Tennessee 5th Grade Computer Science Course Syllabus

One Year for Elementary School, 36 Hours

Course Overview and Goals

The **Tennessee 5th Grade Computer Science** introduces students to foundational programming concepts through **Scratch**, 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, cross-curricular integration, and digital literacy.

Learning Environment: This course is designed to be teacher-led, with ready-to-use lesson plans structured as **Introduction, Guided Practice, Independent Practice, Extension, and Reflection**. Lessons are delivered in an "I do, we do, you do" format, supporting gradual release of responsibility and fostering confidence as students learn.

Unique to Tennessee: The pathway is structured as an approximate 36-hour sequence that balances computer science skill development, interdisciplinary integration, and digital literacy. Core CS modules spiral across grade levels, while interdisciplinary lessons are organized in subject-area menus (Math, Science, and ELA) that teachers can weave in based on local instructional needs. Digital literacy lessons are embedded within the sequence, ensuring students also build essential technology and digital citizenship skills alongside programming.

The full course includes 36 lessons, each approximately 45 minutes, offering a complete school year if taught once per week.

Programming Environment: Students will write and run programs in **Scratch** embedded and 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/26974/overview.

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

Course Breakdown

Unit 1: Optional Review (2 weeks)

This optional unit helps students refresh foundational skills in Scratch and the CodeHop Playground while applying concepts such as the coordinate plane and basic programming vocabulary.

Objectives / Topics Covered	● Log in and navigate the CodeHop Playground. ● Define computer science terms and create a basic Scratch program. ● Use the coordinate plane to program animations.
Lessons	**Welcome to CodeHop! (15-minute lesson)** ● Log in and navigate the CodeHop Playground. **The Coordinate Plane** ● Create an open-ended animation using the coordinate plane in Scratch. **Introduction to Computer Science and Scratch** ● Define important computer science vocabulary and create a simple Scratch program.

Unit 2: Getting Started (3 weeks)

In this unit, students build foundational skills by learning how computing systems work, exploring positive digital citizenship, and applying computational thinking through a creative design activity.

Objectives / Topics Covered	● Identify computing system components and simple hardware/software problems. ● Recognize and respond to different forms of cyberbullying. ● Apply computational thinking to design a neighborhood.
Lessons	**Practicing with Computing Systems** ● Identify parts of the computing system and identify simple hardware and software problems. **Standing Up to Cyberbullying** ● Recognize different types of online hurtful behavior, including cyberbullying, and describe ways to respond or take responsibility. **Computational Thinking: Design a Neighborhood** ● Use computational thinking to design a neighborhood.

Unit 3: Sequences, Events, & Loops (5 weeks)

In this unit, students explore how sequences, events, and loops work together to create animations and interactive programs. They build and refine algorithms, debug programs, and apply looping structures to animations and turtle graphics.

Objectives / Topics Covered	● Compare and refine algorithms for clarity and efficiency. ● Decompose programs to debug and improve functionality. ● Use loops to animate scenes and create patterns. ● Apply event-driven programming concepts in Scratch.
Lessons	**Compare and Refine Algorithms** ● Compare and refine multiple algorithms for the same task to determine which is the most appropriate and efficient. **Debugging: Make a Pizza** ● Decompose a program to debug and make the program run as intended.

	Creating Turtle Graphics ● Use the pen tool in Scratch to create looping turtle graphics. **Animation Loops Project (2-part lesson)** ● Use repeat loop blocks to program an animation with multiple scenes.

Unit 4: Conditionals & Operators (2 weeks)

In this unit, students learn to build interactive programs using conditional logic and operators. They explore if/then and if/then/else structures and apply math-based operators to make decisions in their projects.

Objectives / Topics Covered	● Create programs using if/then conditionals. ● Use operators and variables to control program behavior. ● Apply logical reasoning to predict outcomes of condition-based programs.
Lessons	**Conditionals: Mazes** ● Create a program that uses conditionals. **Operators: Coin Flip** ● Create a coin flipping program using variables and operators.

