PUBLIC SCHOOLS OF EDISON TOWNSHIP

OFFICE OF CURRICULUM AND INSTRUCTION

Computer Science Discoveries

Length of Course:	Term
Elective/Required:	Elective
Schools:	Middle School
Eligibility:	Grade 8
Credit Value:	N/A
Date Approved:	August 26, 2019

TABLE OF CONTENTS

INTRODUCTION

CS Discoveries introduces students to computer science as a vehicle for problem solving, communication, and personal expression. The course focuses on the visible aspects of computing and computer science, and encourages students to see where computer science exists around them and how they can engage with it as a tool for exploration and expression.

Computing is so fundamental to understanding and participating in society that it is valuable for every student to learn as part of a modern education. Computer science is a subject that provides students with a critical lens for interpreting the world around them. Computer science prepares all students to be active and informed contributors to our increasingly technological society whether they pursue careers in technology or not. Computer science can be life-changing, not just skill training.

This curriculum guide was created by Deborah Jasper (WWMS) and Eric McMahon (TJMS)
Advised by Jennifer Fischer - Supervisor of 21st Century Skills

SCOPE & SEQUENCE*

UNIT 1: PROBLEM SOLVING and COMPUTING and Community Building - **MARKING PERIOD 1**

Intro to Problem Solving
The Problem Solving Process
Exploring Problem Solving
What is a Computer?
Input and Output
Processing
Apps and Storage
Project - Propose an App

UNIT 2: WEB DEVELOPMENT- **MARKING PERIODS 1 & 2**

Exploring Websites
Websites for Expression
Intro to HTML
Headings
Digital Footprint
Lists
Intellectual Property and Images
Clean Code and Debugging
Project - Multi-Page Websites
Styling Text with CSS
Styling Elements with CSS
Sources and Search Engines
RGB Colors and Classes
Project - Personal Portfolio Website

UNIT 3: INTERACTIVE GAMES AND ANIMATIONS- **MARKING PERIODS 2, 3 & 4**

Programming for Entertainment
Plotting Shapes
Drawing in Game Lab
Shapes and Randomization
Variables
Sprites
The Draw Loop
Counter Pattern Unplugged
Sprite Movement
Booleans Unplugged
Booleans and Conditionals
Conditionals and User Input
Other Forms of Input
Project - Interactive Card
Velocity
Collision Detection
Complex Sprite Movement
Collisions
Functions
The Game Design Process
Using the Game Design Process
Project - Design a Game

UNIT 4: THE DESIGN PROCESS- **MARKING PERIOD 4**

Analysis of Design
Understanding your user
User centered design Micro Activity
User Interfaces
Feedback and testing
Identifying user needs
Project- Paper prototype

* For students and teachers requiring further enrichment, you may explore Chapter 2 of Unit 4, Units 5 and/or Unit 6

UNIT OF STUDY: UNIT 1: PROBLEM SOLVING

Students learn the problem-solving process, the input-output-store-process model of a computer, and how computers help humans solve problems. Students end the unit by proposing their own app to solve a problem.

Targeted CSTA K-12 Computer Science Standards (2017):

AP - Algorithms & Programming

- **1B-AP-08 - Compare and refine multiple algorithms for the same task and determine which is the most appropriate.**
- **1B-AP-11 - Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process.**
- **1B-AP-16 - Take on varying roles, with teacher guidance, when collaborating with peers during the design, implementation and review stages of program development.**
- **2-AP-10 - Use flowcharts and/or pseudocode to address complex problems as algorithms.**
- **2-AP-17 - Systematically test and refine programs using a range of test cases.**
- **2-AP-15 - Seek and incorporate feedback from team members and users to refine a solution that meets user needs.**
- **2-AP-18 - Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts.**

CS - Computing Systems

- **1B-CS-01 - Describe how internal and external parts of computing devices function to form a system.**
- **1B-CS-02 - Model how computer hardware and software work together as a system to accomplish tasks.**

IC - Impacts of Computing

- **2-IC-20 - Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options.**

21st Century Skills/Career Ready Practices: Critical thinking, Creativity, Collaboration, Communication, Flexibility, Leadership, Productivity, Social skills

Unit Objectives/Enduring Understandings:

Students will be able to…

- Define and use the 4-step Problem Solving Process - Define, Prepare, Try, Reflect
- Understand all Components of a Computer - Input, Output, Storage and Processing

Attitudinal Goals:

Students will...

