Idaho Computer Science 4th Grade Course Syllabus

One Year for Elementary School, 36 Hours

Course Overview and Goals

The **Idaho Computer Science 4th Grade Course** introduces students to foundational programming concepts through a 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.

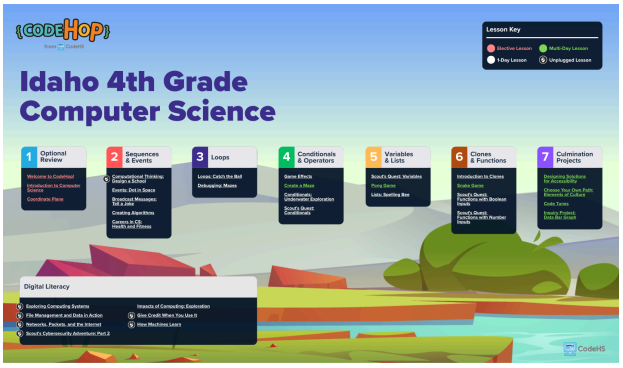
The course includes 36 lessons, each approximately 45 minutes long, providing a full year of instruction when taught once per week. While the course allows for instructional flexibility, some lessons are required to fully meet state computer science standards and are clearly identified within the syllabus.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

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

Assessments: Some units conclude with a brief five-question quiz designed to assess student understanding of the concepts and skills introduced in the unit. Quizzes can be assigned digitally, completed as a QuizBuzz activity, or printed for paper-based completion.

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/26581/overview



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

Course Breakdown

Optional Review

This optional review unit helps students refresh foundational computer science skills and get reacquainted with the platform. It includes lessons on logging in, reviewing vocabulary, and practicing programming with coordinates and simple animations.

Objectives / Topics Covered	● Log in and navigate the Playground. ● Review key computer science terms and 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** ● Review basic computer science vocabulary and create a simple program. **The Coordinate Plane** ● Use the coordinate plane to design an open-ended animation.

Unit 1: Sequences & Events (2 lessons)

Students will build interactive programs using event blocks and broadcast messages to control the flow of actions between sprites.

Objectives / Topics Covered	● Use multiple types of event blocks to trigger actions. ● Coordinate sprite interactions using broadcast messages.
Lessons	**Computational Thinking: Design a School (3-5.CT.5.7, 3-5.CT.5.8)** ● Use computational thinking to design and plan the layout of a school. **Events: Dot in Space** ● Create a program using different types of event blocks to control actions. **Broadcast Messages: Tell a Joke** ● Use broadcast messages to program two sprites to perform a knock knock joke. **Creating Algorithms (3-5.CT.5.6)** ● Create a program using different types of event blocks to control actions. **Careers in CS: Health and Fitness (3-5.DL.3.3)** ● Animate how technology supports health and fitness by amplifying human strengths. **Sequences & Events Quiz (5 questions)**

Unit 2: Loops (2 lessons)

This unit introduces loops as a tool to make programs more efficient. Students also practice debugging to identify and fix looping errors.

Objectives / Topics Covered	● Understand and apply repeat and forever loops. ● Build simple games using looping structures. ● Decompose and debug faulty programs.
Lessons	**Loops: Catch the Ball** ● Use repeat and forever loops to build a simple ball-catching game. **Debugging: Mazes (3-5.CT.5.5)** ● Decompose and fix a maze program to make it run as expected. **Loops Quiz (5 questions)**

Unit 3: Conditionals & Operators (5 lessons)

Students explore conditional logic and operators to create smarter, more responsive programs. They apply these skills to animations and games with custom effects.

Objectives / Topics Covered	● Use if/then conditionals and logical operators. ● Build maze games and interactive experiences. ● Revise programs based on peer feedback.
Lessons	**Game Effects** ● Modify an existing game by adding creative effects and making improvements based on peer feedback. **Create a Maze (2 classes)** ● Design a custom maze backdrop and program Scout to navigate through it using logic and movement. **Conditionals: Underwater Exploration** ● Build an underwater scene that uses conditionals to control character behavior and responses. **Scout's Quest: Conditionals** ● Use if/then conditionals to build a program that responds to different inputs or scenarios. **Conditionals & Operators Quiz (5 questions)**

Unit 4: Variables & Lists (4 lessons)

Students build interactive programs by using variables and lists to track points, display words, and manage user input.

