Utah Computer Science 4th Grade Course Syllabus

One Year for Elementary School, 36 Hours

Course Overview and Goals

The Utah Computer Science 4th Grade Course introduces students to foundational programming concepts through block-based programming language. Students will develop computational thinking and problem-solving skills while learning to create interactive projects, animations, and games. This course emphasizes creativity and collaboration, providing students with a solid base in computer science concepts and digital literacy.

Learning Environment: This course is teacher-led and includes ready-to-use lessons following a consistent structure: Introduction, Guided Practice, Independent Practice, Extension, and Reflection. Instruction follows an "I do, we do, you do" model and incorporates spiral review to reinforce concepts and build confidence over time.

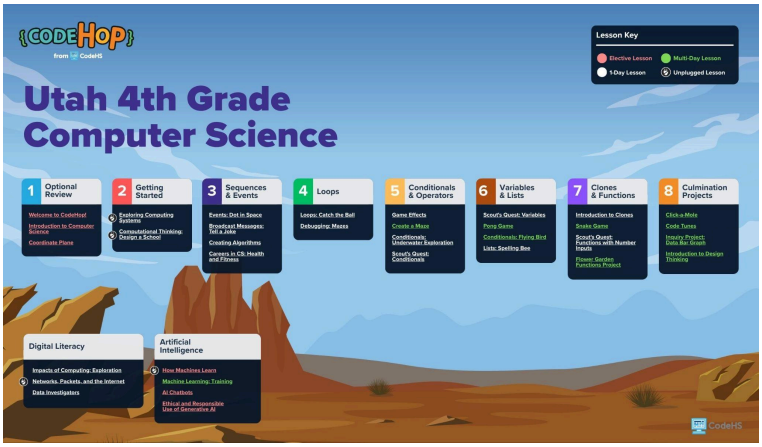
The course includes 36 lessons, each approximately 45 minutes long, providing a full year of instruction when taught once per week. While the course allows for instructional flexibility, some lessons are required to fully meet state computer science standards and are clearly identified within the syllabus. All Digital Literacy lessons are required to ensure full standards alignment, as they address essential non-programming computer science concepts. All lessons are labeled with the specific standards they address to support planning and compliance.

Standards Alignment Note: Standards addressed in each lesson are listed in parentheses after the lesson title. Standards may be reinforced across multiple lessons, and full coverage of state computer science standards is achieved through the course as a whole.

Programming Environment: Students will write and run programs that are saved in students' accounts. The environment supports interactive, hands-on programming, enabling students to create and debug projects in a user-friendly interface.

Prerequisites: There are no prerequisites for this course. It is designed to support all learners, regardless of prior computer science experience.

More Information: Browse the content of this course at https://codehs.com/course/26172/overview



A clickable PDF can be found at https://codehs.com/UT-CSRoadmaps

Course Breakdown

Optional Review

This optional review unit helps students refresh foundational computer science skills and get reacquainted with CodeHop.

Objectives / Topics Covered	● Log in and navigate the Playground. ● Review key computer science terms and programming basics. ● Use the coordinate plane to control sprite movement.
Lessons	**Welcome to CodeHop! (15 minute lesson)** ● Practice logging in and exploring the Playground before starting a full lesson. **Introduction to Computer Science** (4.AP.1) ● Review basic computer science vocabulary and create a simple program. **The Coordinate Plane** (4.AP.1) ● Use the coordinate plane to design an open-ended animation.

Unit 1: Getting Started (2 lessons)

In this introductory unit, students build a foundational understanding of how computing systems work and how to solve problems using computational thinking. Through real-world modeling and design activities, they prepare for deeper exploration of computer science concepts throughout the course.

Objectives / Topics Covered	● Identify components of computing systems and explain how they work together. ● Apply basic troubleshooting strategies to solve hardware and software issues. ● Use computational thinking skills to plan and design a creative solution.
Lessons	**Exploring Computing System** (4.CS.1) ● Identify parts of a computing system, describe how they work together, and practice simple troubleshooting steps. **Computational Thinking: Design a School** (4.CT.1, 4.CT.2) ● Use computational thinking strategies like decomposition and sequencing to design a school with creative problem-solving.

Unit 2: Sequences & Events (5 lessons)

In this unit, students expand their programming skills by using sequences, multiple event types, and broadcast messages to control interactions between sprites.