- Build a collaborative classroom environment
- View computer science as relevant, collaborative, fun and empowering

Essential Questions:

The Problem Solving Process	**Computers and Problem Solving**
• What strategies and processes can I use to become a more effective problem solver?	• How do computers help people to solve problems? • How do people and computers approach problems differently? • What does a computer need from people in order to solve problems effectively?

Unit Assessment: Students propose their own app to solve a problem; Students take a summative Unit Assessment

UNIT 1	**Core Content**		**Instructional Actions**	
Cumulative Progress Indicators	**Concepts** What students will know.	**Skills** What students will be able to do.	**Activities/Strategies** Technology Implementation/ Interdisciplinary Connections	**Assessment Check Points**
Lesson 4 ● 1B-CS-01	Understanding Computing Devices Students should be able to identify input, output, storage, and processing as four essential components of a computing device and explain the role that each component takes when computers are used to solve informational problems.	**Models of Computing Devices:** Identify a computer as a machine that processes information and give a high level description of the input-output-storage-processing model of computing devices. **Input and Output:** Identify the inputs and outputs of common computing devices and select the inputs and outputs used to perform common computing tasks. **Storage:** Provide examples of common types of information that is stored on a computer and explain the need for storage as part of processing information with a computer. **Processing:** Define processing as the work done (possibly by a computer) to turn an input into an output and define an algorithm as the series of commands a computer uses to process information	Lesson 4: What is a computer? ● Picture worksheet ● Groups sort, discuss, and defend what a computer is.	Objectives to be Assessed: ● Understand components of a computer ❑ **Class discussions** ❑ **Journal Questions** ❑ **Quick-check levels** include multiple choice or short answer questions. ❑ **Programming levels** challenge students to complete a small programming task. ❑ **Activity Guides**
Lessons 1, 2, 3 ● 1B-AP-08 ● 1B-AP-11 ● 1B-AP-16	The Problem Solving Process	**Use a structured problem solving process:** Given various problems, identify individual actions that would fall within each step of a structured	Lesson 1: Aluminum boat building ● Collaboration ● Problem solving Lesson 2: The problem solving	Objectives to be Assessed: ● Define and use the 4-step Problem Solving Process

	Students should be able to define and use a structured problem solving process, identifying key components of the process and how they apply to various problems. Students should use multiple strategies to approach problems, iteratively improving on the solution through collaboration and reflection.	process to solve them. **Define a problem to be solved:** Assess how well defined a problem is and use strategies to define the problem more precisely **Plan a solution:** Consider various approaches to solving a problem, and decide which is the most appropriate **Implement a solution:** Carry out and evaluate a solution to a problem, iteratively improving on it as needed	process ● Define, Prepare, Try, and Reflect ● Apply the process to boat activity and other situations for design Lesson 3: Using the problem solving process ● Word search partner activity ● Use DPTR process and discuss how each group worked	❏ **Class discussions** ❏ **Journal Questions** ❏ **Quick-check levels** include multiple choice or short answer questions. ❏ **Programming levels** challenge students to complete a small programming task. ❏ **Activity Guides**
Lesson 5, 6, 7, 8, 9 ● 1B-CS-01 ● 1B-CS-02 ● 2C-CS-02 ● 2-AP-10 ● 2-AP-15 ● 2-AP-17 ● 2-AP-18 ● 2-IC-20	Computing and Algorithms Students should combine their understandings of computing and problem solving to identify and design solutions for computational problems. In doing so, they should develop algorithms that can automate the processing of information, producing a desired output from a given input.	**Identify and define computational problems:** Choose problems that can be solved with computing and justify those choices. **Develop computational solutions:** Identify the inputs associated with a given problem, and define the processing and storage needed to produce the desired output. **Developing algorithms:** Develop and iteratively improve algorithms for processing information.	Lesson 5: Inputs and Outputs ● Worksheet handout for what is an input/output ● Use Definition of input and output to discuss with partners Lesson 6: Card Sorting ● Playing cards ● Analyze method of sorting cards and define computer algorithm Lesson 7: Storage and Processing ● Playing cards ● Analyze cards develop algorithm to find smallest card in stack Lesson 8: App Exploration ● Worksheet on Apps ● Analyze different apps to see what input is needed for functionality of app Lesson 9: Propose an App	Objectives to be Assessed: ● Understand components of a computer ● Use the 4-step Problem Solving Process **Lesson Level Assessments:** ❏ **Class discussions** ❏ **Journal Questions** ❏ **Quick-check levels** include multiple choice or short answer questions. ❏ **Programming levels** challenge students to complete a small programming task. ❏ **Activity Guides** **Unit Level Assessments:** ❏ **End of unit project** ❏ **Student-facing rubrics** ❏ **Practice reflections** ❏ **Post-project test**

