CodeHS

Georgia Computer Science Standards of Excellence: 2nd Grade Course Syllabus

One Year for Elementary School, 36 Hours

Course Overview and Goals

The **Georgia Computer Science Standards of Excellence: 2nd Grade** 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**, each approximately 45 minutes long. This provides a full school year of material if teaching one lesson per week. Digital literacy lessons are also available to complement the programming curriculum with non-programming computer and technology skills. Additionally, this course includes optional interdisciplinary lessons in math, science, ELA, and social studies to support cross-curricular integration.

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/26332/overview?lang=en.

Course Breakdown

Unit 1: Optional Review

In this optional unit, students review foundational programming skills and prepare for more advanced ScratchJr projects. They revisit key coding concepts such as events, loops, and message passing while reinforcing their ability to navigate the CodeHS platform and ScratchJr interface.

Objectives / Topics Covered	● Log in and navigate the CodeHS Playground and ScratchJr interface. ● Use events to control character actions in response to triggers. ● Apply repeat and forever loops to simplify repeating actions in a program. ● Use message events to coordinate interactions between characters.
Lessons	**Welcome to CodeHS!** ● Learn how to log in and use the CodeHS Playground. This short introductory lesson can be used on its own or right before a full lesson. **Introduction to ScratchJr** ● Navigate the ScratchJr interface to create a scene with characters. **Events** ● Explain what an event is in programming and use multiple event blocks in a program. **Introduction to Repeat Loops** ● Use repeat loops to run a section of code multiple times. **Forever Loop Dance Party** ● Create a sequence using a “repeat forever” loop to make characters repeat actions. **Introduction to Message Events** ● Program a relay race that uses messages to cause characters to interact.

Unit 2: Getting Started (2 weeks)

In this introductory unit, students develop a foundational understanding of computer systems and digital responsibility. They explore how hardware and software work together and learn how to act safely and respectfully in digital spaces.

Objectives / Topics Covered	● Define what it means to be a responsible digital citizen. ● Understand digital footprints and how to respond to cyberbullying. ● Identify input, output, hardware, and software components. ● Explain how computer parts work together to complete tasks. ● Practice troubleshooting common computer issues.
Lessons	**Responsible Digital Citizens** ● Explain what it means to be a responsible digital citizen, including understanding digital footprints, discussing cyberbullying, and knowing how to report concerns. **Computer Basics: Connections** ● Learn what a computer is, how we use it, and what to do when it doesn’t work; identify input, output, hardware, and software, and explain how they work together.

Unit 3: Sequences & Events (9 weeks)

In this unit, students deepen their understanding of sequencing, events, and algorithms through real-world connections and creative programming. They practice debugging, use the grid for precise movement, and build games and animations that reflect peer feedback and applied problem-solving.

Objectives / Topics Covered	● Use computational thinking to break down and sequence real-life routines. ● Create and adjust algorithms based on character traits and position. ● Debug programs with sequences and events.

	● Use the grid to control character placement on the stage. ● Apply events and loops to create timers, games, and animations.
Lessons	**Computational Thinking: School Day Routines** ● Use computational thinking concepts to identify patterns, break down tasks, sequence steps, and simplify processes in their school day routines. **Debugging: Events and Sequences** ● Find and fix errors in provided code. **Introduction to the Grid** ● Use the grid feature to move characters to a specific location on the stage. **Careers in CS: Coding for Fashion-Retail** ● Explain how coding helps create and improve fashion designs and create a program to design and animate a fashion character. **Two-Step Dance & Feedback** ● Create a program and revise it based on peer feedback and give attribution to a peer who helped improve their work. **Algorithms: Connecting a Path** ● Create and adjust simple algorithms to move characters based on their size, shape, and starting position. **Making a Timer** ● Use loops, wait blocks, and turn blocks to create and compare two timers with different speeds. **Tap-a-Mole Game (2-part lesson)** ● Create an interactive game using events.

Unit 4: Message Events (3 weeks)

In this unit, students build on their understanding of events by using message events to control the timing and flow of multi-step programs. They explore how messages can trigger actions, represent cycles, and coordinate movement across pages.

Objectives / Topics Covered	● Use message events to sequence actions and coordinate character behavior. ● Model real-world cycles through messaging in code. ● Apply message events to navigate across multiple pages in a program.
Lessons	**Message Events: Scout Plays in the Forest** ● Use message events to control the flow of a program. **Programming a Cycle** ● Use message events to model a cycle. **Pages: Scout's Travels** ● Use messages to help Scout travel between pages in a program.

Unit 5: Loops (4 weeks)

In this unit, students explore how loops can simplify repetitive actions in their programs. They identify patterns, use loops in creative projects, and strengthen debugging skills by analyzing and fixing code with loops and message events.

