Idaho Computer Science 2nd Grade Course Syllabus

One Year for Elementary School, 36 Hours

Course Overview and Goals

The **Idaho Computer Science 2nd Grade Course** introduces students to foundational programming concepts through **ScratchJr**, 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.

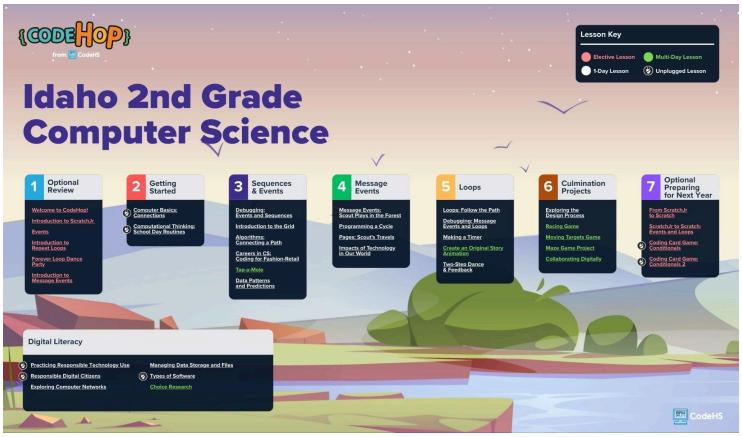
Learning Environment: This course is designed to be teacher-led, with ready-to-use lesson plans that follow a structured format: **Introduction, Guided Practice, Independent Practice, Extension, and Reflection**. Lessons are built with spiral review to reinforce key concepts and culminate in engaging projects to showcase student understanding.

The lessons are delivered in an **"I do, we do, you do"** format, ensuring a gradual release of responsibility and fostering confidence in students as they learn. Teachers can adapt the content to fit their schedule and instructional needs. The concepts taught in this course spiral across grade levels, ensuring that students can revisit and build upon their understanding year after year, even if all lessons are not completed within a single year. The course includes a total of 36 **lessons**, with each lesson approximately 45 minutes long. This provides a full school year of material if teaching one lesson per week. Optional digital literacy lessons are also available to complement the programming curriculum with non-programming computer and technology skills.

Programming Environment: Students will write and run programs in **ScratchJr** embedded and saved in the CodeHS platform. 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/26466/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 skills such as navigating the Playground, using ScratchJr, and understanding events and loops.

Objectives / Topics Covered	● Review basic ScratchJr interface and functionality. ● Practice using events, loops, and messages. ● Reinforce foundational computer science concepts before progressing.
Lessons	**Welcome to CodeHop! (15 minute lesson)** ● Introductory lesson to help students log in and explore the CodeHS Playground; ideal as a warm-up or standalone activity. **Introduction to ScratchJr** ● Navigate the ScratchJr interface and create a scene with characters and background. **Events** ● Use multiple event blocks and explain how events work in programming. **Introduction to Repeat Loops** ● Learn how to use repeat loops to make actions happen more than once in a row. **Forever Loop Dance Party** ● Use “repeat forever” loops to build a looping dance animation.

Unit 1: Getting Started (2 lessons)

Students begin by understanding how computers function and how to apply computational thinking to everyday tasks.

Objectives / Topics Covered	● Learn basic computer components and how they work together. ● Apply computational thinking to familiar routines.
Lessons	**Computer Basics: Connections** ● Learn what a computer is and how hardware, software, input, and output work together. **Computational Thinking: School Day Routines** ● Break down school day routines using patterns, sequencing, and problem-solving skills.

Unit 2: Sequences & Events (8 lessons)

In this unit, students dive deeper into programming logic with sequences, events, and debugging while beginning to use grid-based movement and algorithm design.

Objectives / Topics Covered	● Program sequences and trigger actions with events. ● Debug code and refine algorithms. ● Use the grid and character size to plan movement.
Lessons	**Debugging: Events and Sequences** ● Find and fix errors in sample code to improve program outcomes. **Introduction to the Grid** ● Use the grid to move characters precisely on the stage. **Algorithms: Connecting a Path** ● Write and adjust step-by-step instructions to connect characters on the screen. **Careers in CS: Coding for Fashion-Retail** ● Learn how coding is used in fashion, then animate a custom fashion character.