			<ul><li>Worksheet</li><li>Poster</li><li>Propose an App and determine what inputs are needed for desired output</li></ul>	

Resources: Essential Materials, Supplementary Materials, Links to Best Practices Code.org platform; Journals; Activity Guides; poster paper; rubrics; supplies for aluminum boat activity; playing cards	**Instructional Adjustments:** *Modifications/Student difficulties/Common errors/Possible misunderstandings* Appropriate accommodations and/or modifications as determined by 504s and IEPs: shortened assignments, extended time, copy of class notes or access to notes on Chromebook, preferential seating, oral reminders, etc. Ask students to restate information, directions, and assignments.

UNIT OF STUDY: UNIT 2: WEB DEVELOPMENT (HTML & CSS)

Unit 2 introduces computer languages and how students can use these languages to create digital artifacts. By the end of the unit, students should be able to create a digital artifact that uses multiple computer languages to control the structure and style of their content. They should understand that different languages allow them to solve different problems, and that these solutions can be generalized across similar problems. Lastly, they should understand their responsibilities as both creators and consumers of digital media.

Targeted CSTA K-12 Computer Science Standards (2017):

AP - Algorithms & Programming

- **1B-AP-11 - Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process.**
- **1B-AP-12 - Modify, remix or incorporate portions of an existing program into one's own work, to develop something new or add more advanced features.**
- **1B-AP-15 - Test and debug (identify and fix errors) a program or algorithm to ensure it runs as intended.**
- **2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.**
- **2-AP-15 - Seek and incorporate feedback from team members and users to refine a solution that meets user needs.**
- **2-AP-16 - Incorporate existing code, media, and libraries into original programs, and give attribution.**
- **2-AP-17 - Systematically test and refine programs using a range of test cases.**
- **2-AP-18 - Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts.**
- **2-AP-19 - Document programs in order to make them easier to follow, test, and debug.**
- **3A-AP-20 - Evaluate licenses that limit or restrict use of computational artifacts when using resources such as libraries.**

IC - Impacts of Computing

- **1B-IC-18 - Discuss computing technologies that have changed the world and express how those technologies influence, and are influenced by, cultural practices.**
- **1B-IC-21 - Use public domain or creative commons media and refrain from copying or using material created by others without permission.**
- **2-IC-20 - Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options.**
- **2-IC-21 - Discuss issues of bias and accessibility in the design of existing technologies.**
- **2-IC-23 - Describe tradeoffs between allowing information to be public and keeping information private and secure.**

NI - Networks & the Internet

- **1B-NI-05 - Discuss real-world cybersecurity problems and how personal information can be protected.**

21st Century Skills/Career Ready Practices: Critical thinking, Creativity, Collaboration, Communication, Information literacy, Media literacy, Technology literacy, Flexibility, Leadership, Initiative, Productivity, Social skills

Unit Objectives/Enduring Understandings:

Students will be able to…

- Implement the Problem Solving Process for Programming
- Set Up a Web Page - Content, Structure, Style
- Code in HTML and CSS
- Define and Understand Digital Citizenship

Attitudinal Goals:

Students will...

- View computer science as a form of expression
- See themselves as creators and not simply consumers of online content

- Consider the impact of the choices they make when creating and consuming digital content

Essential Questions:

Web Content and HTML	**Styling and CSS**
• How can text communicate content and structure on a web page? • Why do people create websites? • How can I incorporate content I find online into my own webpage? • What strategies can I use when coding to find and fix issues?	• How do I modify the appearance and style of my web pages? • How do I safely and appropriately make use of the content published on the internet?