Objectives / Topics Covered	● Use various event blocks and sequences to control program flow. ● Apply broadcast messages to create sprite interactions. ● Compare multiple algorithms to determine efficiency and effectiveness. ● Explore careers in computer science and how coding connects to health and culture.
Lessons	**Events: Dot in Space** (4.AP.1) ● Create a program using multiple event blocks to control sprite behavior in a space-themed animation. **Broadcast Messages: Tell a Joke** (4.AP.1) ● Use broadcast and receive blocks to animate two sprites telling a knock-knock joke together. **Creating Algorithms** (4.AP.2) ● Build and compare different algorithms to decide which one works best for solving a task.

	Careers in CS: Health and Fitness (4.IC.1) ● Animate how technology supports health and fitness, and learn how coding can help people reach their goals.

Unit 3: Loops (2 lessons)

In this unit, students deepen their understanding of loops by using them to control repetition in games and animations. They'll apply both finite and infinite loops and learn to debug programs by breaking them down into manageable parts.

Objectives / Topics Covered	● Understand and use different types of loops to simplify code. ● Apply loops in interactive programs. ● Decompose and debug programs to identify and fix errors involving loops.
Lessons	**Loops: Catch the Ball** (4.AP.1) ● Use repeat and forever loops to build a simple ball-catching game. **Debugging: Mazes** (4.AP.3) ● Break down a maze program into smaller parts to find and fix errors involving loops and movement.

Unit 4: Conditionals & Operators (4 lessons)

In this unit, students explore how to use conditionals and operators to make their programs more interactive and responsive. They'll apply these concepts by building games, customizing effects, and using logic to guide characters through mazes and underwater scenes.

Objectives / Topics Covered	● Use if/then conditionals and operators to control actions based on specific conditions. ● Apply conditionals in game logic to enhance interactivity. ● Create custom visuals like mazes and game effects. ● Improve programs through peer feedback and reflection.
Lessons	**Game Effects** (4.AP.3, 4.AP.5) ● Modify an existing game by adding creative effects and making improvements based on peer feedback. **Create a Maze** (2 classes | 4.AP.3) ● Design a custom maze backdrop and program Scout to navigate through it using logic and movement. **Conditionals: Underwater Exploration** (4.AP.1) ● Build an underwater scene that uses conditionals to control character behavior and responses. **Scout's Quest: Conditionals** (4.AP.1) ● Use if/then conditionals to build a program that responds to different inputs or scenarios.

Unit 5: Variables & Lists (6 lessons)

In this unit, students learn how to use variables and lists to store, update, and organize information. They'll apply these concepts to build interactive games like Pong, Flying Bird, and a Spelling Bee, while reinforcing skills through Scout's Quest.

Objectives / Topics Covered	● Create and use variables to track information like scores. ● Use conditionals with variables to control game behavior. ● Create and manage lists to organize and display information in a program.

Lessons	**Scout's Quest: Variables** (4.AP.1) ● Practice creating and updating variables to track points in a program. **Pong Game** (2 classes | 4.AP.1, 4.AP.2) ● Design a pong-style game using variables to keep score as the ball bounces between paddles. **Conditionals: Flying Bird** (2 classes | 4.AP.1, 4.AP.2) ● Use conditionals and variables to control gameplay and scoring in a Flying Bird game. **Lists: Spelling Bee** (4.AP.2) ● Build a spelling bee game that uses lists to store and display words players must spell.

Unit 6: Clones & Functions (7 lessons)

In this unit, students explore advanced programming concepts like cloning and functions with inputs. They'll use these tools to create dynamic projects—from animations and games to creative drawings—while developing efficient, reusable code structures and continuing skill review through Scout's Quest.

Objectives / Topics Covered	● Create and use clones to animate or duplicate sprites in a program. ● Build functions with Boolean and number inputs to control program behavior. ● Use variables and clones to design interactive games. ● Apply functions to automate and simplify complex drawings or repeated tasks.
Lessons	**Introduction to Clones** (4.AP.1) ● Create an animation using clones and explore how duplicate sprites can be used efficiently—and where they may have limitations. **Snake Game** (2 classes | 4.AP.1, 4.AP.2) ● Build a version of the classic Snake game using clones and variables to track movement and score. **Scout's Quest: Functions with Number Inputs** (4.AP.1) ● Use number inputs in functions to create a program that draws with specific values. **Flower Garden Functions Project** (2 classes | 4.AP.1) ● Use a function to draw multiple flowers in a scene, making the code cleaner and more efficient through reuse.

