Utah Computer Science
2nd Grade Course Syllabus
One Year for Elementary School, 36 Hours

Course Overview and Goals

The Utah Computer Science 2nd Grade Course introduces students to foundational programming concepts through 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 teacher-led and includes ready-to-use lessons following a consistent structure: Introduction, Guided Practice, Independent Practice, Extension, and Reflection. Instruction follows an "I do, we do, you do" model and incorporates spiral review to reinforce concepts and build confidence over time.

The course includes 36 lessons, each approximately 45 minutes long, providing a full year of instruction when taught once per week. While the course allows for instructional flexibility, some lessons are required to fully meet state computer science standards and are clearly identified within the syllabus. All Digital Literacy lessons are required to ensure full standards alignment, as they address essential non-programming computer science concepts. All lessons are labeled with the specific standards they address to support planning and compliance.

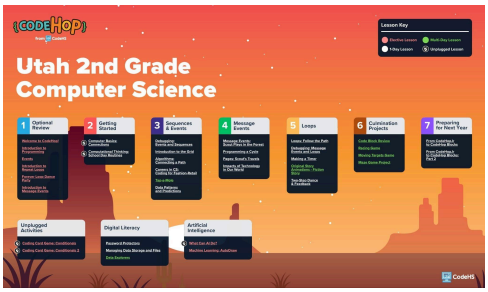
Standards Alignment Note: Standards addressed in each lesson are listed in parentheses after the lesson title. Standards may be reinforced across multiple lessons, and full coverage of state computer science standards is achieved through the course as a whole.

Programming Environment: Students will write and run programs that are saved in students' accounts. 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 four-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/26139/overview



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

Course Breakdown

Optional Review

This optional review unit offers extra practice for students who need reinforcement or a refresher on key programming concepts. It includes introductory lessons and creative activities that revisit sequences, events, loops, and message blocks in a flexible and engaging way.

Objectives / Topics Covered	● Review core programming concepts like events, loops, and message events. ● Strengthen comfort navigating the programming interface. ● Build confidence using basic blocks to create interactive programs. ● Provide flexible support for reteaching or extending learning.
Lessons	**Welcome to CodeHop! (15 minute lesson)** ● Introductory lesson to help students log in and explore the CodeHS Playground; ideal as a warm-up or standalone activity. **Introduction to Programming** (2.AP.1) ● Navigate the coding interface and create a scene with characters and background. **Events** (2.AP.1, 2.AP.4) ● Use multiple event blocks to make characters move, react, or interact based on user input. **Introduction to Repeat Loops** (2.AP.1, 2.AP.4) ● Learn how to use repeat loops to make actions happen more than once in a row. **Forever Loop Dance Party** (2.AP.1, 2.AP.4) ● Create a fun animation where characters continuously dance using the “repeat forever” loop. **Introduction to Message Events** (2.AP.4) ● Program a relay-style animation where characters take turns moving using send and receive message blocks.

Unit 1: Getting Started (2 lessons)

In this introductory unit, students will explore the basics of how computers work and how we use them. They’ll learn how different computer parts work together and use computational thinking skills to break down everyday routines into simple steps.

Objectives / Topics Covered	● Understand what a computer is and how it functions. ● Identify and explain how input, output, hardware, and software work together. ● Use computational thinking strategies like sequencing, pattern recognition, and simplification.
Lessons	**Computer Basics: Connections** (2.CS.1, 2.IC.1) ● Learn what a computer is, how it works, and how its parts (input, output, hardware, and software) work together. **Computational Thinking: School Day Routines** (2.AP.1, 2.AP.2, 2.AP.4) ● Break down a school day routine into steps while identifying patterns and simplifying tasks using computational thinking skills. **Getting Started Quiz (4 questions)**

Unit 2: Sequences & Events (7 lessons)

In this unit, students will build on their understanding of sequences and events by programming characters to follow paths, respond to taps, and interact with data. They’ll learn how to find and fix errors in code, use the grid to guide movement, create simple algorithms, and explore real-world applications of computer science through fashion design and data analysis.

