New York Computer Science & Digital Fluency 3rd Grade Course Syllabus

One Year for Elementary School, 36 Hours

Course Overview and Goals

The **New York Computer Science & Digital Fluency 3rd Grade Course** 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.

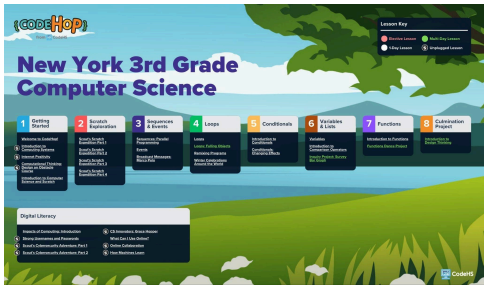
Learning Environment: This course is designed to be teacher-led, with ready-to-use lesson plans that follow a structured format: **Introduction, Guided Practice, Independent Practice, Extension, and Reflection**. Lessons are built with spiral review to reinforce key concepts and culminate in engaging 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. Digital literacy lessons are also available to complement the programming curriculum with non-programming computer and technology skills.

Programming Environment: Students will write and run programs in **Scratch** embedded and 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_3/overview



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

Course Breakdown

Unit 1: Getting Started (4 weeks)

In this introductory unit, students are introduced to the basics of computing by identifying key parts of a computing system and learning how to troubleshoot simple issues. They also begin developing foundational computational thinking skills by designing and sequencing steps in an obstacle course challenge.

Objectives / Topics Covered	● Log in and navigate the Playground. ● Identify parts of a computing system. ● Practice basic troubleshooting strategies for common computer problems. ● Apply computational thinking skills.
Lessons	**Welcome to CodeHop!** ● Learn how to log in and navigate the Playground to get comfortable using the platform. **Introduction to Computing Systems** ● Identify the main parts of a computing system—hardware, software, input, and output—and solve basic problems when something doesn't work. **Internet Positivity** ● Create a code of conduct for responsible online behavior. **Computational Thinking: Design an Obstacle Course** ● Use computational thinking skills like sequencing and problem-solving to plan and design an obstacle course. **Introduction to Computer Science and Scratch** ● Define key computer science vocabulary and create a simple Scratch program to apply foundational programming concepts.

Unit 2: Scratch Exploration (4 weeks)

Students explore the basics of Scratch by using commands to move and animate sprites, building familiarity with sequencing, motion, and dialogue. As they progress through a story-driven unit, they apply programming concepts such as events and loops to create interactive and animated stories.

Objectives / Topics Covered	● Use basic commands to program sprites. ● Explore the Scratch interface.
Lessons	**Scout's Scratch Expedition Part 1** ● Use basic Scratch commands to program a sprite to move and talk while beginning a story-driven coding adventure. **Scout's Scratch Expedition Part 2** ● Add new sprites and create a sequence of actions to animate a simple story in Scratch. **Scout's Scratch Expedition Part 3** ● Create an animated story using loops, events, looks, and motion blocks to bring characters to life. **Scout's Scratch Expedition Part 4** ● Continue building an animated story by using events, looks, and motion blocks to enhance interactivity and animation.

Unit 3: Sequences and Events (3 weeks)

Students build foundational programming skills by creating sequences and using events to control when actions happen. They also explore parallel programming and broadcast messages to coordinate interactions between sprites.

Objectives	● Create programs using sequences.

/ Topics Covered	• Use event blocks to trigger actions and coordinate program flow. • Apply broadcast messages to enable communication between multiple sprites.
Lessons	**Sequences: Parallel Programming** • Create a program with multiple sequences running at the same time to control different sprite actions. **Events** • Use event blocks to trigger actions in a Scratch program and control when things happen. **Broadcast Messages: Marco Polo** • Use broadcast messages to make sprites communicate and respond to each other's actions.