Unit Assessment: Students develop their own personal websites; Students take a summative Unit Assessment

UNIT 2	**Core Content**		**Instructional Actions**	
Cumulative Progress Indicators	**Concepts** What students will know.	**Skills** What students will be able to do.	**Activities/Strategies** Technology Implementation/ Interdisciplinary Connections	**Assessment Check Points**
Lessons 3, 4, 10, 11, 12, 13 • 1B-AP-11 • 1B-AP-15 • 2-AP-16 • 2-AP-17 • 2-AP-19 • 2-IC-20 • 2-IC-21 • 2-IC-23	Using Computer Languages Students should understand the need for computer languages, and how to choose a language based on the task at hand. They should understand that different languages use different syntax, and understand the need for precision and syntax in using multiple computer languages.	**Understand and explain the need for computer languages:** Understand why specialized languages exist to communicate with computers and describe the features a language might need. **Attend to precision and syntax when creating a digital artifact:** Understand the need for precision when using computer languages and use appropriate syntax. **Combine computer languages within a digital artifact:** Use multiple computer languages to manage the complexity of a digital artifact. **Choose an appropriate**	Lesson 3: Intro to HTML • Unplugged analysis of web pages • Explore weblab • Start to use HTML code to open and close pages Lesson 4: Headings • Introduce different tags • Device to write code and access internet • Learn basic HTML commands Lesson 10: Styling Text with CSS • Learn the differences between HTML and CSS • Use a separate style sheet to enhance HTML Code	Objectives to be Assessed: • Set Up a Web Page - Content, Structure, Style • Code in HTML and CSS ❏ **Journal Questions** ❏ **Quick-check levels** include multiple choice or short answer questions. ❏ **Programming levels** challenge students to complete a small programming task. ❏ **Activity Guides**

		computer language: Understand differences between HTML and CSS and choose the most appropriate language for a given task.	● Device and Internet ● Develop CSS style poster sheet Lesson 11: Styling Elements with CSS ● Create CSS rule sheet that will help enhance style and appearance of webpage ● Internet device Lesson 12: Sources and Search Engines ● Worksheets for scavenger hunt and strange animals ● Use internet device to research privacy and validity of information on the web Lesson 13: RGB Colors and Classes ● Activity worksheet for RGB colors ● Students use internet device to investigate how to get different colors on webpages	
Lessons 1, 2 ● 2-IC-20 ● 2-AP-13 ● 1B-IC-18	Modularity and Abstraction Students should be able to break down complex problems into their component parts, distinguishing between content, structure, and formatting. They should also recognize and use abstraction as it is built into computer languages, grouping elements by type or by classes that they create.	**Logically separate the content, structure and formatting of a digital artifact:** Distinguish between the content, structure, and formatting in the design of a digital artifact and ensure that they are logically separated in its encoding (e.g. by using HTML for structuring and CSS for formatting). **Create classes that can be referenced and affected as a group:** Use classes to identify and set the properties	Lesson 1: The purpose of Websites ● Worksheet for website analysis ● Device for Internet access ● Partner work what are websites used for? Lesson 2: Websites for Expression ● Worksheet for developing personal website ● Device for Internet access	Objectives to be Assessed: ● Implement the Problem Solving Process for Programming ❏ **Journal Questions** ❏ **Quick-check levels** include multiple choice or short answer questions. ❏ **Programming levels** challenge students to complete a small programming task. ❏ **Activity Guides**

