Virginia Computer Science
5th Grade Course Syllabus
One Year for Elementary School, 36 Hours

Course Overview and Goals

The Virginia Computer Science 5th Grade Course introduces students to foundational programming concepts through 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 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. Required lessons are labeled with the specific standards they address to support planning and compliance.

Programming Environment: Students will write and run programs that are saved in the CodeHop 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/27184/overview



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

Course Breakdown

Optional Review

In this foundational unit, students get reacquainted with CodeHop and begin to build familiarity with programming blocks, coordinate systems, and creative tools.

Objectives / Topics Covered	● Log in and navigate the CodeHop Playground. ● Understand computer science terms and programming basics. ● Explore the coordinate plane and drawing tools.
Lessons	**Welcome to CodeHop!** ● Practice logging in and exploring the Playground before starting a full lesson. **Introduction to Computer Science** ● Review basic computer science vocabulary and create a simple program. **The Coordinate Plane** ● Use the coordinate plane to design an open-ended animation. **Drawing Tools** ● Create customized sprites and backdrops using the drawing tools.

Unit 1: Getting Started (3 lessons)

In this introductory unit, students begin developing confidence with technology and coding by learning about computing systems, digital data, and computational thinking. Lessons establish a foundation for identifying hardware and software, understanding file storage, and applying computational thinking to real-world scenarios.

Objectives / Topics Covered	● Identify and describe parts of a computing system. ● Distinguish between hardware and software problems. ● Explain how different types of digital data are stored and organized. ● Apply computational thinking practices such as decomposition, sequencing, and abstraction to design solutions.
Lessons	**Practicing with Computing Systems (5.CSY.3.a, 5.CSY.3.b)** ● Identify parts of a computing system and basic troubleshooting steps. **File Management and Data Exploration (5.CSY.1.a, 5.CSY.1.b, 5.NI.1.a, 5.NI.1.b, 5.NI.1.c, 5.NI.1.d)** ● Explain how digital data varies in size and where it can be stored. **Computational Thinking: Design a Neighborhood (5.AP.1.a, 5.AP.1.b, 5.AP.1.c, 5.AP.1.d)** ● Apply computational thinking to plan and design a neighborhood layout.

Unit 2: Sequences, Events & Loops (7 lessons)

In this unit, students expand their programming skills by combining sequences, events, and loops to build more complex projects. They learn to create animations, debug code, refine algorithms, and explore computer science innovators. Lessons emphasize both creativity and critical thinking, while reinforcing how computing can model real-world processes and ideas.

Objectives / Topics Covered	● Create and customize characters and backdrops using drawing tools. ● Use event blocks to trigger actions and control sprite interactions. ● Compare and refine algorithms to determine efficiency. ● Apply loops and pen tools to create patterns and animations. ● Debug programs by identifying and fixing errors. ● Connect coding concepts to real-world innovation through research.

Lessons	**Drawing Tools: Sea Creatures** ● Use image editing tools to create and program deep-sea creatures. **Events: Traveling with Scout** ● Use event blocks to trigger character actions in a travel-themed program. **Compare and Refine Algorithms (5.AP.3.d, 5.AP.3.e)** ● Evaluate and improve multiple algorithms for a task. **Debugging: Make a Pizza** ● Break down and fix a pizza-making program to ensure it runs correctly. **Creating Turtle Graphics** ● Use loops and the pen tool to draw repeated, artistic patterns. **CS Innovators Research Project (2 classes)** ● Research a computer science innovator and create an informational program.

Unit 3: Conditionals & Operators (4 lessons)

In this unit, students learn how to add decision-making to their programs using conditionals and operators. They practice creating branching logic, explore “if/then” and “if/then/else” blocks, and plan larger projects that combine multiple coding skills.

Objectives / Topics Covered	● Use simple conditionals to control sprite behavior. ● Apply complex conditionals with “if/then/else” blocks. ● Plan and decompose steps for creating larger projects.
Lessons	**Conditionals: Mazes (5.AP.2.a, 5.AP.2.c)** ● Create a maze program that uses conditionals to navigate paths. **Complex Conditionals: Chase the Star (5.AP.2.a, 5.AP.2.c, 5.AP.3.b)** ● Build a game using “if/then/else” blocks for branching decisions. **Plan a Quest (2 classes)** ● Break down and plan the steps needed to build a quest-style program.

