New York Computer Science and Digital Fluency: 4th Grade Course Syllabus

One Year for Elementary School, 36 Hours

Course Overview and Goals

The **New York Computer Science and Digital Fluency: 4th Grade** 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.

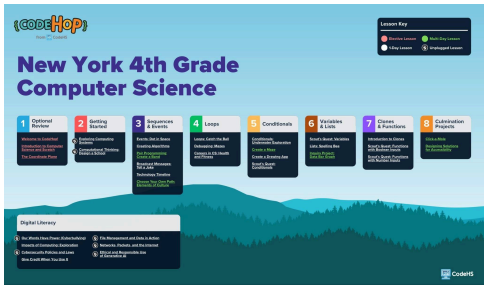
Learning Environment: This course is teacher-led and includes ready-to-use lessons following a consistent structure: **Introduction, Guided Practice, Independent Practice, Extension, and Reflection**. Instruction follows an "I do, we do, you do" model and incorporates spiral review to reinforce concepts 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. 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 students' accounts. 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/NY_4/overview



A clickable PDF can be found at https://codehs.com/NY-K-5Roadmaps

Course Breakdown

Optional Review

Students review foundational CodeHop skills from 3rd grade — logging in, core CS vocabulary, and coordinate plane animations — to refresh their programming foundation before new content.

Objectives / Topics Covered	● Log in and navigate the CodeHop Playground ● Define key computer science vocabulary ● Create a simple program in CodeHop ● Use the coordinate plane to position and animate sprites
Lessons	**Welcome to CodeHop!** ● Log in and explore the CodeHop Playground interface. **Introduction to Computer Science (4-6.DL.4)** ● Define important computer science vocabulary and create a simple program. **The Coordinate Plane (4-6.DL.4)** ● Create an open-ended animation using the coordinate plane to position sprites.

Unit 1: Getting Started (2 lessons)

Students identify parts of a computing system, troubleshoot hardware and software problems, and apply computational thinking to a school design challenge.

Objectives / Topics Covered	● Identify hardware and software components of a computing system ● Troubleshoot simple hardware and software problems ● Apply computational thinking to plan a multi-step design ● Decompose a complex problem into smaller steps
Lessons	**Exploring Computing Systems (4-6.CT.1, 4-6.DL.5, 4-6.NSD.2, 4-6.NSD.3)** ● Identify parts of a computing system and diagnose simple hardware and software problems. **Computational Thinking: Design a School (4-6.CT.10, 4-6.CT.4)** ● Use computational thinking strategies to design a school layout through sequencing and abstraction.

Unit 2: Sequences and Events (8 lessons)

Students create and compare algorithms, use broadcast messages and keyboard inputs, build interactive timelines, and explore how CS connects to sports, careers, and cultural themes.

Objectives / Topics Covered	● Program multiple algorithms and evaluate which best meets a goal ● Use broadcast messages to coordinate sprite interactions ● Collaborate through pair programming to build a shared program ● Create interactive programs that connect CS to real-world topics
Lessons	**Creating Algorithms (4-6.CT.6, 4-6.DL.4)** ● Program multiple algorithms for the same task and assess which approach is most effective. **Pair Programming: Create a Band (2 classes | 4-6.CT.10, 4-6.CT.4, 4-6.CT.8, 4-6.DL.4)** ● Collaborate through pair programming to design and code a band using keyboard inputs. **Broadcast Messages: Tell a Joke (4-6.CT.4, 4-6.DL.4)** ● Use broadcast messages to program two sprites to tell a knock-knock joke.

	Technology Timeline (4-6.IC.1) ● Create an interactive timeline of music player technology and explain its cultural influence. **Careers in CS: Major League Baseball (4-6.IC.3, 4-6.IC.5, 4-6.IC.7)** ● Explain how coding is used in sports and build a timeline program from key events in an article. **Choose Your Own Path: Elements of Culture (2 classes | 4-6.CT.4, 4-6.DL.4, 4-6.IC.2)** ● Identify elements of culture and create a choose-your-own-path game built around a cultural theme.

Unit 3: Loops (2 lessons)

Students use two types of loops to build a game, then practice decomposing and debugging programs to fix errors and make them run as intended.

Objectives / Topics Covered	● Use repeat and forever loops to build a game ● Distinguish between different loop types and their use cases ● Decompose a program to locate and understand bugs ● Debug a program to make it run as intended
Lessons	**Loops: Catch the Ball (4-6.DL.4)** ● Use two types of loops to build a simple ball-catching game. **Debugging: Mazes** ● Decompose a maze program to identify and fix bugs so it runs correctly.

Unit 4: Conditionals (3 lessons)

Students use conditionals to control program behavior, build a drawing app with keyboard and mouse inputs, and complete a Scout's Quest skill-review project using if/then logic.

Objectives / Topics Covered	● Create a program that uses conditional (if/then) logic ● Combine loops, keyboard inputs, and conditionals in one program ● Use mouse inputs to trigger conditional actions ● Apply conditionals in a culminating skill-review project
Lessons	**Conditionals: Underwater Exploration (4-6.CT.8, 4-6.DL.4)** ● Create an underwater-themed program that uses conditional statements to control sprite behavior. **Create a Drawing App (4-6.CT.8, 4-6.DL.4)** ● Build a drawing app by programming keyboard and mouse inputs, loops, and conditional statements together. **Scout's Quest: Conditionals (4-6.CT.8, 4-6.DL.4)** ● Create a program using if/then conditionals as part of the Scout's Quest skill review series.