Unit 5: Variables & Lists (3 weeks)

In this unit, students explore how to store, update, and organize data using variables and lists in Scratch. They apply these skills to create interactive and customizable programs.

Objectives / Topics Covered	● Use variables to store and update values. ● Apply lists to organize and retrieve information. ● Combine variables, lists, and operators in a program.
Lessons	**Variables in Dance** ● Use variables to control pitch and dance speeds in a program. **Lists: Shopping with Scout (2-part lesson)** ● Create a shopping simulator using variables, lists, and operators.

Unit 6: Clones & Functions (2 weeks)

In this unit, students explore how to reuse code efficiently using functions with inputs and how to duplicate sprites with clones to create more interactive projects.

Objectives / Topics Covered	● Create and use functions with inputs. ● Use clones to duplicate sprites and manage multiple instances. ● Apply cloning and functions in interactive programs.
Lessons	**Clones: Throwing Acorns Game** ● Create a throwing acorns game using clones. **Functions: About Me** ● Create and use a function with input in a program.

Unit 7: Culmination Projects (6 weeks)

In this unit, students apply design thinking and programming concepts to develop original projects. Students will design a game and an app that demonstrate their coding and problem-solving skills.

Objectives / Topics Covered	● Design a game using loops, conditionals, and variables. ● Apply the design thinking process to develop an app for a real-world need.

Lessons	**Game Design Project (3-part lesson)** ● Design and create a game using multiple programming skills such as loops, conditionals, and variables. **Design an App (3-part lesson)** ● Use the design thinking process to design an app that helps to solve a user's need.

Unit 8: Digital Literacy (5 weeks)

This unit helps students navigate digital tools and online spaces safely and responsibly. Students explore how to give credit, analyze data using spreadsheets, and conduct inquiry-driven research projects.

Objectives / Topics Covered	● Practice giving credit and using attributions in digital work. ● Use search engines effectively and evaluate online sources. ● Organize and visualize data using spreadsheets. ● Conduct and present a digital inquiry project.
Lessons	**Giving Credit Through Attributions** ● Give appropriate attribution when creating or remixing programs online. **Use and Search the Right Way** ● Search for information to answer questions online and provide proper attribution to sources. **Introduction to Microsoft Excel™/ Introduction to Google Sheets™** **Two versions of this lesson are provided depending on your software needs.* ● Enter, organize, and visualize data in a spreadsheet. **Inquiry Project: Line Graph (2-part lesson)** ● Follow the inquiry process and modify a program to display the results of their investigation.

Unit 9: Interdisciplinary Computer Science (11 weeks)

This unit provides interdisciplinary Scratch lessons that reinforce core concepts across math, science, ELA, music, and social studies. These are for flexible use throughout the year; within each subject area, lessons are listed in order of increasing coding complexity.

Objectives / Topics Covered	● Apply math operations like division and fractions using loops and conditionals. ● Model scientific processes such as ecosystems, pollution, and land formation. ● Use Scratch to support ELA learning through storytelling and grammar games. ● Research and present computer science innovators through programming.
Lessons	**Animating Sprites with Division** ● Use division to animate sprites with loops and wait blocks. **Decimal Division and Conditionals** ● Solve division problems with decimals and use conditionals to program an interactive division game with levels. **Add and Subtract Fractions** ● Use broadcast messages and comparison operators to create a fractions quiz game and recognize patterns in code. **Exploring Ecosystems** ● Program a model to illustrate the flow of energy in an ecosystem. **Constructive and Destructive Processes** ● Create an animation that models how volcanoes change surface features through a constructive process. **Effects of Pollution** ● Write a program with if/then/else blocks in Scratch to explain how actions can affect water pollution. **Punctuate a Title** ● Create a game using conditionals and operators to demonstrate understanding of

	punctuation in titles. **Creative Storytelling (2-part lesson)** ● Plan and animate a story using events and sequences. **CS Innovators Research Project (2-part lesson)** ● Research an innovator and abstract facts from an article to use as an informational program.

4th - 5th 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 Scratch Templates	Empower your students to explore and express their knowledge creatively with our versatile Scratch 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**.	