Unit 4: Variables & Lists (4 lessons)

In this unit, students build an understanding of how programs store, use, and organize data. They explore variables to control actions, operators to simulate randomness, and lists to keep track of multiple items. By applying these concepts, students create interactive projects that rely on data storage and decision-making.

Objectives / Topics Covered	● Use variables to change attributes like pitch and speed. ● Apply operators with variables to simulate random outcomes. ● Create interactive programs that use lists to store and manage data.
Lessons	**Variables in Dance** ● Use variables to change dance speed and pitch, creating a dynamic and musical animation. **Operators: Coin Flip (5.AP.3.b)** ● Create a coin flipping program using variables and operators. **Lists: Shopping with Scout (2 classes)** ● Build a shopping simulator that uses lists to track items and variables and operators to calculate prices.

Unit 5: Culmination Projects (8 lessons)

Students apply the skills they have developed throughout the course to larger, open-ended projects. These lessons emphasize creativity, game design, applied computer science concepts, and real-world problem solving through coding.

Objectives / Topics Covered	• Apply coding concepts such as variables, loops, conditionals, and clones in creative projects. • Explore game mechanics and randomness through interactive design. • Use the design thinking process to solve real-world problems with technology. • Conduct inquiry-based projects to represent and interpret data visually.
Lessons	**Clones: Throwing Acorns Game** • Create a game where acorns are cloned and thrown toward targets. **Classes and Objects in Games (2 classes)** • Learn about classes and objects while programming a game that randomizes object characteristics. **Design an App (3 classes | 5.AP.1.d, 5.AP.2.b, 5.AP.3.a, 5.AP.3.c, 5.AP.3.d, 5.AP.3.e, 5.IC.5.b, 5.IC.5.d)** • Use design thinking to plan an app that solves a real-world problem. **Inquiry Project: Line Graph (2 classes | 5.DA.2.a, 5.DA.2.b, 5.DA.2.c, 5.DA.2.d)** • Investigate a question and modify a program to present the results as a line graph.

Unit 8: Digital Literacy (10 lessons)

Students build digital literacy skills by learning how to research, evaluate, and safely use technology. Lessons cover topics such as online research, data use and storage, cybersecurity, networks, and the cultural impacts of computing. Students also explore artificial intelligence and its role in decision-making and problem-solving.

Objectives / Topics Covered	• Search for information online and provide proper attribution. • Understand and manage digital identity and online behavior. • Recognize and respond to cyberbullying and online risks. • Learn about cybersecurity policies, laws, and real-world applications. • Explain how networks and protocols enable communication. • Analyze the cultural impacts of technology and predict future trends. • Explore how machines learn and classify data.
Lessons	**Use and Search the Right Way (5.IC.4.a, 5.IC.4.b)** • Search online effectively and provide proper attribution to sources. **Data Analysts (5.DA.1.a, 5.DA.1.b, 5.DA.1.c, 5.DA.1.d, 5.DA.2.c, 5.DA.2.d, 5.IC.2.a)** • Evaluate and analyze data for reliability to draw conclusions and make predictions. **Digital Identity (5.IC.4.c, 5.IC.5.c)** • Connect real-world and online identities and identify positive digital footprint actions. **Standing Up to Cyberbullying (5.IC.1.a, 5.IC.1.b, 5.IC.1.c)** • Recognize types of online hurtful behavior and describe ways to respond. **Cybersecurity Policies and Laws (5.CYB.2.a, 5.CYB.2.b, 5.CYB.2.c)** • Learn about cybersecurity policies, then research and explain a law specific to your state. **Scout's Cybersecurity Mission (2 classes | 5.CYB.1.a, 5.CYB.1.b, 5.CYB.1.c, 5.CYB.1.d, 5.CYB.1.e)** • Create a Scratch program that demonstrates solving a real-world cybersecurity issue. **Networks and Protocols (5.NI.1.a, 5.NI.1.b, 5.NI.1.c, 5.NI.1.d)** • Learn how data is transferred through networks and compare WiFi, wired, and cellular connections. **Impacts of Computing: Innovation (5.IC.2.a, 5.IC.2.b, 5.IC.2.c, 5.IC.3.a, 5.IC.3.b, 5.IC.5.a)** • Explore how computing innovations have changed the way we live and work. **How Machines Learn (5.CSY.2.a, 5.CSY.2.b, 5.DA.3.a, 5.DA.3.b, 5.DA.3.c)** • Explain machine learning approaches and build a classification system using a tree structure.

Virginia Computer Science 5th 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 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 **CodeHop Resources Page**.	