California 3rd Grade
Computer Science Course Syllabus
One Year for Elementary School, 36 Hours

Course Overview and Goals

The **California 3rd Grade Computer Science Course** introduces students to foundational programming concepts through Codehop, 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.

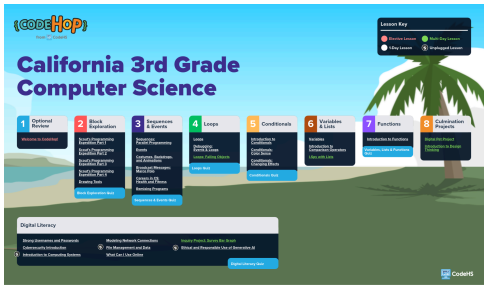
The lessons are delivered in an **"I do, we do, you do"** format, ensuring a gradual release of responsibility and fostering confidence in students as they learn. Teachers can adapt the content to fit their schedule and instructional needs. The concepts taught in this course spiral across grade levels, 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 **CodeHop** 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.

Assessments: Some units conclude with a brief two-question quiz designed to assess student understanding of the concepts and skills introduced in the unit. Quizzes can be assigned digitally, completed as a QuizBuzz activity, or printed for paper-based completion.

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/26162/overview



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

Course Breakdown

Unit 1: Optional (1 optional lesson)

Students are optionally introduced to the platform.

Objectives / Topics Covered	● Logging in and navigating the platform
Lessons	**Welcome to CodeHop!** ● 15-minute lesson - combine with the next lesson if time allows. ● Explore the playground and learn how to log in.

Unit 2: Block Exploration (5 lessons)

Students begin programming through a sequential, story-driven unit featuring Scout the Squirrel. Students learn foundational programming concepts such as sequences, events, and basic animations.

Objectives / Topics Covered	● Basic programming commands ● Use sequences and events
Lessons	**Scout's Programming Expedition Part 1:** ● Program a sprite to move and talk using basic CodeHop commands. **Scout's Programming Expedition Part 2:** ● Add sprites and create a sequence to animate a simple story. **Scout's Programming Expedition Part 3:** ● Develop a looping animation using events, looks, and motion blocks. **Scout's Programming Expedition Part 4:** ● Enhance the animated story with advanced events and motion blocks. **Drawing Tools: Sea Creatures** ● Create customized sprites and backdrops using CodeHop's built-in drawing tools. **Block Exploration Quiz (5 questions)**

Unit 3: Sequences & Events (6 lessons)

Students dive deeper into programming concepts, learning how to create interactive programs using sequences and events. This unit emphasizes creative expression through animations and storytelling while introducing key concepts like broadcast messages and pair programming.

Objectives / Topics Covered	● Collaborate with peers ● Animate sprites in programs
Lessons	**Sequences: Parallel Programming** ● Create a program using parallel sequences. **Events** ● Design a program where events (e.g., keypress or sprite clicks) trigger specific actions. **Costumes, Backdrops, and Animations** ● Create animated sprites and interactive backdrops using costume changes and motion blocks. **Broadcast Messages: Marco Polo** ● Use broadcast and receive blocks to trigger actions between sprites in a "Marco Polo" game. **Careers in CS: Health and Fitness** ● Explain how fitness and coding can amplify human strengths and create an animation to demonstrate how technology can be used in health and

	fitness. **Remixing Programs** ● Remix digital programs and give appropriate credit to original creators. **Sequences & Events Quiz (5 questions)**

Unit 4: Loops (5 lessons)

Students explore how loops enable efficiency in programming by repeating commands. This unit introduces the concept of loops both through unplugged activities, emphasizing their application in animations and debugging.

