CodeHop

Georgia Computer Science 1st Grade Course Syllabus

One Year for Elementary School, 36 Hours

Course Overview and Goals

The **Georgia Computer Science 1st 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. Digital literacy lessons are also available to complement the programming curriculum with non-programming skills. Additionally, this course includes optional interdisciplinary lessons to support cross-curricular integration.

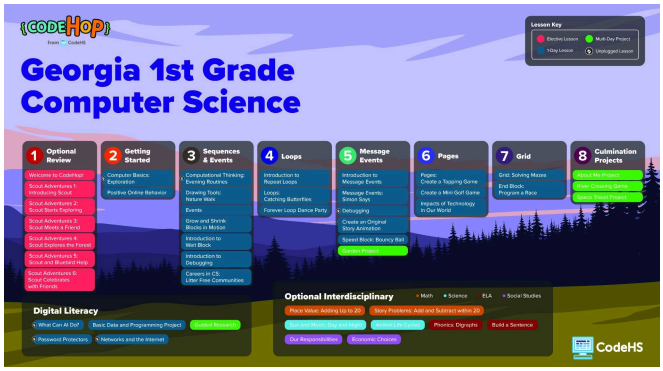
Learning Environment: This course is designed to be teacher-led, with ready-to-use lesson plans that follow a structured format: **Introduction, Guided Practice, Independent Practice, Extension, and Reflection**. Lessons are built with spiral review to reinforce key concepts and culminate in engaging 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. The concepts taught in this course spiral across grade levels, ensuring that students can build upon their understanding year after year. The course includes approximately **36 instructional hours**, providing a full school year of material if teaching one lesson per week. Depending on your pacing, you may find opportunities to adjust 1–2 lessons to best fit your calendar. Lesson prep notes on each lesson page provide ideas for streamlining activities while maintaining key learning goals.

Programming Environment: Students will write and run programs in **ScratchJr** embedded and saved in the CodeHop 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/26290/overview?lang=en.



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

Course Breakdown

Unit 1: Optional Review

In this optional unit, students learn to navigate the CodeHop platform and build foundational programming skills by exploring ScratchJr to add and modify characters, backgrounds, and pages, culminating in creating animated scenes using sequences of motion and event blocks to tell a story.

Objectives / Topics Covered	● Log in and navigate the CodeHop Playground. ● Explore the ScratchJr interface and add characters and backgrounds. ● Create multi-page stories with scenes and transitions. ● Use motion blocks to build simple sequences. ● Introduce events to trigger character actions.
Lessons	**Welcome to CodeHop! (15 minute lesson)** ● Log in and explore how to use the CodeHop Playground. **Scout Adventures 1: Introducing Scout** ● Add characters to the stage using ScratchJr. **Scout Adventures 2: Scout Starts Exploring** ● Add backgrounds and a new page to create a multi-scene story. **Scout Adventures 3: Scout Meets a Friend** ● Delete and edit characters to shape the story. **Scout Adventures 4: Scout Explores the Forest** ● Use motion blocks to move characters around the stage. **Scout Adventures 5: Scout and Bluebird Help** ● Build a sequence of motion blocks to guide characters in collecting objects. **Scout Adventures 6: Scout Celebrates with Friends** ● Create a celebration scene using events and multi-step animations.

Unit 2: Getting Started (2 weeks)

In this introductory unit, students explore what computers are, how we interact with them, and how to use them safely and responsibly. Students identify computer components, distinguish between hardware and software, and recognize examples of positive online behavior through interactive ScratchJr lessons.

Objectives / Topics Covered	● Identify what a computer is and how we use it. ● Recognize hardware, software, input, and output. ● Practice basic troubleshooting steps. ● Understand and model safe and respectful online behavior. ● Create programs that reflect real-world technology use.
Lessons	**Computer Basics: Exploration** ● Learn what a computer is, how we use it, and what to do when it doesn't work; identify input, output, hardware, and software. **Positive Online Behavior** ● Identify appropriate and inappropriate online behavior in a ScratchJr program.

