# Georgia's Introduction to Software Technology Syllabus

High School - One Year (160 hours)

## Course Overview and Goals

This course is the foundational course for Cloud Computing, Computer Science, Game Design, Internet of Things, Programming, Web and Digital Design, and Web Development pathways. This course is designed for high school students to understand, communicate, and adapt to a digital world as it impacts their personal life, society, and the business world. Exposure to foundational knowledge in programming languages, software development, app creation, and user interfacing applications are all taught in a computer lab with hands-on activities and project-focused tasks.

### Learning Environment

This course utilizes a blended classroom approach. The content is fully web-based, with students writing and running code in the browser. Each module of the course is broken down into lessons. Lessons consist of video tutorials, short quizzes, example programs to explore, and programming exercises.

### Programming Environment

For the programming units, students write and run JavaScript programs in the browser using the CodeHS editor. They will also explore and use the SQL programming language. For the web design unit, students write HTML, CSS, and JavaScript code in the browser using the CodeHS online editor. Teachers can choose to have students write code using either blocks or text.

### Prerequisites

This course is designed for complete beginners with no previous background in computer science. The course is highly visual, dynamic, and interactive, making it engaging for new coders.

### More Information

Browse the content of this course at https://codehs.com/course/15529

## Course Breakdown

### Module 1: Digital Citizenship and Cyber Hygiene (1 week/5 hours)

Students learn topics on Internet etiquette and how to stay safe on the world wide web. Students will look at the potential effects of their digital footprints, how to protect information from online risks, and the importance of creative credit.

| | |
|---|---|
| Objectives / Topics Covered | • Digital Footprint and Reputation<br>• Privacy and Security<br>• Creative Credit and Copyright |

| | |
|---|---|
| | ● Hacking Ethics<br>● Common Security Problems |
| Example Assignments | ● Digital Footprint and Reputation<br>○ What is a digital footprint?<br>○ What is ***your*** digital footprint and reputation?<br>○ What does it mean that the internet is public and permanent?<br>○ Who looks at your digital footprint and reputation?<br>○ What are some recommended social media guidelines?<br>○ How can you maintain your digital footprint?<br>○ What does your digital footprint say about you?<br>○ Example activities:<br>■ What is your digital footprint?<br>■ Are you going to make any changes in what you post on social media?<br>● Privacy and Security<br>○ What are data privacy and security?<br>○ How can you keep personal data secure and private?<br>○ What can happen if your data is stolen and what can you do about it?<br>○ Example activities:<br>■ Test out various passwords on a site<br>■ Explore Google's privacy policy: What do they know about you?<br>● Creative Credit and Copyright<br>○ What is copyright?<br>○ What are the different types of copyright licenses<br>○ Example activities:<br>■ Create citations for sources<br>■ Explore image search tools<br>● Hacking Ethics<br>○ What are hackers?<br>○ Are there different kinds of hackers? (white, black, grey)<br>○ What are bug bounty programs?<br>○ Is hacking always illegal?<br>○ What are the consequences of illegal hacking?<br>○ Example activities:<br>■ Explore what penetration testing is<br>■ Sign an ethical hacker agreement |

## Module 2: Programming with Karel (7 weeks/35 hours)

Students learn foundational skills in programming with Karel by simplifying JavaScript to four simple commands. Using these commands, students solve increasingly difficult problems and explore how complex programming languages can be developed from simple commands.

| | |
|---|---|
| Objectives / Topics Covered | ● Commands<br>● Defining vs. Calling Methods<br>● Designing methods |

| | |
|---|---|
| | ● Program entry points<br>● Control flow<br>● Looping<br>● Conditionals<br>● Classes<br>● Commenting code<br>● Preconditions and Postconditions<br>● Top Down Design |
| Example Assignments | ● 30 Karel Programming Exercises and Challenges in total<br>● Program-specific tasks for Karel the Dog<br>○ Example Exercise: Pyramid of Karel<br>Write a program to have Karel build a pyramid. There should be three balls on the first row, two in the second row, and one in the third row.<br>● Teach Karel new commands like `turnRight()` or `makePancakes()`<br>○ Example Exercise: Pancakes<br>Karel is the waiter. He needs to deliver a stack of pancakes to the guests on the 2nd, 4th, and 6th avenue. Each stack of pancakes should have three pancakes. Create a method called `makePancakes()` to help Karel solve this problem.<br>● Solve large Karel problems by breaking them down into smaller, more manageable problems using Top Down Design<br>○ Example Exercise: The Two Towers<br>In this program, Karel should build two towers of tennis balls. Each tower should be 3 tennis balls high.<br>At the end, Karel should end up on top of the second tower, facing East.<br>● Using control structures and conditionals to solve general problems<br>○ Example Exercise: Random Hurdles<br>Write a program that has Karel run to the other side of first street, jumping over all of the hurdles. However, the hurdles can be in random locations. The world is fourteen avenues long. |

## Module 3: Karel Challenges (1 week/5 hours)

In this challenge module, students apply all the foundational concepts from Karel to solve some programming challenges.

| | |
|---|---|
| Objectives / Topics Covered | ● Solving large and more complex problems using Karel |
| Example Challenges | ● Several Karel challenges to tie everything learned in the Karel module together<br>○ Example Exercise: Super Cleanup Karel<br>Karel’s world is a complete mess. There are tennis balls all over the place, and you need to clean them up. Karel will start in the bottom left corner of the world facing east and should clean up all of the tennis balls in the world. |

| | |
|---|---|
| | This program should be general enough to work on any size world with tennis balls in any locations. |

