CodeHS

Florida Computer Science Foundations 3rd Grade Course Syllabus

One Year for Elementary School, 36 Hours

Course Overview and Goals

The **Florida Computer Science Foundations 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. Optional 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/FL_3/overview

Course Breakdown

Unit 1: Getting Started (2 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 CodeHS!** ● 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. **Computational Thinking: Design an Obstacle Course** ● Use computational thinking skills like sequencing and problem-solving to plan and design an obstacle course.

Unit 2: Scratch Exploration (5 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. ● Use drawing tools to create unique sprites and backdrops.
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. **Scratch Drawing Tools** ● Create customized sprites and backdrops using drawing tools.

Unit 3: Sequences and Events (7 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 / Topics Covered	● Create programs using sequences. ● 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. **Costumes, Backdrops, and Animations** ● Create a program that includes animated sprites and interactive backdrops. **Remixing Programs** ● Create or remix digital projects using appropriate content while giving credit to original creators. **Broadcast Messages: Marco Polo** ● Use broadcast messages to make sprites communicate and respond to each other's actions. **Nonfiction Animated Recordings** ● Use events to create an animated reading of a nonfiction text.

	Measuring Lengths ● Create an interactive measurement game using events.

Unit 4: Loops (8 weeks)

Students learn how loops repeat instructions and use them to create more efficient programs in Scratch. They also build debugging skills by analyzing and fixing errors in programs, and compare different types of loops to understand their effects and uses in animations.

Objectives / Topics Covered	● Use loops to simplify code. ● Identify and fix issues related to loops and events. ● 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. **Debugging: Events and Loops** ● Break down a program to find and fix problems involving events and loops. **Animating Poetry** ● Create an animated reading of a poem using events and loops. **Adding with Loops** ● Use loops to repeat commands and add multi-digit numbers based on place value. **Modeling Network Connections** ● Understand how digital devices connect and communicate over networks. **Loops: Falling Objects (2 part lesson)** ● Create a program using different types of loops and compare how each affects how the program runs. **Animating Sprites with Multiplication** ● Use multiplication to animate sprites with loops and wait blocks.

Unit 5: Conditionals and Variables (5 weeks)

Students explore key programming concepts by learning how to use conditionals, variables, and comparison operators to control the behavior of their programs.

Objectives / Topics Covered	● Use if/then and if/else blocks in Scratch programs. ● Learn how variables store information. ● Use comparison operators to create more complex decision-making logic.
Lessons	**Introduction to Conditionals** ● Learn how to use if/then blocks to make decisions in a program based on specific conditions. **Fossils and Past Environments** ● Use conditionals and loops to model interpreting data from fossils that show evidence of past environments. **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. **Fractions and Variables** ● Represent fractions on a number line using variables and conditionals.

Unit 6: Culmination Project (3 weeks)

Students apply their programming knowledge using 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	**Digital Pet Project (3 part lesson)** ● Create an interactive digital pet using events, conditionals, variables, comparison operators, and broadcast messages to demonstrate mastery of key programming concepts.

Unit 7: Digital Literacy (7 weeks)
Students explore how their actions shape their digital identity and learn strategies for staying safe online, including creating strong usernames and passwords and recognizing common cybersecurity threats.

Objectives / Topics Covered	● Understand how to build a positive digital footprint. ● Learn strategies for staying safe online. ● Recognize common digital threats.
Lessons	**Digital Identity** ● Explore how real-world and online actions shape digital identity and learn ways to build a positive digital footprint. **Scout's Cybersecurity Adventure: Part 1** ● Understand basic cybersecurity concepts, identify common online threats, and learn tips for staying safe online. **How Machines Learn** ● Explain different machine learning approaches. **Introduction to Productivity Software** ● Compare and select software applications to complete different tasks. **What Can I Use Online?** ● Research information to answer questions online and give credit to original sources. **Inquiry Project: Survey Bar Graph (2 part lesson)** ● Follow the inquiry process to collect data and modify a program to display the results using a bar graph.

Interdisciplinary Connections (Supplemental)
In this unit, students strengthen their programming skills by applying them to interdisciplinary concepts in math, science, social studies, and ELA. These flexible, supplemental lessons can be integrated throughout the year to enrich core instruction and provide meaningful, real-world connections across subjects.

Objectives / Topics Covered	● Use conditionals and loops to build simulations and practice content-area skills. ● Create animations to connect programming to the real world.
Lessons	**Classifying Shapes by Category** ● Use events to classify quadrilaterals based on their properties. **Multiplication and Conditionals** ● Use conditionals to review multiplication. **Animating Unit Fractions** ● Use loops to animate repeated addition of unit fractions on a number line. **Weather and Climate** ● Use events and climate data to predict typical weather conditions.

	Exploring Adaptations ● Create an interactive program to explain how adaptations help animals survive. **Modeling Life Cycles** ● Use broadcast messages to model the stages of a frog's life cycle. **Nutrition Maze** ● Use conditionals to create an interactive nutrition maze game. **Classifying Rocks** ● Use conditionals to classify rocks based on the Mohs Hardness Scale. **Balanced and Unbalanced Forces** ● Demonstrate how balanced and unbalanced forces impact an object's speed using conditionals and variables. **Parts of Speech: Random Sentence Generator** ● Generate random numbers to create simple sentences. **Punctuation Game** ● Create a game using conditionals to add punctuation to dialogue and addresses. **Communities Adapt To & Modify Their Environment** ● Use click events to create a scene that shows how communities adapt to or modify their environments.

Florida Computer Science 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 **Elementary Resources Page**.	