Unit 5: Variables and Lists (4 lessons)

Students use variables to track score, work with lists to store multiple values, and complete an inquiry project that collects and visualizes data in a bar graph.

Objectives / Topics Covered	● Create and update variables to track data in a program ● Use lists to store and retrieve multiple values ● Build a game that uses variables and conditionals together

	● Follow the inquiry process to collect and display data
Lessons	**Scout's Quest: Variables (4-6.CT.4, 4-6.CT.7, 4-6.CT.8)** ● Create and use variables to track points in a scoring program as part of the Scout's Quest series. **Lists: Spelling Bee** ● Use lists to store and access words in a functional spelling bee game. **Inquiry Project: Data Bar Graph (2 classes | 4-6.CT.1, 4-6.CT.2, 4-6.CT.3)** ● Follow the inquiry process, collect data, and modify a program to display results as a bar graph.

Unit 6: Clones and Functions (3 lessons)

Students create sprite clones to build animations, then write functions with boolean and number inputs to organize code and control program behavior in Scout's Quest projects.

Objectives / Topics Covered	● Use clones to create multiple copies of a sprite in a program ● Create functions with boolean inputs to branch program behavior ● Create functions with number inputs to customize outputs ● Organize complex programs using reusable functions
Lessons	**Introduction to Clones (4-6.CT.7, 4-6.CT.8, 4-6.DL.4)** ● Create an animation using clones and investigate the limits of how many clones a program can handle. **Scout's Quest: Functions with Boolean Inputs (4-6.CT.4, 4-6.CT.5, 4-6.CT.8, 4-6.DL.4)** ● Write a function with a boolean input that performs different actions depending on whether a password is correct. **Scout's Quest: Functions with Number Inputs (4-6.CT.4, 4-6.CT.5, 4-6.DL.4)** ● Create a drawing program using functions that accept number inputs to customize output.

Unit 7: Culmination Projects (4 lessons)

Students synthesize their skills by building a complete Whack-a-Mole style game and applying the design thinking process to redesign a game for improved accessibility.

Objectives / Topics Covered	● Combine conditionals, variables, booleans, and events in one game ● Apply the design thinking process to a real-world problem ● Redesign a program to improve accessibility for diverse users ● Test, iterate, and present a finished project
Lessons	**Click-a-Mole (2 classes | 4-6.CT.4, 4-6.CT.8, 4-6.CT.9)** ● Build an interactive Whack-a-Mole style game using conditionals, variables, booleans, and events. **Designing Solutions for Accessibility (2 classes | 4-6.CT.10, 4-6.CT.4, 4-6.CT.8, 4-6.CT.9, 4-6.DL.4, 4-6.IC.6, 4-6.NSD.1)** ● Use the design thinking process to redesign a game so it is more accessible and usable for diverse players.

Unit 8: Digital Literacy (10 lessons)

Students explore computing's impact on culture, digital citizenship, cybersecurity, cryptography, research attribution, file management, networking, and the ethical use of AI.

Objectives / Topics Covered	• Explain how technology and culture influence each other • Describe digital footprints and strategies to manage online identity • Model how networks transfer information using packets • Explain the ethical and responsible use of generative AI
Lessons	**Impacts of Computing: Exploration (4-6.IC.1, 4-6.IC.3)** • Explain how technology and culture influence each other and create a project comparing a past and present technology. **Exploring Digital Etiquette and Communication (4-6.DL.2)** • Demonstrate proper digital etiquette when communicating in an online community. **Standing Up to Cyberbullying (4-6.DL.6, 4-6.DL.7)** • Recognize types of online hurtful behavior and describe ways to respond responsibly. **Managing Digital Footprints (4-6.CY.3, 4-6.DL.6, 4-6.DL.7)** • Explain how online actions create permanent digital footprints and describe how to manage a digital identity responsibly. **Exploring Cryptography (4-6.CY.4)** • Encrypt and decrypt messages using Atbash, Caesar, and student-created ciphers to explain the purpose of cryptography. **Cybersecurity Policies and Laws (4-6.CY.1, 4-6.CY.2, 4-6.CY.3, 4-6.CY.5, 4-6.IC.2, 4-6.IC.4)** • Explain how cybersecurity policies relate to school rules and research a cybersecurity law specific to their state. **Give Credit When You Use It (4-6.DL.3, 4-6.IC.2)** • Search for information online to answer a question and provide proper attribution to sources. **File Management and Data in Action (4-6.CT.7, 4-6.IC.4, 4-6.NSD.5)** • Explain how different types of digital data take up varying amounts of storage space and can be stored in different ways. **Networks, Packets, and the Internet (4-6.NSD.4)** • Compare wired and wireless network connections and model how devices communicate using rules and packets. **Ethical and Responsible Use of Generative AI (4-6.CY.1, 4-6.CY.2, 4-6.DL.7, 4-6.IC.1, 4-6.IC.3, 4-6.IC.5)** • Describe the pros and cons of generative AI and co-create a class Code of Conduct for AI use.

New York Computer Science and Digital Fluency: 4th Grade Supplemental Materials

Resources	**Description**
Parent Welcome Letter (Spanish)	Send this letter home to introduce families to their new computer science curriculum.
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.
These resources and more are found on the **CodeHop Resources Page**.	