		multiple elements as a group **Create rules that affect entire groups of elements:** Use classes to create formatting rules that will be applied to groups of elements, either by tag or by class. **Use stylesheets to apply formatting rules across multiple web pages:** Create and reference style sheets so that consistent formatting rules will apply to multiple web pages.	● Start to plan what you want on your website	
Lessons 9 & 14 ● 2-AP-15 ● 2-AP-16 ● 2-AP-17 ● 2-AP-18 ● 2-AP-19 ● 1B-IC-21	Creating a Digital Artifact Students should be able to design and create their own digital artifact using multiple computer languages.	**Structure content on a web page using HTML:** Use HTML to create a web page that includes hierarchical headings, paragraphs, lists, and images. **Apply formatting in a web page using CSS:** Use external style sheets to control placement, size, and appearance of elements. **Define colors with RGB codes:** Use RGB color codes to specify colors for elements on a web page.	Lesson 9: Project - Multi-Page Websites ● Students will use Peer review worksheet to examine web pages ● Continue to develop website to be published ● Internet device for research Lesson 14: Project - Final Personal Website ● Students use activity guide to create, analyze, and use peer review for personal website final project ● Internet device and worksheets	Objectives to be Assessed: ● Set Up a Web Page - Content, Structure, Style ● Code in HTML and CSS **<u>Chapter and Unit Level Assessments:</u>** ❏ **End of chapter projects** ❏ **Student-facing rubrics** ❏ **Practice reflections** ❏ **Post-project tests**
Lessons 6, 8 ● 1B-AP-12 ● 1B-AP-15 ● 2-AP-19 - D	Debugging and Clean Code Students should understand the importance of clean, readable code and should use appropriate formatting and commenting conventions to make their code	**Format code to make it easier to read and maintain:** Use whitespace and indentation to make code easier to read and maintain. **Comment code where appropriate:** Use comments	Lesson 6: Lists ● Device to learn how to create different kinds of list in HTML Lesson 8: Clean Code and Debugging ● Students work with partners to learn about a bug and debugging	Objectives to be Assessed: ● Implement the Problem Solving Process for Programming ❏ **Journal Questions** ❏ **Quick-check levels** include multiple choice or short answer questions.

	easier to read and maintain. Students should use multiple strategies to find and eliminate bugs from their code.	to make code more readable. **Use multiple debugging strategies:** Develop a set of techniques for preventing bugs in HTML and CSS code and finding them when they occur.	HTML Code. ● Activity with white board and sticky notes	❏ **Programming levels** challenge students to complete a small programming task. ❏ **Activity Guides**
Lessons 5, 7 ● 2-AP-16 ● 3A-AP-20 ● 1B-IC-21 ● 1B-N!-05 ● 2-IC-20 ● 2-IC-23	Responsible Creation and Consumption of Digital Media Students should recognize their responsibilities as creators and consumers of digital media. They should be able to make ethical and safe choices when publishing information online. They should understand that not all information found online is trustworthy, and have strategies for finding relevant and reliable information on the web.	**Use good judgement in sharing personal information online:** Justify and adhere to guidelines for safely publishing information online, **Respect Copyright:** Explain the purpose of copyright and follow copyright law, accurately attributing others when using their work. **Vet sources:** Use basic web searching techniques to find relevant information online, identify elements that contribute to a website's trustworthiness or untrustworthiness.	Lesson 5: Digital Footprint ● Social Sleuth and social privacy worksheets ● Internet and code device to do detective work and investigate where privacy and content needs to be protected Lesson 7: Intellectual Property and Images ● Worksheet on LIcensing your work ● Students research different ways to protect their content ● Device for Internet and coding	Objectives to be Assessed: ● Define and Understand Digital Citizenship ❏ **Journal Questions** ❏ **Quick-check levels** include multiple choice or short answer questions. ❏ **Programming levels** challenge students to complete a small programming task. ❏ **Activity Guides**

Resources: Essential Materials, Supplementary Materials, Links to Best Practices Code.org platform; Journals; Activity Guides; **Web Lab** — A browser-based tool for creating and publishing HTML and CSS web sites.	**Instructional Adjustments:** *Modifications/Student difficulties/Common errors/Possible misunderstandings* Appropriate accommodations and/or modifications as determined by 504s and IEPs: shortened assignments, extended time, copy of class notes or access to notes on Chromebook, preferential seating, oral reminders, etc. Ask students to restate information, directions, and assignments.

UNIT OF STUDY: UNIT 3: INTERACTIVE GAMES AND ANIMATIONS (JavaScript)

Unit 3 focuses on algorithms and programming. By the end of the unit, students should be able to create an interactive animation or game that includes basic programming concepts such as control structures, variables, user input, and randomness. They should manage this task by working with others to break it down using objects (sprites) and functions. Throughout the process, they should give and respond constructively to peer feedback and work with their teammates to complete a project.