Objectives / Topics Covered	● Strengthen understanding of sequences and event-driven programming. ● Use the grid to program precise character movement. ● Develop and adjust algorithms based on visual and spatial reasoning. ● Debug programs by identifying and correcting errors. ● Create interactive experiences using tap events. ● Recognize data patterns and communicate predictions through animation. ● Connect computer science skills to real-world careers.
Lessons	**Debugging: Events and Sequences** (2.AP.1, 2.AP.2, 2.AP.4) ● Find and fix problems in sample code to improve how characters move and respond to events. **Introduction to the Grid** (2.AP.1) ● Use the grid to guide characters to specific locations with accurate movement. **Algorithms: Connecting a Path** (2.AP.1, 2.AP.2, 2.AP.4) ● Build and adjust algorithms to help characters navigate a path, considering size, shape, and start positions. **Careers in CS: Coding for Fashion-Retail** (2.AP.1) ● Explore how coding supports fashion design and animate a digital fashion character. **Tap-a-Mole Game** (2 classes | 2.AP.1, 2.AP.4) ● Create a fun, interactive game using tap events to control characters. **Data Patterns and Predictions** (2.DA.2, 2.DA.3) ● Analyze patterns in data and program characters to share predictions using events and animations. **Sequences & Events Quiz (4 questions)**

Unit 3: Message Events (4 lessons)

In this unit, students will learn how to use message events to make characters communicate and control the flow of a program. They'll model real-world ideas like life cycles and travel, explore how message events can link actions across pages, and reflect on how technology impacts daily life through coding.

Objectives / Topics Covered	● Use message events to coordinate actions between characters. ● Model real-world processes such as cycles and travel using event-driven programming. ● Link actions across multiple pages using messages. ● Create programs that show how technology affects the world around us.
Lessons	**Message Events: Scout Plays in the Forest** (2.AP.1) ● Use message events to control when and how characters take turns interacting in a forest scene. **Programming a Cycle** (2.AP.1) ● Model a repeating cycle (like seasons or life stages) using message events to sequence the stages. **Pages: Scout's Travels** (2.AP.4) ● Help Scout travel between scenes by combining message events and page transitions. **Impacts of Technology in Our World** (2.IC.1) ● Create a program that shows ways technology is used in daily life and its effects on the world. **Message Events Quiz (4 questions)**

Unit 4: Loops (6 lessons)

In this unit, students will explore how loops can make programs shorter and more efficient by repeating actions. They'll identify patterns, debug programs with loops and message events, build creative projects like timers and stories, and practice collaboration by giving and using feedback to improve their work.

Objectives / Topics Covered	• Identify patterns that can be repeated using loops. • Use loops, wait blocks, and turn blocks to control timing and animation. • Debug programs involving both loops and message events. • Create original animations and revise them based on peer feedback. • Practice giving credit and recognizing collaboration in the coding process.
Lessons	**Loops: Follow the Path** (2.AP.1) • Identify repeating patterns in a path and use loops to simplify a character's movement. **Debugging: Message Events and Loops** (2.AP.1, 2.AP.2, 2.AP.4) • Find and fix bugs in code that uses message events and loops to control character actions. **Making a Timer** (2.AP.1, 2.AP.4) • Create two animated timers using loops, wait, and turn blocks, and compare their different speeds. **Create an Original Story Animation** (2 classes | 2.AP.4) • Design and animate a unique story using characters, loops, and events. **Two-Step Dance & Feedback** (2.AP.3, 2.AP.5) • Program a two-step dance, share it with a peer, revise it based on feedback, and give credit to collaborators. **Loops Quiz (4 questions)**

Unit 5: Culmination Projects (10 lessons)

In this culminating unit, students will apply what they've learned throughout the course to create original, interactive projects. They'll combine sequences, events, loops, and message blocks to build games and reflect on how different code blocks work together.