Objectives / Topics Covered	● Loops in programming
Lessons	**Loops (Unplugged)** ● Explain that a loop repeats a command or a set of commands and interpret the control flow of a program. **Loops** ● Create a program that uses loops to repeat actions, such as animating a sprite's movement. **Debugging: Events and Loops** ● Decompose a pre-written program to identify and fix errors involving loops and events. **Loops: Falling Objects** (2 lessons) ● In this two-part project, students develop a program with falling objects using different loop structures and compare the efficiency of each algorithm. **Loops Quiz (5 questions)**

Unit 5: Conditionals (3 weeks)

Students learn how to use conditionals in programming to make decisions based on specific criteria. This unit covers basic if/then blocks, advanced if/then/else logic, and applies these concepts in creative and interactive projects.

Objectives / Topics Covered	● Use conditionals in programming ● Using if/then blocks
Lessons	**Introduction to Conditionals** ● Explain and demonstrate conditionals by creating a program using if/then blocks. **Conditionals: Color Sense** ● Program a sprite to respond to specific colors using conditionals. **Conditionals: Changing Effects** ● Create a program that alters a sprite’s effects (e.g., color or size) based on conditions. **Conditionals Quiz (5 questions)**

Unit 6: Variables & Lists (4 lessons)

Students are introduced to variables and lists, learning how to store, manipulate, and use data in their programs. This unit explores comparison operators and creative applications of lists.

Objectives / Topics Covered	● Variables in programs ● Comparison operators ● Lists to store data
Lessons	**Variables** ● Explain what a variable is and create program to store and change values.

	Introduction to Comparison Operators ● Use comparison operators (e.g., >, <, =) in if/else blocks, incorporating numbers and variables. **I-Spy with Lists** (2 lessons) ● In this two-part project, create an interactive I-Spy game where players search for objects, utilizing lists to store and retrieve data.

Unit 7: Functions (1 week)

Students learn how to create and use functions to simplify their programs and make their code reusable.

Objectives / Topics Covered	● Functions in programs to simplify and organize
Lessons	**Introduction to Functions** ● Create a program with custom functions and use them in sequences. **Variables, Lists & Functions Quiz (5 questions)**

Unit 8: Culmination Projects (3 lessons)

Students synthesize their learning from previous units by creating a digital pet project. This capstone project incorporates multiple programming concepts and allows students to showcase their creativity and technical skills.

Objectives / Topics Covered	● Culminating creative projects
Lessons	**Digital Pet Project** (3 lessons) ● In this three-part project, students design and create a digital pet that responds to user interactions. ● Use variables to track the pet's "needs" (e.g., hunger, happiness). ● Implement conditionals to determine the pet's behavior based on its state. ● Use broadcast messages to trigger actions or animations in the program.

Unit 9: Digital Literacy (9 lessons)

In this unit, students will develop essential digital citizenship skills by exploring how computers work, how to stay safe and secure online, and how to use digital tools responsibly. Through hands-on projects and guided discussions, they'll build knowledge in cybersecurity, file management, research strategies, and ethical technology use, including generative AI.

Objectives / Topics Covered	● Inputs and outputs in computing systems ● Hardware and software problems ● Explore digital identity and footprint ● Online Safety ● Source relevance and keywords ● Machine Learning
Lessons	**Cybersecurity Introduction** ● Understand basic cybersecurity concepts, identify common cyber threats, and explain practical tips for staying safe online. **Modeling Network Connections** ● Describe how messages move between devices using a network and create a program that shows how data travels from one device to another. **File Management and Data** ● Explain that different types of digital data take up different amounts of space, and identify where digital data can be stored. **What Can I Use Online?**

	• Search for information to answer questions online and provide proper attribution to sources. **Inquiry Project: Survey Bar Graph** • In this two-part project, follow the inquiry process and modify a program to display the results of their investigation. **Ethical and Responsible Use of Generative AI** • Describe the pros and cons of generative AI. • Complete a class Code of Conduct to follow when using AI. **Digital Literacy Quiz (5 questions)**

California 3rd 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 Templates	Empower your students to explore and express their knowledge creatively with our versatile 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**.	