Unit 7: Culmination Projects (8 lessons)

In this final unit, students put their programming and problem-solving skills into action through creative, real-world projects. They'll apply conditionals, variables, operators, and data visualization to design interactive games, music tools, and visual displays, and explore 3D modeling through an accessibility-focused design challenge.

Objectives / Topics Covered	● Apply programming concepts like conditionals, events, variables, booleans, and operators in original projects. ● Follow the inquiry process to collect, analyze, and display data through coding. ● Design solutions with creativity and purpose using code and 3D modeling tools.
Lessons	**Click-a-Mole** (2 classes | 4.AP.1, 4.AP.2) ● Design a Whack-a-Mole–style game using conditionals, variables, booleans, and events to create interactive gameplay. **Code Tunes** (2 classes | 4.AP.1, 4.AP.2) ● Build a custom music player using conditionals, operators, and variables to control features. **Inquiry Project: Data Bar Graph** (2 classes | 4.DA.2, 4.DA.3) ● Conduct an investigation and modify a program to display survey results using a bar graph.

	Introduction to Design Thinking (2 classes | 4.IC.1) ● Use the design thinking process to create a tool that is accessible to more users.

Unit 8: Digital Literacy (3 lessons)

In this unit, students explore how computing systems impact people and communities while learning about networks, online safety, and data. They investigate how information travels across the internet, evaluate data for reliability, and examine the effects of technology on everyday life.

Objectives / Topics Covered	● Explore how computing impacts people and communities. ● Understand how information travels across networks. ● Identify strategies for staying safe online. ● Evaluate data to draw conclusions and make predictions.
Lessons	**Impacts of Computing: Exploration** (4.IC.1) ● Explore how computing has changed over time and its impact on people and communities. **Networks, Packets, and the Internet** (4.NI.1, 4.NI.2) ● Explore how devices communicate and transfer information across networks. **Data Investigators** (4.DA.2) ● Evaluate data for reliability and use it to draw conclusions and make predictions.

Optional: Artificial Intelligence (5 lessons)

In this optional unit, students explore how artificial intelligence learns from data, recognizes patterns, and makes predictions. They investigate how machine learning models are trained, how AI chatbots respond to questions, and how to use generative AI responsibly.

Objectives / Topics Covered	● Explain how machine learning models are trained. ● Explore how training data affects AI predictions. ● Investigate how AI chatbots use information to answer questions. ● Describe responsible and ethical uses of generative AI.
Lessons	**How Machines Learn** (4.DA.2) ● Explore how machine learning models use data to make predictions. **Machine Learning: Training** (4.DA.2) ● Investigate how training data affects the accuracy of AI models. **AI Chatbots** (4.IC.1) ● Explore how AI chatbots use information to answer questions. **Ethical and Responsible Use of Generative AI** (4.IC.1) ● Explore responsible and ethical ways to use generative AI.

Utah Computer Science 4th Grade Course Supplemental Materials

Resources	**Description**
Parent Welcome Letter (Spanish)	Send this letter home to introduce families to their new computer science curriculum.
Warm-Up Activities	This warm-up activity slide deck provides 5-10 minute problems aligned with

	computer science skills to engage students at the start of class, allowing teachers to preview or review concepts with answer keys and discussion tips included in the Speaker Notes.
Program Self-Assessment (Spanish)	This is a student self-assessment tool designed to help K-6 learners reflect on their programming projects, evaluate their skills in algorithms, debugging, collaboration, and reflection, and set goals for improvement.
Peer Review Resources (Spanish)	This provides structured worksheets to facilitate student feedback during collaborative coding projects. It encourages reflection by guiding students to highlight successes, ask questions, and offer constructive feedback on their partner's work.
Lesson Reflection & Computational Thinking (Spanish)	This guides students in engaging with computational thinking concepts, preparing for discussions, reflecting on lessons, and applying their learning to real-world problem-solving.
Design-Your-Own-Lesson Templates	Empower your students to explore and express their knowledge creatively with our versatile graphic organizer templates. Designed with adaptability and ease of use in mind, these interactive tools transform any subject into an engaging, hands-on learning experience.
These resources and more are found on the **CodeHop Resources Page**.	