CodeHS

Utah Computer Science 3rd Grade Course Syllabus

One Year for Elementary School, 36 Hours

Course Overview and Goals

The **Utah Computer Science 3rd Grade Course** 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 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 **Scratch** 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/26171/overview

Course Breakdown

Optional Review

This brief review lesson helps students get comfortable logging in and navigating the CodeHS Playground.

Objectives / Topics Covered	● Log in and explore basic site features to prepare for upcoming lessons.
Lessons	**Welcome to CodeHS! (15 minute lesson)** ● Practice logging in and exploring the CodeHS Playground before starting a full lesson.

Unit 1: Getting Started (3 lessons)

In this introductory unit, students explore how computers work, what to do when something goes wrong, and how to stay safe online. They'll identify key parts of computing systems, apply troubleshooting steps, and think like computer scientists by using computational thinking to creatively design a school.

Objectives / Topics Covered	● Identify and describe the components of a computing system and how they work together. ● Apply basic troubleshooting steps to solve common computer problems. ● Understand and describe simple cybersecurity threats and ways to stay safe online. ● Use computational thinking to break down problems, recognize patterns, and design a solution.
Lessons	**Introduction to Computing Systems and Troubleshooting** ● Learn about the parts of a computer system and how they work together, then apply simple strategies to fix common hardware and software problems. **Scout's Cybersecurity Adventure: Part 1** ● Explore basic cybersecurity concepts, identify online threats, and discover safe practices for using technology. **Computational Thinking: Design a School** ● Use computational thinking to plan and design a new school by breaking the problem into parts, finding patterns, and sequencing ideas logically.

Unit 2: Scratch Exploration (5 lessons)

In this story-driven unit, students will follow Scout on an animated adventure as they learn to use Scratch. They'll develop core programming skills like sequencing, using events and loops, and customizing their animations with drawing tools—all while creating interactive stories.

Objectives / Topics Covered	● Use Scratch to program characters with motion, dialogue, and animations. ● Sequence blocks to create interactive stories. ● Apply events, loops, and looks blocks to enhance animations. ● Customize characters and backgrounds using Scratch's drawing tools.
Lessons	**Scout's Scratch Expedition Part 1** ● Use basic Scratch blocks to program Scout the sprite to move and talk in the beginning of the story. **Scout's Scratch Expedition Part 2** ● Continue the story by adding sprites and creating a sequence that animates a simple scene. **Scout's Scratch Expedition Part 3** ● Create a looping animation using events, looks, and motion blocks as the story becomes more interactive.

	Scout’s Scratch Expedition Part 4 ● Add complexity to the story by combining events, motion, and looks to animate multiple characters. **Scratch Drawing Tools** ● Explore Scratch’s built-in drawing tools to design custom sprites and backdrops for your projects.

Unit 3: Sequences & Events (6 lessons)

In this unit, students will deepen their understanding of how sequences and events work in Scratch. They’ll explore parallel programming, animated sprites, interactive backdrops, and broadcast messages to create engaging and dynamic programs—including an original story that brings their ideas to life.

Objectives / Topics Covered	● Create programs using sequences, events, and parallel actions. ● Animate sprites and switch backdrops to enhance storytelling. ● Use broadcast messages to coordinate actions between characters. ● Plan and program original stories with multiple interactive elements.
Lessons	**Sequences: Parallel Programming** ● Create a program where multiple sprites perform actions at the same time using parallel sequences. **Events** ● Use events to trigger actions in a program, such as when the green flag is clicked or a sprite is tapped. **Costumes, Backdrops, and Animations** ● Animate sprites by changing costumes and switching between backdrops to create interactive scenes. **Broadcast Messages: Marco Polo** ● Use broadcast and receive blocks to trigger communication between sprites in a Marco Polo–style animation. **Creative Storytelling (2 day lesson)** ● Plan and animate a custom story using sequences and events to bring characters and scenes to life.

Unit 4: Loops (4 lessons)

In this unit, students explore how loops help simplify code by repeating actions. They'll practice using loops in Scratch to build efficient programs, work through debugging challenges, and collaborate with a partner to create an interactive digital band using keyboard inputs.

