California 5th Grade
Computer Science Course Syllabus

One Year for Elementary School, 36 Hours

Course Overview and Goals

The **California 5th Grade Computer Science Course Syllabus** 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/26166/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	● Explore the CodeHop platform ● Create programs in Scratch ● Utilize different Scratch tools
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 learn how to use events to control program flow and build meaningful, interactive stories. They'll practice debugging by decomposing code and applying their knowledge to create a program that shares how a winter holiday is celebrated around the world.

Objectives / Topics Covered	● Create and decompose in Scratch ● Understand influential people in CS ● Learn about different winter celebrations
Lessons	**Events: Traveling with Scout** ● Use events in a program. **Debugging: Make a Pizza** ● Decompose a program to debug and make the program run as intended. **CS Innovators Research Project (2 lessons)** ● In this two-part project, research an innovator and abstract facts from an article to use as an informational program. **Winter Celebrations Around the World** ● Learn about different winter celebrations. ● Create a program to tell how a specific holiday is celebrated. **Compare and Refine Algorithms** ● Compare and refine multiple algorithms for the same task to determine which is the most appropriate and efficient.

Unit 3: Loops (2 lessons)

In this Loops unit, students will use loops and the pen tool in Scratch to create turtle graphics, reinforcing how repetition can generate patterns and artistic designs through code.

Objectives / Topics Covered	● Practice using Loops
Lessons	**Creating Turtle Graphics** ● Use the pen tool in Scratch to create looping turtle graphics. **Game Effects**

Unit 4: Conditionals & Operators (4 lessons)

In this Conditionals & Operators unit, students will use if/then logic and operators to create responsive, decision-based programs. Through projects like mazes, quests, and a coin flip simulation, they'll apply conditional statements, variables, and generalized algorithms to control outcomes and structure program behavior.

Objectives / Topics Covered	● Use Conditionals and Operators in programs ● Plan and decompose programs
Lessons	**Conditionals: Mazes** ● Create a program that uses conditionals. **Plan a Quest (2 lessons)** ● Plan and decompose the steps needed to create a quest program. **Operators: Coin Flip** ● Create a coin flipping program using variables and operators.

Unit 5: Variables & Lists (4 lessons)

In this Variables & Lists unit, students will explore how to use variables, lists, and operators to manage data and control game behavior. Through projects like a dance simulator, a shopping game, and a multi-part pinball game, they'll apply comparison logic, store and manipulate data, and build fully interactive experiences.

Objectives / Topics Covered	● Use variables, lists, and operators
Lessons	**Variables in Dance** ● Use variables to control pitch and dance speeds in a program. **Game Mechanics with Comparison Operators** ● Use comparison operators and variables to create ending game mechanics. **Lists: Shopping with Scout (2 lessons)** ● In this two-part project, create a shopping simulator using variables, lists, and operators.

Unit 6: Clones & Functions (2 weeks)

In this **Clones & Functions** unit, students will learn how to use clones to efficiently manage repeated elements in games and explore how functions with inputs can organize and simplify code. Through engaging projects like a throwing acorns game, an endless runner, and a personalized “About Me” program, students will build dynamic, reusable code structures.

Objectives / Topics Covered	● Use clones ● Use functions with inputs
Lessons	**Clones: Throwing Acorns Game** ● In this three-part project, 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 lessons)

In this unit, students will apply advanced programming concepts to build interactive, personalized applications. Through projects like designing an app and a house design project.

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

Unit 8: Digital Literacy (12 lessons)

In this Digital Literacy unit, students will explore how data, networks, and cybersecurity shape the digital world around them. Through hands-on projects and real-world examples, they will analyze data, learn about digital pioneers like Grace Hopper, understand how information travels securely online, and practice responsible online behavior, research, and attribution.

Objectives / Topics Covered	● Binary coding ● File management and data ● Network and protocols ● Digital literacy and digital citizenship: social media, cyberbullying, cybersecurity, responsible digital research and attribution
Lessons	**CS Innovators: Grace Hopper** ● Explain how Grace Hopper’s work was important to computer science, and use binary code to decompose mystery words. **Stand Up to Cyberbullying** ● Recognize different types of online hurtful behavior, including cyberbullying, and describe ways to respond or take responsibility. **Cybersecurity Policies and Laws** ● Explain policies and how they relate to their classroom or school. ● Research and explain a cybersecurity law specific to their state. **Scout’s Cryptography Escape Room** ● Learn basic cryptography and solve multiple ciphers. **File Management and Data Exploration** ● Explain that different types of digital data take up different amounts of space and identify where digital data can be stored. **Practicing with Computing Systems** ● Identify parts of the computing system and identify simple hardware and software problems. **Networks and Protocols** ● Explain how network protocols enable data transfer and compare WiFi, wired, and cellular networks. **Use and Search the Right Way** ● Search for information to answer questions online and provide proper attribution to sources. **AI Chatbots** ● Describe how AI representations support reasoning. ● Answer questions by training a model AI chatbot program. **How Machines Learn** ● Explain the different machine learning approaches and create a classification system using a tree structure. **Inquiry Project: Data Bar Graph** ● In this two-part project, follow the inquiry process. ● Modify a program to display the results of an investigation.

California 5th Grade Computer Science Course Syllabus

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**.	