Objectives / Topics Covered	● Create and modify variables to track game data. ● Use lists to manage multiple data points like spelling words. ● Apply variable and list logic in games.
Lessons	**Pong Game (2 classes)** ● Create a pong game that uses variables to keep score. **Scout's Quest: Variables** ● Practice creating and updating variables to track points in a program. **Lists: Spelling Bee** ● Build a spelling bee game that uses a list of words for interactive play. **Variables & Lists Quiz (5 questions)**

Unit 5: Clones & Functions (5 lessons)

Students explore advanced programming tools—clones and functions with inputs—to create more dynamic and efficient projects.

Objectives / Topics Covered	● Use clones to animate repeated or duplicated elements. ● Create functions with boolean or number inputs. ● Apply functions in creative and logic-based animations.
Lessons	**Introduction to Clones** ● Create an animation using clones and explore how duplicate sprites can be used efficiently—and where they may have limitations. **Snake Game (2 classes)** ● Build a version of the classic Snake game using clones and variables to track

	movement and score. **Scout's Quest: Functions with Boolean Inputs** ● Create a function with a true/false input to determine actions, like checking a password. **Scout's Quest: Functions with Number Inputs** ● Use number inputs in functions to create a program that draws with specific values. **Clones & Functions Quiz (5 questions)**

Unit 6: Culmination Projects (8 lessons)

In this unit, students apply their programming knowledge and design skills to create meaningful, original projects. They'll solve real-world problems, personalize interactive tools, and present data-driven findings.

Objectives / Topics Covered	● Apply the design thinking process to solve real-world problems. ● Create interactive cultural, musical, and data-driven experiences. ● Use conditionals, variables, lists, and operators in original projects.
Lessons	**Designing Solutions for Accessibility (2 classes | 3-5.CT.2.3, 3-5.CT.5.1, 3-5.CT.5.2, 3-5.CT.5.3, 3-5.CT.5.4, 3-5.CT.5.9)** ● Use design thinking to build a program that improves access or inclusion. **Choose Your Own Path: Elements of Culture (2 classes | 3-5.CT.1.4)** ● Design a cultural choose-your-own-path interactive game. **Code Tunes (2 classes | 3-5.CT.1.3)** ● Create a custom music player using variables, conditionals, and operators. **Inquiry Project: Data Bar Graph (2 classes | 3-5.CT.2.1, 3-5.CT.2.4, 3-5.CT.2.5, 3-5.CT.2.6, 3-5.CT.3.3, 3-5.DL.3.2)** ● Follow the inquiry process to collect data and display it in a bar graph.

Unit 7: Digital Literacy (7 lessons)

Students deepen their understanding of how digital systems work, how to stay safe online, and how computers learn and process data.

Objectives / Topics Covered	● Identify components of computing systems and solve basic tech issues. ● Understand file types, storage, and how data moves through networks. ● Practice safe online behavior and responsible attribution. ● Explore how machine learning works and how to model decision trees.
Lessons	**Exploring Computing Systems (3-5.CT.1.1, 3-5.DL.3.1)** ● Identify parts of a computing system and resolve simple hardware/software problems. **File Management and Data in Action (3-5.CT.2.2)** ● Explain how different digital files vary in size and storage methods. **Networks, Packets, and the Internet (3-5.CT.4.1)** ● Model how data travels as packets across a network and is reassembled securely. **Scout's Cybersecurity Adventure: Part 2 (3-5.CT.3.2, 3-5.DL.2.1, 3-5.DL.2.2)** ● Practice safe online habits and understand digital protection tools. **Impacts of Computing: Exploration (3-5.DL.4.1)** ● Explore how computing has changed communities and evolved over time. **Give Credit When You Use It (3-5.CT.3.1, 3-5.CT.3.4, 3-5.DL.1.1)** ● Conduct research and cite online sources appropriately. **How Machines Learn (3-5.CT.1.2)** ● Learn about types of machine learning and create a classification system using a decision tree. **Digital Literacy Quiz (5 questions)**

Idaho Computer Science 4th Grade Course Supplemental Materials

Resources	Description
Parent Welcome Letter (Spanish)	Send this letter home to introduce families to computer science.
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.
All of these resources and more are found on the **CodeHop Resources Page**.	