Nebraska Technology 5th Grade Course Syllabus

One Year for Elementary School, 36 Hours

Course Overview and Goals

The **Nebraska Technology 5th 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_5/overview

Course Breakdown

Optional Review

In this unit, students review key computer science concepts by logging into the Playground, creating simple Scratch programs, using the coordinate plane for animation, and customizing projects with Scratch's drawing tools.

Objectives / Topics Covered	● Log in and navigate the Playground. ● Create simple Scratch programs. ● Use the coordinate plane to position sprites.
Lessons	**Welcome to CodeHop!** ● Learn how to log in and navigate the Playground. **Introduction to Computer Science and Scratch**

	- Define basic computer science vocabulary and create a simple Scratch program to practice foundational coding skills. **The Coordinate Plane** - Use the coordinate plane in Scratch to position sprites and create an open-ended animation. **Drawing Tools: Sea Creatures** - Use image editing tools to create and program deep sea creatures.

Unit 1: Getting Started (3 weeks)

In this unit, students learn to use technology responsibly by creating a personal screen time plan and applying computational thinking to design a neighborhood. They will also begin exploring computing systems in an upcoming lesson.

Objectives / Topics Covered	- Understand the importance of balancing screen time and using technology responsibly. - Apply computational thinking skills. - Explore how to interact with and operate basic computing systems.
Lessons	**Safe and Responsible Use of Technology** - Learn the importance of balancing screen time and using technology responsibly by creating a personal screen time plan. **Practicing with Computing Systems** - Interact with and troubleshoot basic computing systems. **Computational Thinking: Design a Neighborhood** - Use computational thinking skills—like breaking down tasks and identifying patterns—to design a neighborhood layout.

Unit 2: Sequences and Events (2 weeks)

In this unit, students build foundational programming skills by using events, comparing and refining algorithms for efficiency, and developing debugging strategies to identify and fix problems in a program.

Objectives / Topics Covered	- Use events to control when actions occur in a program. - Compare and refine multiple algorithms to choose the most effective solution for a task. - Decompose programs to identify and fix errors through debugging.
Lessons	**Compare and Refine Algorithms** - Explore multiple algorithms for completing the same task and refine them to choose the most efficient solution. **Debugging: Make a Pizza** - Break down a Scratch program to identify and fix errors, ensuring it runs correctly and as intended.

Unit 3: Loops (3 weeks)

In this unit, students explore how loops can be used to create visual effects by programming turtle graphics and building multi-scene animations using repeat blocks.

Objectives / Topics Covered	- Use repeat loops to simplify and organize code.
Lessons	**Creating Turtle Graphics** - Use the pen tool and loops in Scratch to create turtle-style drawings with repeated patterns.

	Animation Loops Project (2 part lesson) - Use repeat loop blocks to build a multi-scene animation, demonstrating how loops can simplify and organize repeated actions.

Unit 4: Conditionals and Variables (6 weeks)

In this unit, students deepen their understanding of conditionals, variables, and lists by programming interactive activities such as mazes, quests, and games. They apply comparison operators, control elements like speed and pitch, and use lists to manage and simulate data in engaging, real-world scenarios.

Objectives / Topics Covered	- Use conditionals to control decision-making in programs. - Create programs using variables, lists, and operators to manage and track information.
Lessons	**Conditionals: Mazes** - Create a Scratch program that uses conditionals to guide characters through a maze. **Plan a Quest (2 part lesson)** - Break down the steps needed to design a quest-based program, using logical sequencing and planning. **Variables in Dance** - Use variables to control features like pitch and speed in a dance animation program. **Game Mechanics with Comparison Operators** - Apply comparison operators and variables to program game-ending conditions and logic. **Giving Credit Through Attributions** - Learn to give proper credit when using or remixing programs and images to support respectful digital content use.

Unit 5: Clones and Functions (6 weeks)

In this unit, students explore advanced programming concepts by using clones to build interactive games, learning about classes and objects, and applying randomization to create variety in gameplay. They also create and use functions with inputs to organize code and personalize projects, reinforcing the value of reusable code structures.

Objectives / Topics Covered	- Use clones to create repeated sprites. - Learn about classes and objects by programming games that modify object properties. - Create and use functions to organize and reuse code efficiently.
Lessons	**Clones: Throwing Acorns Game** - Create an interactive game using clones to throw acorns and manage repeated sprite actions. **Classes and Objects in Games (2 part lesson)** - Learn about classes and objects by creating a game that uses randomizers to change object characteristics. **Functions: About Me** - Create and use a function with input to personalize a program that shares information about the student. **Functions: Paint by Numbers (2 part lesson)** - Design a function that takes input to draw and fill shapes, reinforcing how inputs can control repeated actions in a program.

Unit 6: Culmination Project (3 weeks)

In this unit, students apply the design thinking process to create an app that addresses a specific user need, integrating their programming knowledge with problem-solving, creativity, and user-centered design.

Objectives / Topics Covered	- Combine creativity, problem-solving, and programming skills to develop a user-centered digital project.
Lessons	**Design an App (3 part lesson)** - Use the design thinking process to plan and design an app that addresses a specific user need, combining creativity, problem-solving, and coding skills.

Unit 7: Digital Literacy (9 weeks)

In this unit, students build digital literacy by practicing responsible online collaboration and learning how to give proper attribution when using digital content. They also investigate data through an inquiry project, study how network protocols support data transfer, and begin to understand the broader social and technical impacts of computing.

Objectives / Topics Covered	- Follow the inquiry process and represent data visually. - Explore how online and offline collaboration enhances communication. - Learn how network protocols support data transfer and compare different types of network connections.
Lessons	**Standing Up to Cyberbullying** - Discuss responsible social media use and learn how to prevent and respond to cyberbullying. **Use and Search the Right Way** - Search for information to answer questions online and provide proper attribution. **Inquiry Project: Line Graph (2 part lesson)** - Collect and analyze data through an inquiry project, then modify a Scratch program to display the results using a line graph. **Collaborating Globally (2 part lesson)** - Practice global collaboration skills while creating a project with others online. **AI Products: Ethical and Responsible Choices** - Analyze the benefits and challenges of various AI products from multiple perspectives. **File Management and Data Exploration** - Explore how to save, organize, and manage digital files effectively. **Networks and Protocols** - Learn how network protocols work, how data is transferred, and compare the functions of WiFi, wired, and cellular networks.

Unit 8: Productivity Software (4 weeks)

In this unit, students will be introduced to the different types of productivity software and will complete a project using different types of software.

Objectives / Topics Covered	- Understand the purposes of different types of productivity software. - Use productivity software to organize, communicate, and present information.
Lessons	**Introduction to Productivity Software** - Select appropriate software applications for different tasks. **Productivity Software Inquiry Project (3 part lesson)** - Use productivity software to organize research and present information to others.

Nebraska Technology 5th 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**.	