Unit 3: Sequences & Events (7 weeks)

In this unit, students apply sequencing and events to solve real-world problems and tell interactive stories. They use computational thinking to break down routines, practice drawing and timing in ScratchJr, and explore how computer science supports their community.

Objectives / Topics Covered	• Use sequencing to organize actions and solve problems. • Apply events to trigger specific behaviors in a program. • Use visual effects like grow, shrink, and wait to enhance animations. • Identify and fix bugs in a program. • Explore real-world applications of computer science in community settings.
Lessons	**Computational Thinking: Evening Routines** • Use computational thinking concepts to identify patterns, break down tasks, sequence steps, and simplify processes in their evening routines. **Drawing Tools: Nature Walk** • Use drawing tools to create a nature-walk scene. **Events** • Explain what an event is in programming and use multiple event blocks in a program. **Grow and Shrink Blocks in Motion** • Create a program using motion blocks and grow and shrink blocks to change the size of characters. **Introduction to the Wait Block** • Use "wait" blocks to cause characters to pause in a program. **Introduction to Debugging** • Find and correct bugs in sequences. **Careers in CS: Litter Free Communities** • Explain how computer science can help solve community problems and create a program to collect and sort litter.

Unit 4: Loops (3 weeks)

In this unit, students explore how loops can make code more efficient by repeating actions. Through animations and simple games, students learn to use both finite and infinite loops to build dynamic, interactive programs.

Objectives / Topics Covered	• Use repeat loops to run a sequence multiple times. • Create continuous animations using the “forever” loop. • Combine loops with motion and visibility blocks to enhance storytelling.
Lessons	**Introduction to Repeat Loops** • Use repeat loops to run a section of code multiple times. **Loops: Catching Butterflies** • Use "show" and "hide" blocks and loops to create a butterfly-catching game. **Forever Loop Dance Party** • Create a sequence using a “repeat forever” loop to make characters repeat actions.

Unit 5: Message Events (7 weeks)

In this unit, students explore how characters can communicate through message events in ScratchJr. They build interactive stories and animations that rely on coordination between characters, while also applying debugging strategies and advanced block features like speed, size, and visibility changes.

Objectives / Topics Covered	• Use message events to control when and how characters interact. • Coordinate actions between multiple characters using broadcast messages. • Combine message events with visual effects like speed, size, and visibility. • Identify and fix bugs in multi-step programs. • Create original programs that demonstrate character communication and storytelling.

Lessons	**Introduction to Message Events** ● Program a relay race that uses messages to cause characters to interact. **Message Events: Simon Says** ● Use message events to make one character communicate to many characters in a program. **Debugging** ● Describe what bugs are and find and correct bugs in sequences. **Create an Original Story Animation** ● Create a program to animate an original story. **Speed Block: Bouncy Ball** ● Use speed blocks and messages to program a character to move at different speeds. **Garden Project (2-part lesson)** ● Use message events, grow, shrink, hide, and show blocks to animate seeds growing in a garden.

Unit 6: Pages (3 weeks)

In this unit, students use pages and navigation blocks to create interactive, multi-scene programs. They apply sequencing, loops, and message events to build simple games and animations that reflect real-world themes and digital storytelling.

Objectives / Topics Covered	● Use “go to page” blocks to navigate between scenes. ● Create multi-page programs with interactive elements. ● Combine loops and message events to build simple games. ● Demonstrate understanding of technology’s role in the world through programming.
Lessons	**Pages: Create a Tapping Game** ● Create a game that moves from one page to the next using "go to page" blocks. **Create a Mini Golf Game** ● Use messages and loops to create a mini golf game. **Impacts of Technology in Our World** ● Create a program to demonstrate how technology impacts our world.

Unit 7: Grid (2 weeks)

In this unit, students use the ScratchJr grid to plan and control character movement with precision. They design and navigate mazes, apply directional logic, and use the “end” block to complete animations with a clear stopping point.