	Tap-a-Mole Game (2 part lesson) ● Design a game using events where users tap characters to score points. **Data Patterns and Predictions** ● Analyze patterns in data and program an animation to communicate insights. **Impacts of Technology in Our World** ● Create a program to show how technology influences daily life.

Unit 3: Message Events (3 lessons)

Students explore how messages can control program flow and build animations that model real-world systems and storytelling.

Objectives / Topics Covered	● Program interactions between characters using message events. ● Model cycles and communication flows. ● Coordinate scenes with messaging across pages.
Lessons	**Message Events: Scout Plays in the Forest** ● Control a program's flow using messages to trigger actions. **Programming a Cycle** ● Use messages to represent and loop through a natural or repeating cycle. **Pages: Scout's Travels** ● Navigate between pages using messages to help Scout move through a story.

Unit 4: Loops (6 lessons)

Students will explore how loops simplify patterns in code, practice debugging, build timers, and create animated stories. They'll also learn to revise programs using peer feedback and give proper credit.

Objectives / Topics Covered	● Identify and use loops to repeat actions and patterns. ● Debug code involving message events and loops. ● Use loops with timing and motion blocks. ● Create and revise animations with feedback and attribution.
Lessons	**Loops: Follow the Path** ● Identify repeated patterns and use loops to move characters efficiently. **Debugging: Message Events and Loops** ● Find and fix coding errors involving loops and message events. **Making a Timer** ● Use loops, wait, and turn blocks to create two timers with different speeds. **Create an Original Story Animation (2 part lesson)** ● Design and animate a unique story using characters, loops, and events. **Two-Step Dance & Feedback** ● Code a dance, revise it with peer feedback, and give credit to collaborators.

Unit 5: Culmination Projects (11 lessons)

Students apply everything they've learned to build creative, interactive programs. These open-ended projects reinforce the design process, interactivity, and peer review.

Objectives / Topics Covered	● Plan and revise programs using the design process. ● Build original games with events, messages, loops, and pages. ● Collaborate and revise work based on feedback.
Lessons	**Exploring the Design Process** ● Apply all steps of the design process independently to create and improve a program

	that includes loops to solve a real-world problem from a user's perspective. **Racing Game (2 part lesson)** ● Create a racing game using events, loops, and messages to control character actions. **Moving Targets Game (3 part lesson)** ● Program a game with moving targets that uses sequences, events, and pages. **Maze Game Project (3 part lesson)** ● Design a maze game and revise it using feedback while applying key programming concepts. **Collaborating Digitally (2 part lesson)** ● Create and share a program digitally, give and receive feedback, and practice collaboration by improving a project with input from peers.

Unit 6: Digital Literacy (7 lessons)

This unit helps students become responsible and informed technology users while exploring how digital systems work and how data is stored and managed.

Objectives / Topics Covered	● Understand the importance of usernames and passwords. ● Practice responsible online behavior and digital citizenship. ● Research a topic, assess sources, and share results visually.
Lessons	**Practicing Responsible Technology Use** ● Demonstrate how to safely and respectfully use digital tools. **Responsible Digital Citizens** ● Explore digital footprints, cyberbullying, and how to report unsafe online behavior. **Exploring Computer Networks** ● Model how devices send and receive information over a network. **Managing Data Storage and Files** ● Recognize how computers store data and how it is organized. **Types of Software** ● Compare and choose appropriate software tools for tasks. **Choice Research (2 part lesson)** ● Research a self-selected topic using reliable sources and create a visual program to share findings.

Optional Preparing for Next Year

These optional lessons help students build on what they've learned by applying conditionals in unplugged activities and transitioning from ScratchJr to Scratch programming.

Objectives / Topics Covered	● Use conditionals to create step-by-step instructions. ● Collaborate to solve maze challenges with logic. ● Explore the Scratch interface and basic programming tools. ● Create simple Scratch programs using events and loops.
Lessons	**From ScratchJr to Scratch** ● Learn the basics of Scratch by building a simple program from scratch. **ScratchJr to Scratch: Events and Loops** ● Create a Scratch program that includes an event and a repeating action using a loop. **Coding Card Game: Conditionals** ● Use conditionals to guide Scout through a maze using a card-based activity. **Coding Card Game: Conditionals 2** ● Continue practicing conditional logic with new maze challenges.

Idaho Computer Science 2nd 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.
All of these resources and more are found on the **Elementary Resources Page**.	