California 4th Grade
Computer Science Course Syllabus
One Year for Elementary School, 36 Hours

Course Overview and Goals

The **California 4th Grade Computer Science 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 **contact hours**, with each lesson approximately 45 minutes long. This provides a full school year of material if teaching one lesson per week.

Programming Environment: Students will write and run programs in **Scratch** embedded and saved in the 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/26165/overview?lang=en

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

Course Breakdown

Unit 1: Optional (4 optional lessons)

Students are optionally introduced to the platform and can build foundational skills in Scratch by exploring basic computer science vocabulary and tools. They'll create simple programs, design custom sprites and backdrops, and use the coordinate plane to animate creative scenes.

Objectives / Topics Covered	● Getting started with CodeHop ● Review Scratch topics
Lessons	**Welcome to CodeHop!** ● 15-minute lesson - combine with the next lesson if time allows. ● Explore the Playground and learn how to log in. **Introduction to Computer Science and Scratch** ● Define important computer science vocabulary and create a simple program in Scratch. **The Coordinate Plane** ● Create an open-ended animation using the coordinate plane in Scratch. **Scratch Drawing Tools** ● Create customized sprites and backdrops using the drawing tools.

Unit 2: Sequences & Events (6 lessons)

In this Sequences & Events unit, students will explore how sequences, events, and algorithms drive program behavior and communication between sprites. They'll build interactive projects using event blocks and broadcast messages, investigate real-world applications of coding in sports and innovation, and practice abstracting information to create timelines and informational programs.

Objectives / Topics Covered	● Events and sequences ● Use broadcast messages ● Create a timeline ● Program and compare multiple algorithms ● Research a computer science innovators
Lessons	**Careers in CS: Major League Baseball** ● Explain how coding can be used in sports, and abstract events from an article to retell important events in a timeline program. **Creating Algorithms** ● Program multiple algorithms and assess which one best meets their needs. **Pair Programming Create a Band (2 lessons)** ● Collaborate through pair programming to design and code a band in Scratch using keyboard inputs. **Broadcast Messages: Tell a Joke** ● Use broadcast messages to program two sprites to tell a knock-knock joke.

Unit 3: Loops (2 lessons)

In this Loops unit, students will strengthen their understanding of repetition in programming by using different types of loops to build interactive projects. They will also practice decomposing and debugging code to ensure their programs run as intended.

Objectives / Topics Covered	• Loops in programming
Lessons	**Loops: Catch the Ball** • Use two types of loops to create a simple game in Scratch. **Debugging: Mazes** • Decompose a program to debug and make the program run as intended.

Unit 4: Conditionals & Operators (4 lessons)

In this unit, students will enhance their programs by using conditional statements and interactive inputs. Through projects like a maze game, a drawing app, and feedback-driven game updates, students will apply if/then logic and user controls to create dynamic, engaging experiences.

Objectives / Topics Covered	• Use if/then conditionals and operators • Use keyboard and mouse inputs
Lessons	**Create a Maze (2 lessons)** • Draw a maze backdrop in Scratch and program Scout to navigate through the maze. **Create a Drawing App** • Create a drawing app by programming keyboard and mouse inputs, loops, and conditional statements. **Scout's Quest: Conditionals** • Create a program using if/then conditionals. *Part 4 of 4 in Scout's Quest skill review series.*

Unit 5: Variables & Lists (4 lessons)

In this unit, students will create games like a spelling bee and a randomized object game, applying key concepts such as tracking points, storing data, and using object properties to enhance gameplay.

Objectives / Topics Covered	• Variables to track points • Lists • Use classes, objects, and randomizers
Lessons	**Scout's Quest: Variables** • Create and use variables to track points in a program. *Part 2 of 4 in Scout's Quest skill review series.* **Lists: Spelling Bee** • Use lists to create a spelling bee game. **Classes and Objects in Games (2 lessons)** • Create an interactive game and use randomizers to change the characteristics of objects.

Unit 6: Clones & Functions (3 lessons)

In this Clones & Functions unit, students will explore how to create reusable and dynamic code using cloning and functions. They'll build animations with clones and develop programs using functions with boolean and number inputs to control behavior and create interactive outputs.

Objectives / Topics Covered	● Clones ● Boolean Inputs ● Functions
Lessons	**Introduction to Clones** ● Create an animation using clones and investigate the limitations of their program. **Scout's Quest: Functions with Boolean Inputs** ● Create a function including a boolean input to perform different actions based on whether a password is correct. *Part 1 of 4 in Scout's Quest skill review series.* **Scout's Quest: Functions with Number Inputs** ● Create a drawing using functions with number inputs. *Part 3 of 4 in Scout's Quest skill review series.*

Unit 7: Culmination Projects (6 lessons)

In this unit, students will apply advanced programming concepts to build interactive, personalized applications. Through projects like designing an AI chatbot and creating a custom music player, they'll use lists, variables, operators, and conditionals to develop programs that respond to user input and showcase creativity.

Objectives / Topics Covered	● Culmination projects
Lessons	**Program an AI Chatbot (2 lessons)** ● Two-part project, use lists to create a chatbot to store prompts, responses, and answer questions. **Code Tunes (2 lessons)** ● Two-part project, use variables, operators, and conditionals to create their own custom music player in Scratch. **Designing Solutions for Accessibility (2 lessons)** ● In this two-part project, use the design thinking process to identify and solve real-world problems by redesigning a game to improve accessibility and usability for diverse users.

Unit 8: Digital Literacy (12 lessons)

In this Digital Literacy unit, students will explore how to responsibly navigate, communicate, and create in the digital world. They'll conduct research, visualize data, examine the evolution of technology, practice safe online habits, and learn to give proper credit—all while building meaningful digital projects like PSAs, timelines, and secure communication models.

Objectives / Topics Covered	• Follow the inquiry process • Networks and the internet • Attribution and giving credit • Create a timeline • Digital citizenship and digital safety • Create 3D designs
Lessons	**Technology Timeline** • Create an interactive timeline to illustrate the key developments in music player technology. • Explain how music player technology has influenced cultural practices. **Scout's Cybersecurity Adventure: Part 2** • Demonstrate how to stay safe online by practicing secure habits and understanding the tools and technologies that protect your information. **Exploring Computing Systems** • Identify parts of the computing system and identify simple hardware and software problems. **File Management and Data in Action** • Explain that different types of digital data take up different amounts of space and can be stored in different ways. **Giving Credit When You Use It** • Search for information to answer questions online and provide proper attribution to sources. **Networks, Packets, and the Internet** • Explain how information is communicated through the Internet. • Model how communication is broken into smaller pieces, transmitted as packets, and reassembled at the destination. • Design and implement a secure communication method within the classroom. **Inquiry Project: Line Graph** • In this two-part project, follow the inquiry process and modify a program to display the results of their investigation. **3D Design: Keyboard Accommodations** • In this two-part project, use the align tool to position shapes together as they create an accessible keyboard in Tinkercad®. • *This lesson requires student accounts on an external site.* **3D Design: Recreate an Animal** • In this two-part project, add, move, scale, and rotate shapes in Tinkercad® to create a 3D model of an animal. • *This lesson requires student accounts on an external site.*

California 4th Grade Computer Science Course
Course Supplemental Materials

Resources	Description
Parent Welcome Letter	Introduce parents 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	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	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	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**.	