Targeted CSTA K-12 Computer Science Standards (2017):

AP - Algorithms & Programming

- **2-AP-10 - Use flowcharts and/or pseudocode to address complex problems as algorithms.**
- **2-AP-11 - Create clearly named variables that represent different data types and perform operations on their values.**
- **2-AP-12 - Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.**
- **2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.**
- **2-AP-15 - Seek and incorporate feedback from team members and users to refine a solution that meets user needs.**
- **2-AP-16 - Incorporate existing code, media, and libraries into original programs, and give attribution.**
- **2-AP-17 - Systematically test and refine programs using a range of test cases.**
- **2-AP-18 - Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts.**
- **2-AP-19 - Document programs in order to make them easier to follow, test, and debug.**

IC - Impacts of Computing

- **2-IC-21 - Discuss issues of bias and accessibility in the design of existing technologies.**

21st Century Skills/Career Ready Practices: Critical thinking, Creativity, Collaboration, Communication, Technology literacy, Flexibility, Leadership, Initiative, Productivity, Social skills

Unit Objectives/Enduring Understandings:

Students will be able to…

- Implement the Problem Solving Process for Programming
- Learn and correctly implement sequencing and program flow
- Implement abstraction in programming
- Define and correctly use common programming structures: Variables, Conditionals, Functions
- Code using JavaScript

Attitudinal Goals:

Students will...

- View programming as fun, creative, and expressive
- See themselves as computer programmers
- Feel that programming is a form of communication

Essential Questions:

Images and Animations	Building Games
• What is a computer program? • What are the core features of most programming languages? • How does programming enable creativity and expression? • Which practices and strategies will help me as I write programs?	• How do software developers manage complexity and scale? • How can programs be organized so that common problems need to be solved only once? • How can I build on previous solutions to create even more complex behavior?

Unit Assessment: Students design their own animations and games; Students take a summative Unit Assessment

UNIT 3	**Core Content**		**Instructional Actions**	
Cumulative Progress Indicators	**Concepts** What students will know.	**Skills** What students will be able to do.	**Activities/Strategies** Technology Implementation/ Interdisciplinary Connections	**Assessment Check Points**
Lessons 1, 14, 19, 20, 21, 22 ● 2-IC-21 ● 2-AP-10 ● 2-AP-11 ● 2-AP-12 ● 2-AP-13 ● 2-AP-14 ● 2-AP-15 ● 2-AP-16 ● 2-AP-17 ● 2-AP-18 ● 2-AP-19	Program Development Students should be able to collaborate with peers to develop a piece of software. This process involves defining the needs of the program, designing a program to meet those needs, and breaking down the design into manageable pieces. Student code should be written so that others can read and understand it, and they should give and receive feedback on their work, as well as test and revise the program.	**Constructively give and respond to peer feedback:** Give feedback to peers using a structured process that points out strengths and areas for growth in a project, and incorporate given feedback for their own programs into their revisions. **Write readable code:** Organize code such that it is readable and make comments where appropriate to help readers understand the purpose of specific sections. Use reasonable variable and function names. **Using a structured process to plan and develop a program:** Use a structured process to describe a program's behavior, identify the core programming constructs necessary to complete the project, then use them as a guide to complete the program. **Collaborate effectively with team members in distributing and completing tasks on time:** Break up tasks so that each team member can make a meaningful contribution. Ensure that the code will work together once it is	Lesson 1: Programming for Entertainment ● Activity worksheet for examining the impact of computer science in different entertainment areas ● Device for research Lesson 14: Project - Interactive Card ● Students make an interactive greeting card using cumulative programming skills ● Peer review ● Device and Internet Lesson 19: Functions ● Learn how to create functions and organize code ● Make it more readable and remove repeated blocks of code ● Device and Internet Lesson 20: The Game Design Process ● Use project guide handout to plan for game design for the rest of the unit ● Define sprites, variables and functions	Objectives to be Assessed: ● Implement the Problem Solving Process for Programming ● Learn and correctly implement sequencing and program flow ● Define and correctly use common programming structures: Variables, Conditionals, Functions ● Code using JavaScript ❑ **Journal Questions** ❑ **Quick-check levels** include multiple choice or short answer questions. ❑ **Programming levels** challenge students to complete a small programming task. ❑ **Activity Guides** **Chapter and Unit Level Assessments:** ❑ **End of chapter projects** ❑ **Student-facing rubrics** ❑ **Practice reflections** ❑ **Post-project tests**