Objectives / Topics Covered	● Identify and apply patterns to create loop-based sequences. ● Use loops to make code more efficient in storytelling and animations. ● Debug programs involving loops and message events. ● Develop and animate original fiction stories using learned coding concepts.
Lessons	**Loops: Follow the Path** ● Identify patterns and create a program using loops. **Debugging: Message Events and Loops** ● Find and fix (debug) message events and loop errors in the provided code.

	Original Story Animations – Fiction Story (2-part lesson) ● Develop an original story and create a program to animate the story.

Unit 6: Culmination Projects (13 weeks)

In this unit, students apply their full range of computer science skills to design and build original, interactive projects. These games and animations showcase their understanding of sequences, loops, events, and message passing, and give them opportunities to reflect, revise, and demonstrate creativity through code.

Objectives / Topics Covered	● Use a variety of coding blocks to create interactive programs. ● Apply sequences, loops, events, and message events in game design. ● Build multi-page, story-based games and animations. ● Revise and improve projects based on peer feedback. ● Demonstrate mastery of key computer science concepts through original creations.
Lessons	**Code Block Review** ● Use a variety of coding blocks in a program and explain their function within the program. **Racing Game** ● Create an interactive racing game with events, loops, and messages. **Moving Targets Game** ● Create a moving target game using sequences, events, and pages. **Maze Game Project** ● Create and explore multiple ways to program an interactive game using events, messages, loops, and sequences, and revise the program based on feedback. **Design an Adventure Game** ● Create a story-based, multi-page game using computer science skills they have learned.

Unit 7: Digital Literacy (5 weeks)

In this unit, students explore how computers store and use data, and how to communicate findings through visual programming. They analyze patterns, make predictions, and conduct research while building foundational skills in data literacy and digital communication.

Objectives / Topics Covered	● Recognize that computers store data as files and model data storage. ● Identify and describe patterns in data and use them to make predictions. ● Use programming to visually represent data and research findings. ● Practice assessing sources and communicating research clearly. ● Explore the design process and how it supports problem-solving.
Lessons	**Managing Data Storage and Files** ● Recognize that computers store data as files and model how data is collected and stored. **Data Patterns and Predictions** ● Identify and describe patterns in data visualizations, then create a program using events to communicate patterns and predictions from a given data set. **Choice Research (2-part lesson)** ● Collect and assess sources to answer a research question and communicate their findings visually. **Exploring the Design Process** ● *This lesson is coming soon!*

Unit 8: Preparing for Next Year (4 weeks)
In this optional unit, students are introduced to concepts that prepare them for more advanced programming. They explore conditionals and variables, begin working with the Scratch programming environment, and bridge their learning from ScratchJr to Scratch by applying familiar coding concepts in a new setting.

Objectives / Topics Covered	● Use conditionals to control decision-making in a program. ● Simulate variable use to keep score in interactive activities. ● Navigate the Scratch interface and build simple programs. ● Apply events and loops in Scratch to extend prior learning. ● Collaborate to solve problems using structured logic.
Lessons	**Coding Card Game: Conditionals** ● Work together to create a sequence of instructions with conditionals to move Scout through a maze. **Coding Card Game: Conditionals 2** ● Work together to create a sequence of instructions with conditionals to move Scout through a maze. **Variables: Keeping Score** ● Create a program that simulates keeping score using a variable. **From ScratchJr to Scratch** ● Navigate the basic interface of the Scratch editor to create a simple program. **ScratchJr to Scratch: Events and Loops** ● Create a program in Scratch that uses an event and a loop.

Unit 9: Optional Interdisciplinary (10 weeks)
In this optional cross-curricular unit, students integrate computer science concepts with academic subjects including math, science, ELA, and social studies. Through interactive ScratchJr programs, they model core concepts and demonstrate their understanding using sequences, events, and loops.

Objectives / Topics Covered	● Apply sequences, events, and loops to represent academic concepts. ● Model scientific processes and environmental change through animation. ● Reinforce math skills like place value and telling time. ● Build and read complete sentences using coding elements.
Lessons	**Telling Time** ● Use sequences and events to create an analog clock and display time in digital and analog forms. **Place Value: Ones, Tens, and Hundreds** ● Connect a digit’s place in a number to its value and create an interactive program that uses events. **Changes in the Environment** ● Identify changes in the environment and their causes, and use animation to model. **Seed Dispersal** ● Create a program using message events and loops to model how an animal can help disperse seeds. **Build a Sentence** ● Create an interactive program that uses events to write sentences and then read them aloud. **Punctuation: Write a Great Sentence!** ● Create sequences with loops to write sentences with correct punctuation and spacing. **Communities Modify Their Environment** ● Create a program that shows how people modify their environment in a community. **Create a Map (3-day lesson)** ● Create a map and program a character to follow the map.

2nd Grade Course Supplemental Materials

Resources	**Description**
Parent Welcome Letter (Spanish)	Send this letter home to introduce families to computer science with CodeHS.
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.
These resources and more are found on the **Elementary Resources Page**.	