Unit 4: Loops (5 weeks)

Students learn how loops repeat instructions and use them to create more efficient programs in Scratch. They will compare different types of loops to understand their effects and uses in animations.

Objectives / Topics Covered	• Use loops to simplify code. • Develop efficient, repeatable patterns in animations and interactive projects.
Lessons	**Loops** • Learn that loops repeat one or more instructions and use them in Scratch to simplify and improve programs. **Loops: Falling Objects (2 part lesson)** • Create a program using different types of loops and compare how each affects how the program runs. **Remixing Programs** • Remix digital programs and give appropriate attribution. **Winter Celebrations Around the World** • Create a program to tell how a specific holiday is celebrated.

Unit 5: Conditionals (2 weeks)

Students explore conditional logic in Scratch to make programs respond dynamically to different conditions and inputs.

Objectives / Topics Covered	• Use if/then and if/else blocks in Scratch programs. • Use conditionals to build interactive and reactive programs.
Lessons	**Introduction to Conditionals** • Learn how to use if/then blocks to make decisions in a program based on specific conditions. **Conditionals: Changing Effects** • Create a program that uses conditionals to control the state of sprites.

Unit 6: Variables (4 weeks)

In this unit, students explore variables and comparison operators to manage and organize data in programs.

Objectives / Topics Covered	• Learn how variables store information. • Use comparison operators to create more complex decision-making logic.
Lessons	**Variables** • Understand what variables are and how to create and update them to store changing information in a program.

	Introduction to Comparison Operators ● Use comparison operators with numbers and variables to create more complex if/else conditions. **Inquiry Project: Survey Bar Graph (2 part lesson)** ● Follow the inquiry process and modify a program to display the results of an investigation.

Unit 7: Functions (4 weeks)

Students will explore functions in this unit, allowing them to refine and organize more complex programs.

Objectives / Topics Covered	● Use functions to organize code into reusable parts.
Lessons	**Introduction to Functions** ● Create and use functions in a Scratch program to organize code and make actions easier to repeat and manage. **Functions Dance Project (3 part project)** ● Use functions to call dance moves in a sequence that aligns with music.

Unit 8: Culmination Project (2 weeks)

Students apply their programming knowledge to make a tool more accessible to users, with events, conditionals, variables, comparison operators, and broadcast messages to bring their project to life. This final project reinforces key concepts and allows for creativity and problem-solving in a self-directed build.

Objectives / Topics Covered	● Design, create, and personalize a project that demonstrates mastery of core coding skills.
Lessons	**Introduction to Design Thinking (2 part lesson)** ● Explore ways to make digital tools more accessible using the design process.

Unit 7: Digital Literacy (8 weeks)

Students explore strategies for staying safe online, including creating strong usernames and passwords and recognizing common cybersecurity threats. Students also learn how technology has impacted people and proper etiquette for researching and collaborating online.

Objectives / Topics Covered	● Understand how technology impacts society. ● Learn strategies for staying safe online. ● Recognize common digital threats. ● Follow laws when researching online.
Lessons	**Impacts of Computing: Introduction** ● Explore how computing affects people and society. **Strong Usernames and Passwords** ● Learn how to create strong usernames and passwords and understand why they are important for keeping information safe. **Scout's Cybersecurity Adventure: Part 1** ● Understand basic cybersecurity concepts, identify common online threats, and learn tips for staying safe online. **Scout's Cybersecurity Adventure: Part 2** ● Practice secure habits and understand tools and techniques to protect information online. **CS Innovators: Grace Hopper** ● Learn about Grace Hopper's impact on computer science and use binary code to decipher mystery words.

	What Can I Use Online? ● Research information to answer questions online and give credit to original sources. **Online Collaboration** ● Understand that collaborating with others can provide diverse perspectives. **How Machines Learn** ● Explain different machine learning approaches and create a classification system.

New York Computer Science & Digital Fluency 3rd Grade Course 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.
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 **CodeHop Resources Page**.	