Nebraska Technology 3rd Grade Course Syllabus

One Year for Elementary School, 36 Hours

Course Overview and Goals

The **Nebraska Technology 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 and productivity software 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/NE_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 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. **Computational Thinking: Design an Obstacle Course** ● Use computational thinking skills like sequencing and problem-solving to plan and design an obstacle course.

Unit 2: 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, culminating in the creation of an animated story that demonstrates their understanding of sequencing and events.

Objectives / Topics Covered	● Create programs using sequences. ● Use event blocks to trigger actions and coordinate program flow. ● Program simple animations and interactions.
Lessons	**Costumes, Backdrops, and Animations** ● Animate sprites and backdrops in a program. **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. **Careers in CS: Health and Fitness** ● Explain how fitness and coding can amplify human strengths and create an animation to demonstrate how technology is used in health and fitness. **Remixing Programs** ● Create or remix digital projects using appropriate content while giving credit to original creators. **Creative Storytelling (2 part lesson)** ● Plan and animate a story by combining sequences and events to build interactive scenes.

Unit 3: Loops (4 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. **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.

Unit 4: Conditionals and Variables (3 weeks)

Students explore key programming concepts by learning how to use conditionals, variables, and comparison operators to control the behavior of their programs. They apply these skills to design and debug a platform-style game, using loops and keyboard events to create interactive gameplay that responds to changing conditions.

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. ● Use functions to organize code into reusable parts.
Lessons	**Introduction to Conditionals** ● Learn how to use if/then blocks to make decisions in a program based on specific conditions. **Introduction to Functions** ● Create and use functions in a Scratch program to organize code and make actions easier to repeat and manage. **Introduction to Comparison Operators** ● Use comparison operators with numbers and variables to create more complex if/else conditions.

Unit 5: Culmination Project (2 weeks)

Students apply their programming knowledge to modify a game for user accessibility, 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	**Introduction to Design Thinking (2 part lesson)** ● Explore ways to make digital tools more accessible using the design process.

Unit 6: Digital Literacy (11 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	**Our Words Have Power (Cyberbullying)** ● Understand cyberbullying and how it affects others. **Digital Identity** ● Explore how real-world and online actions shape digital identity and learn ways to build a positive digital footprint. **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. **Collaborating Globally (2 part lesson)** ● Practice global collaboration skills while creating a project with others online. **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. **File Management and Data** ● Organize, save, and manage digital files effectively.

Unit 7: Productivity Software (7 weeks)

In this unit, students will be introduced to the different types of productivity software and will practice using word processing, presentation, and spreadsheet skills.

Objectives / Topics Covered	● Understand the purposes of different types of productivity software. ● Practice using word processing software. ● Practice using presentation software. ● Practice using spreadsheet software.
Lessons	**Introduction to Productivity Software** ● Select appropriate software applications for different tasks. **Introduction to Google Docs™ | Microsoft Word™** ● Make edits in a word processing application. **Introduction to Google Slides™ | Microsoft Powerpoint™** ● Create an “All About Me” presentation. **Introduction to Google Sheets™ | Microsoft Excel™** ● Enter, organize, and visualize data in a spreadsheet. **Productivity Software Inquiry Project (3 part lesson)** ● Use productivity software to organize research and present information to others.

Nebraska Technology 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**.	