CodeHS

Computing Ideas
Common Core State Standards for Math
Standards Alignment Overview

The Computing Ideas course is a first computer science course introducing the basics of programming with Karel the Dog, the basics of designing a web page, how information and images are represented with computers, and the functionality and impact of the Internet. This document is an overview of how the Computing Ideas course aligns with the Common Core State Standards for Mathematics.

Standards for Mathematical Practice

1. **Make sense of problems and persevere in solving them.**
   In the CodeHS Computing Ideas course, students design and test basic web pages and sites that they build, and these pages and sites will invariably have bugs, performance issues, browser rendering differences, deprecated code, and more. Students are encouraged during testing to start by explaining to themselves the context of a problem on their site and looking for entry points to its solution. They plan a solution pathway rather than jumping into a solution attempt with the issues on their sites. They persevere in solving them by using different browsers, checking their HTML and CSS code for errors, and looking up specific issues on the Internet or consulting others, as needed until their sites are usable and presentable.

   Computing Systems is a core concept throughout the Computing Ideas course. Students learn about various computing devices and how humans interact with them, including devices that extend the capabilities of humans. Students learn about computer organization including the relationship between hardware and software. Troubleshooting computing systems is a core concept of the Computing Ideas course as well. Computing systems might not work as expected because of problems in the software. Students are expected to identify problems in their programs and fix them.

2. **Reason abstractly and quantitatively.**
   The Computing Ideas course encourages students to make sense of quantities and their relationships in problem situations. Students need to consider layouts in design using HTML and CSS, which means that every pixel counts and any shift can throw off the entire look and feel and usability. When designing user interfaces, students need to think abstractly and conceptually about their designs, and drill down to details and consider the same designs quantitatively within the site code.

Algorithms and programming are central to the Computing Ideas course, so students need to continually reason abstractly and quantitatively. Students learn the core principles of developing their own algorithms and implementing them in the Karel programming language. Algorithms, variables, control, modularity, and program development are all taught in this course.

The Computing Ideas course teaches students how data is stored in a computer as an abstract representation. Students learn exactly how text and image data is organized and stored as physical bits in a computing system. Students also learn how sensors must convert physical data into a digital representation that can be stored in a computer, and how data collection can be automated with sensors.

3. **Construct viable arguments and critique the reasoning of others.**
   Computing Ideas students are able to analyze situations by breaking them into cases so they can create websites that are user-friendly for all. They justify their conclusions and decisions for choices that they make on their sites, communicate them to others, and respond to the arguments of others about advantages and disadvantages for designing and developing their site in a certain way.

   Computing has had significant impacts in several fields, and students critique the reasoning of others when discussing various topics. In this course, students learn about the positive and negative impacts the Internet has had on culture, social interactions, safety, and privacy. Students also learn the ethical considerations of sharing their code with others, and finding solutions to CodeHS exercises online.

4. **Model with mathematics.**
   Students in the CodeHS Computing Ideas course use geometry to solve website design layout problems. Students in the course can also apply what they know and are comfortable making assumptions and approximations to simplify a complicated designs, realizing that these may need revision later.

5. **Use appropriate tools strategically.**
   Students might use pencil and paper, concrete models, a ruler, a calculator, a spreadsheet, and other tools to create their sites and apps in the Computing Ideas course. Students detect possible errors in their code by strategically using estimation and other mathematical knowledge.

6. **Attend to precision.**
   The Computing Ideas course teaches students how to make choices about how data elements are organized and where data is stored on websites. Students can consider these choices in terms of speed, reliability, accessibility, privacy, and integrity and how those choices impact their own site development.

7. **Look for and make use of structure.**
   Students look closely to discern a pattern or structure in websites and applications. Complex websites are designed as systems of interacting or nested modules, each with a specific role, coordinating for a common overall purpose. These modules can be combinations of data (images, text, etc.) which allow for better management of complex sites. In the Internet unit, students learn learn about network communication and organization, basic Internet protocols, and Internet addressing, which all also make use of structural elements.

   Students also make use of structure when working with various number systems to understand and use digital information. They encode text with binary and work with hexadecimals in pixel images. Students then use the digital information to manipulate images with consideration for data and lossy compression.

8. **Look for and express regularity in repeated reasoning.**
   In the Computing Ideas course, students notice if calculations are repeated, and look both for general methods and shortcuts to make make their code more concise and reusable.

Standards for Mathematical Content

Computing Ideas (Lovelace)
The Computing Ideas course is a first computer science course introducing the basics of programming with Karel the Dog, the basics of designing a web page, and how information and images are represented with computers. Students will learn to code using blocks to drag and drop, but they can switch between blocks and text as desired. With a unique focus on creativity, problem solving and project based learning, Computing Ideas gives students the opportunity to explore several important topics of computing using their own ideas and creativity and develop an interest in computer science that will foster further endeavors in the field.
Standards for Mathematical Content Addressed
6.RP.1 Understand the concept of a ratio and use ratio language to describe a ratio relationship between two quantities. **6.NS.5** Understand that positive and negative numbers are used together to describe quantities having opposite directions or values (e.g., temperature above/below zero, elevation above/below sea level, credits/debits, positive/negative electric charge); use positive and negative numbers to represent quantities in real-world contexts, explaining the meaning of 0 in each situation.

6.NS.6c Find and position integers and other rational numbers on a horizontal or vertical number line diagram; find and position pairs of integers and other rational numbers on a coordinate plane.
7RP.2 Recognize and represent proportional relationships between quantities.
7.NS.1 Apply and extend previous understandings of addition and subtraction to add and subtract rational numbers; represent addition and subtraction on a horizontal or vertical number line diagram.
7.NS.1b Understand p + q as the number located a distance |q| from p, in the positive or negative direction depending on whether q is positive or negative. Show that a number and its opposite have a sum of 0 (are additive inverses). Interpret sums of rational numbers by describing real-world contexts.
N-Q.1 Use units as a way to understand problems and to guide the solution of multi-step problems; choose and interpret units consistently in formulas; choose and interpret the scale and the origin in graphs and data displays.