Objectives / Topics Covered	● Use the grid to plan and program movement across the stage. ● Apply directional sequencing to navigate mazes. ● Use the “end” block to signal the completion of an animation.
Lessons	**Grid: Solving Mazes** ● Design a maze and use the grid to program a character to move through the maze. **End Block: Program a Race** ● Complete a project to program a race and use the “end” block in an animation.

Unit 8: Culmination Projects (6 weeks)

In this unit, students synthesize what they've learned across the course to create original projects that demonstrate their understanding of sequencing, events, loops, and messaging. These final programs allow students to showcase their creativity, storytelling, and problem-solving skills.

Objectives / Topics Covered	● Plan and create original programs that reflect personal interests and storytelling. ● Use messages, loops, and sequences to coordinate character actions. ● Apply prior knowledge to build interactive games with adjustable difficulty. ● Demonstrate mastery of multi-page navigation and event-based programming.
Lessons	**About Me Project (2-part lesson)** ● Plan a program to describe their characteristics and interests. **River Crossing Game (2-part lesson)** ● Program obstacles in a game and change the level of difficulty using speed blocks. **Space Travel Project (2-part lesson)** ● Create and explore ways to program a multipage story using messages, loops, and sequences.

Unit 9: Digital Literacy (6 weeks)

In this unit, students build foundational digital literacy skills by learning how data, artificial intelligence, online safety, and networks affect the way we use and interact with technology. Through hands-on programming and research projects, students explore how to use digital tools responsibly and effectively.

Objectives / Topics Covered	● Identify real-world applications of artificial intelligence and how it uses data. ● Collect and present data through visual programming. ● Conduct guided research and communicate findings through code. ● Understand the importance of secure usernames and passwords. ● Explain how networks and the Internet allow people to share information.
Lessons	**What Can AI Do?** ● Identify tools that use AI, explain that AI uses data to learn and make decisions, and compare tasks that are better suited for humans versus AI. **Basic Data and Programming** ● Collect data and create a program to present their data visually. **Guided Research (2-part lesson)** ● Find information using research sources and create a program to communicate their research visually. **Password Protectors** ● Understand the importance of usernames and passwords and demonstrate strategies to keep login information safe. **Networks and the Internet** ● Explain what a network is and how people communicate over networks and the Internet; model how messages are communicated using the Internet.

Unit 10: Optional Interdisciplinary (8 weeks)

In this optional unit, students integrate computer science with core subjects—math, science, ELA, and social studies—by creating interactive ScratchJr projects that reinforce academic concepts through coding.

Objectives / Topics Covered	● Use sequences, events, and loops to model academic concepts in multiple subjects. ● Represent math strategies and story problems through interactive animations.

	• Apply coding to demonstrate scientific cycles and observable patterns. • Reinforce literacy skills through phonics and sentence-building programs. • Explore civic and economic concepts through character interactions and choices.
Lessons	**Place Value: Adding Up to 20** • Use events in ScratchJr to illustrate how to decompose a two-digit number into tens and ones. **Story Problems: Add and Subtract within 20** • Use events to create a scene that represents an addition or subtraction story problem. **Sun and Moon, Day and Night** • Use loops to model the movements of the sun and moon and show the pattern of day and night. **Animal Life Cycles** • Use message events to model the butterfly life cycle. **Phonics: Digraphs** • Create a phonics program with digraphs using events and recordings. **Build a Sentence** • Create an interactive program that uses events to write sentences and then read them aloud. **Our Responsibilities** • Use sequences to program two characters to explain how to be responsible in school and at home. **Economic Choices** • Use message events to cause character interaction and describe how people make choices between wants and needs.

1st Grade Course Supplemental Materials

Resources	**Description**
Parent Welcome Letter (Spanish)	Send this letter home to introduce families to computer science with CodeHop.
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**.	