course Notes

CSPB 2400 - Computer Systems

CSPB 2400

4.0.1 Course Overview

The overview of this course can be seen below.

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

CSPB 2400 - Computer Systems

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 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

4.0.2 Course Notes

The notes for this course can be found below:

CSPB 2400 - Computer Systems Course Notes

CSPB 2824

5.0.1 Course Overview

The overview of this course can be seen below.

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

CSPB 2824 - Discrete Structures

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 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

5.0.2 Course Notes

The notes for this course can be found below:

CSPB 2824 - Discrete Structures Course Notes

CSPB 3104

6.0.1 Course Overview

The overview of this course can be seen below.

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

CSPB 3104 - Algorithms

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 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

6.0.2 Course Notes

The notes for this course can be found below:

CSPB 3104 - Algorithms Course Notes

CSPB 3155 - Principles Of Programming Languages

CSPB 3155

7.0.1 Course Overview

The overview of this course can be seen below.

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

CSPB 3155 - Principles Of Programming Languages

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 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

8.0.1 Course Overview

The overview of this course can be seen below.

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

CSPB 3308 - Software Development Methods And Tools

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 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 With Computer Science Applications

CSPB 2820

9.0.1 Course Overview

The overview of this course can be seen below.

CSPB 2820 - Linear Algebra With Computer Science Applications - Prerequisites: CSPB 2824 - Credits: 3

CSPB 2820 - Linear Algebra With Computer Science Applications

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

9.0.2 Course Notes

The notes for this course can be found below:

CSPB 2820 - Linear Algebra With Computer Science Applications Course Notes

CSPB 3022 - Introduction to Data Science With Probability & Statistics

CSPB 3022

10.0.1 Course Overview

The overview of this course can be seen below.

CSPB 3022 - Introduction to Data Science With Probability & Statistics - Prerequisites: CSPB 1300 - Credits: 3

CSPB 3022 - Introduction to Data Science With Probability & Statistics

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 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

10.0.2 Course Notes

The notes for this course can be found below:

CSPB 3022 - Introduction To Data Science With Probability And Statistics Course Notes