Objectives / Topics Covered	● Understand how loops repeat instructions in a program. ● Use loops in Scratch to simplify and organize code. ● Debug programs that involve loops and events. ● Collaborate with peers through pair programming to build interactive projects.
Lessons	**Loops** ● Learn how a loop repeats instructions and use loops to animate or control characters in a Scratch program. **Debugging: Events and Loops** ● Break down a program into smaller parts to find and fix errors involving events and loops. **Pair Programming: Create a Band (2 day lesson)** ● Work with a partner to design and code a band using keyboard inputs and loops to make instruments play.

Unit 5: Conditionals (5 lessons)
In this unit, students dive into conditionals to learn how computers make decisions. They'll use if/then and if/then/else blocks to control what happens under certain conditions in a program. Alongside coding practice, students will explore how to remix projects responsibly and give proper credit to original creators.

Objectives / Topics Covered	● Understand and use conditionals to control program behavior. ● Create programs that respond to specific conditions using if/then and if/then/else blocks. ● Develop awareness of copyright, plagiarism, and giving credit when remixing digital content.
Lessons	**Introduction to Conditionals** ● Learn what a conditional is and build a program that uses if/then blocks to make decisions. **Conditionals: Color Sense** ● Use conditionals to trigger actions when specific colors are touched in a program. **Conditionals: Changing Effects** ● Create a program that uses if/then logic to change how sprites behave or react. **Complex Conditionals: Balloon Game** ● Explore if/then/else conditionals by building a balloon-popping game that responds to different outcomes. **Remixing Programs** ● Create or remix a Scratch project while practicing digital citizenship by giving credit to original creators and understanding copyright basics.

Unit 6: Variables & Lists (4 lessons)
In this unit, students explore how variables and lists are used to store and manage information in Scratch. They'll build programs that track values, make comparisons using conditions, and create an I-Spy–style game that uses both variables and lists to make projects more interactive.

Objectives / Topics Covered	● Understand what a variable is and how it can be used to store changing information. ● Use comparison operators in conditionals with variables and numbers. ● Create and use lists to organize and manage multiple pieces of information. ● Apply variables and lists to build interactive games and animations.
Lessons	**Variables** ● Create a program that uses a variable to store and change a value, such as keeping score. **Introduction to Comparison Operators** ● Use comparison operators (>, <, =) with numbers and variables to build conditional logic using if/else blocks. **I-Spy with Lists (2 day lesson)** ● Design an I-Spy–style game in Scratch using lists to track hidden objects and variables to manage interactions.

Unit 7: Functions (3 lessons)
In this unit, students are introduced to functions as a way to organize and reuse code. They'll learn how to create custom blocks in Scratch to simplify their programs and build a fun dance project that uses functions to control movement in sync with music.

Objectives / Topics	● Understand what a function is and how it helps organize code. ● Create and use custom blocks (functions) in Scratch.

Covered	● Apply functions to simplify repeated actions in a sequence. ● Build a creative project that uses functions to control timing and choreography.
Lessons	**Introduction to Functions** ● Create and use custom blocks (functions) to organize and reuse code in a Scratch project. **Functions Dance Project (2 day lesson)** ● Build a dance animation by creating functions for different moves and sequencing them to match music.

Unit 8: Culmination Project (5 lessons)

In this final unit, students apply the skills they've learned throughout the course to complete creative, inquiry-based projects. From exploring the legacy of a computer science pioneer to building a racing game and visualizing survey results, students demonstrate mastery of key concepts like conditionals, variables, lists, and data representation.

Objectives / Topics Covered	● Reflect on the impact of historical figures in computer science. ● Apply core programming skills—like conditionals, variables, and lists—to design interactive projects. ● Use the inquiry process to collect, analyze, and display data through coding. ● Create original programs that combine creativity, coding, and real-world problem solving.
Lessons	**CS Innovators: Grace Hopper** ● Learn about Grace Hopper's contributions to computer science and decode mystery words using binary. **Race Track Game (2 day lesson)** ● Design a race car and track, then program a racing game that includes scorekeeping with conditionals, variables, and lists. **Inquiry Project: Survey Bar Graph (2 day lesson)** ● Follow the inquiry process to investigate a topic and modify a Scratch program to display survey results using a bar graph.

Utah Computer Science 3rd 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 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.
All of these resources and more are found on the **Elementary Resources Page**.	