Objectives / Topics Covered	• Use a wide variety of code blocks to design and explain programs. • Combine core concepts—like events, loops, messages, and sequences—into interactive games and animations. • Revise projects based on peer feedback.
Lessons	**Code Block Review** (2 classes | 2.AP.1, 2.AP.4) • Build a program that uses a variety of code blocks and explain how each one functions in the animation. **Racing Game** (2 classes | 2.AP.4) • Create an interactive racing game using events, loops, and message blocks to animate character movement. **Moving Targets Game** (3 classes | 2.AP.1, 2.AP.4) • Program a game with moving targets that uses sequences, events, and pages. **Maze Game Project** (3 classes | 2.AP.4) • Design a maze game that uses multiple coding concepts and revise the game based on peer feedback.

Unit 6: Preparing for Next Year (2 lessons)

This transition unit helps students take their first steps from CodeHopJr to CodeHop. They'll explore the new interface, build confidence using familiar concepts like events and loops, and prepare for the more advanced programming tools they'll use in 3rd grade and beyond.

Objectives / Topics Covered	• Navigate the editor and understand how it compares to CodeHopJr. • Transfer knowledge of events and loops into a new programming environment. • Build simple programs using familiar logic and blocks.

Lessons	**From CodeHopJr to CodeHop Blocks** (2.AP.1) ● Explore the interface and create a simple animation using familiar blocks and tools. **From CodeHopJr to CodeHop Blocks: Part 2** (2.AP.1) ● Create a program that includes an event and a repeating action using a loop. **Preparing for Next Year Quiz (4 questions)**

Digital Literacy (4 lessons)

In this unit, students explore how to use technology safely while learning how computers store, organize, and use information. They practice creating strong passwords, discover how computers manage files, and collect and present data through a simple programming project.

Objectives / Topics Covered	● .Understand how usernames and passwords help protect personal information. ● Explore how computers collect, store, and organize data as files. ● Collect and organize data using a survey.
Lessons	**Password Protectors** (2.IC.2, 2.N1.1) ● Explore how usernames and passwords help keep information safe. **Managing Data Storage and Files** (2.DA.1) ● Explore how computers store, organize, and manage files. **Data Explorers** (2 classes I 2.DA.2, 2.DA.3) ● Collect data and create a program to visually present the results. **Digital Literacy Quiz (4 questions)**

Optional: Artificial Intelligence (2 lessons)

In this optional unit, students explore how artificial intelligence uses data and patterns to solve problems. Through hands-on activities, students discover how AI recognizes information, makes decisions, and helps people complete everyday tasks.

Objectives / Topics Covered	● .Identify examples of artificial intelligence in everyday life. ● Explain that AI uses data and patterns to make decisions. ● Compare tasks that are better suited for humans or AI.
Lessons	**What Can AI Do?** (2.IC.1) ● Explore how AI uses data to help people solve problems and complete tasks. **Machine Learning: AutoDraw** (2.DA.3) ● Explore how AI recognizes patterns to identify drawings and classify objects. **Artificial Intelligence Quiz (4 questions)**

Optional: Unplugged Activities

This optional unplugged unit gives students a hands-on way to explore conditionals without using a device. Working together in small groups, students use coding cards to move Scout through mazes based on “if-then” logic, reinforcing their understanding of conditionals and collaborative problem-solving.

Objectives / Topics Covered	● Practice creating and following conditional instructions in a physical activity. ● Understand how conditionals guide decisions in programming. ● Work collaboratively to solve coding challenges using logic and reasoning.
Lessons	**Coding Card Game: Conditionals** (2.AP.1) ● Work together to move Scout through a maze using cards that include “if-then” logic to respond to obstacles. **Coding Card Game: Conditionals 2** (2.AP.1)

	• Solve a new maze challenge with conditionals by building logical instructions that respond to different scenarios.

Utah Computer Science 2nd 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.
All of these resources and more are found on the **CodeHop Resources Page**.	