## Module 4: JavaScript and Graphics (4 weeks/20 hours)

This module introduces students to the basics of JavaScript, including variables, user input, mathematics, basic graphics, and image representations.

| | |
|---|---|
| Objectives / Topics Covered | ● Variables<br>● User Input<br>● Arithmetic Expressions<br>● Graphics<br>● Image Representation |
| Example Assignments | ● Using variables and getting user input using JavaScript<br>○ Example Exercise: Dinner Plans<br>Prompt the user for their name, then ask them what time you should meet for dinner.<br>Greet them by name and tell them you will meet them at the time they specified!<br>● Create basic graphics in JavaScript<br>○ Example Exercise: Flag of the Netherlands<br>This program should draw the flag for the Netherlands. The top third of the canvas is red, the middle third is white, and the bottom third is blue, as shown below. You will need to use Rectangle objects in this program. |

## Module 5: Computing Basics (1 week/5 hours)

Students will learn about the physical elements of computers and networking such as motherboards, RAM, routers, and the use of port numbers, ethernet, and wireless devices.

| | |
|---|---|
| Objectives / Topics Covered | ● Internal Components of a Computer<br>● Peripheral Devices<br>● Software vs. Hardware<br>● Future of Computing |
| Example Assignments | ● Different Types of CPU<br>● RAM vs. Hard Drive<br>● Personalized Peripherals<br>● Emerging Technologies<br>○ DNA Storage<br>○ Pros and Cons of AI |

## Module 6: Operating Systems and Software (2 weeks/10 hours)

Students will compare and contrast common operating systems (Windows, Linux, OS) and explain the importance of compatibility. Students will also explore software and apps while discussing software licenses and running through the software development life cycle.

| | |
|---|---|
| Objectives / Topics | ● Operating Systems |

| | |
|---|---|
| Covered | - Software and Applications |
| Example Assignments | - Understanding Operating Systems<br>- Comparing Operating Systems<br>  ○ Installing an OS<br>- File Management<br>  ○ What Processor are you Running?<br>- Software Licenses<br>- Software Development<br>  ○ Development Life Cycle<br>  ○ Agile/SCRUM |

### Module 7: Project: IT Professional (2 weeks/10 hours)

In this project, students will explore career pathways and build skills that will be needed within these fields such as communication.

| | |
|---|---|
| Objectives / Topics Covered | - Computer Science Career Pathways<br>- Customer Service and Communication<br>- Contributing to a Knowledge Base<br>- Creating an Instructional Video<br>- Using a Database |
| Example Assignments | - Act it out! Pair up with a partner and create a short script of a customer support scenario based on a common mobile device issue.<br>- Write a KB Article: Create an internal knowledge base article that will be shared with other technicians.<br>- Star in a Video! Create a 2-5 minute video tutorial based on a common mobile device issue<br>- Keeping a Database: Create a SQL table using unstructured data. |

### Module 8: Web Design (6 weeks/30 hours)

Students will learn and use HTML and CSS to build and design websites.

| | |
|---|---|
| Objectives / Topics Covered | - Creating webpages using HTML<br>  ○ Links<br>  ○ Images<br>  ○ Lists<br>  ○ Tables<br>  ○ Inline styling<br>- Styling webpages with CSS<br>  ○ Creating CSS rules<br>  ○ CSS classes<br>  ○ CSS IDs |
| Example Assignments | - Links<br>  ○ Learn how to link different pages together<br>  ○ Example Exercise: Create a webpage that provides links to your favorite books<br>- Images |

| | |
|---|---|
| | ○ Learn how to add and format images<br>○ Example Exercise: Create a collage of images<br>● Tables<br>○ Learn how to create and style tables<br>○ Example Exercise: Create a table describing your favorite music artists and songs<br>● Styling with CSS<br>○ Use CSS to add background colors, font colors, font styles, borders, and position elements on the page<br>○ Example Exercise: Create CSS classes and IDs to apply formatting to a BINGO board<br>○ Example Exercise: Create CSS classes to style a music library web page<br>○ Example Exercise: Create CSS Rules to put a Karel puzzle together<br>● Final Project<br>○ Build your own homepage using everything you've learned in the module<br>○ This homepage can serve as your personal portfolio of creative projects as you continue through the course! |

### Module 9: Web Development (7 weeks/35 hours)

Students will learn and apply the basic elements of web development, such as using JavaScript and the DOM, collecting and storing data, and using file structure.

| | |
|---|---|
| Objectives / Topics Covered | ● The Script Tag<br>● Using the DOM<br>● Collecting and Storing Data<br>● Storing Data with Firebase<br>● Using APIs<br>● Using a CMS<br>● Web Optimization |
| Example Assignments | ● Debugging with the Console<br>● getElementById<br>● Local Storage vs. Cookies<br>● Generating and Using an API Key<br>● Website Folder Structure |

### Module 10: Computer Science Careers (1 week/5 hours)

In this short module, students take some time to explore and discover different computer science careers. They will also examine inclusive coding and how to avoid bias in computer programming.

| | |
|---|---|
| Objectives / Topics Covered | ● Computer Science Careers<br>● Computer Science used in non-CS Careers<br>● Inclusive Coding |
| Example Assignments | ● Career Research<br>● Computer Science concepts that are used in non-computer |

| | |
|---|---|
| | science positions<br>● How can a computer program's bias become dangerous?<br>● Why is it important to have a more diverse group of people in the computer science field? |