		finished.	before beginning ● Device and Internet Lesson 21: Using the Game Design ● Draw game in project guide game platform handout ● Design game to catch alien stars Lesson 22: Project - Design a Game ● Students will design and build their own game using project guide from previous two lessons ● Describe games behavior and scope of variables ● Handouts, Device, and Internet	
Lessons 4, 6, 7, 8, 9 ● 1B-AP-11 ● 1B-AP-12 ● 1B-AP-15 ● 1B-IC-21 ● 2-IC-23 ● 2-AP-15 ● 2-AP-16 ● 2-AP-17 ● 2-AP-18 ● 2-AP-19 ● 3-A-AP-20	Modularity Students should be able to break down complex problems into their component parts, both to increase readability and organization of code and to allow them to reuse portions of code many times. Algorithms should be broken into functions, and screen elements into sprites/objects. They should also recognize and use abstraction as it is built into programming languages.	**Use objects to manage the complexity of on-screen elements:** Create and modify objects (sprites) to represent on screen elements and their associated properties. Use dot notation to get and set sprite properties. Manage complexity through abstraction: Use abstraction to reason about a program at different levels of complexity. Describe the benefits of abstraction, including simplifying code to more easily program complex behavior. **Define and call functions in a program:** Create and use functions to organize reuse code within the same program. Modify and use functions created	Lesson 4: Shapes and Randomization ● Students explore drawing ellipses and rectangles using “parameters” ● Internet and device Lesson 6: Sprites ● Use sprites, shapes, and text to create a simple scene ● Device and Internet Lesson 7: The Draw Loop ● Introduce draw loop in game lab ● Study flip books and then try basic animation ● Device and Internet Lesson 8: Counter Pattern	Objectives to be Assessed: ● Implement the Problem Solving Process for Programming ● Learn and correctly implement sequencing and program flow ● Implement abstraction in programming ● Define and correctly use common programming structures: Functions ● Code using JavaScript ❏ **Journal Questions** ❏ **Quick-check levels** include multiple choice or short answer questions. ❏ **Programming levels** challenge students to

		by others	Unplugged ● Connect sprites location and sprites movement on a screen ● Unplugged worksheets for variables unplugged and counter pattern Lesson 9: Sprite Movement ● Combine draw loop and counter pattern to move sprites across screen ● Device and Internet	complete a small programming task. ❑ **Activity Guides**
Lessons 10, 11, 12, 13 ● 2-AP-10 ● 2-AP-11 ● 2-AP-12 ● 2-AP-13 ● 2-AP-16 ● 2-AP-17 ● 2-AP-19	Algorithms and Control Students should be able to use basic programming constructs to create a wide range of behaviors in their programs. These constructs should be combined to create complex behaviors, such as screen elements that move according to user input, or properties that change after a certain threshold has been reached. Programs should run differently each time according to user input or random chance.	**Use arguments to change the way a method runs:** Use arguments to change the way a method runs, distinguishing between the roles of multiple arguments passed to a method **Detect and respond to user input:** Use input from the keyboard and mouse to change the behavior of a program while it is running. **Use iteration to repeat behavior within a program:** Use loops to repeat behavior in a program, combing repetition with other control structure to produce ongoing complex behaviors. **Use conditionals to control the flow of a program:** Use conditional statements to control the flow of a program based on user input, variable values, or properties of objects (sprites). **Generate and use random**	Lesson 10: Booleans Unplugged ● Use activity worksheet to organize simple shapes based on boolean commands Lesson 11: Booleans and Conditionals ● Learn to predict the outcome of certain boolean statements ● Device and Internet Lesson 12: Conditionals and User Input ● Use conditionals to react to keyboard input ● Move sprites in response to keyboard input ● Device and Internet Lesson 13: Other Forms of Input ● Explore ways to use conditional statements to take user input ● Device and Internet	Objectives to be Assessed: ● Implement the Problem Solving Process for Programming ● Learn and correctly implement sequencing and program flow ● Define and correctly use common programming structures: Conditionals, ● Code using JavaScript ❑ **Journal Questions** ❑ **Quick-check levels** include multiple choice or short answer questions. ❑ **Programming levels** challenge students to complete a small programming task. ❑ **Activity Guides**

		numbers in a program: Use random numbers to introduce variation in how a program is run.		
Lessons 2, 3, 15, 16, 17, 18 ● 2-AP-10 ● 2-AP-11 ● 2-AP-12 ● 2-AP-13 ● 2-AP-16 ● 2-AP-17 ● 2-AP-19	Position and Movement Students should use the coordinate plane to place and move screen elements. They should be able to model various types of motion, including acceleration, linear movement and simulating gravity.	**Place elements on screen using a coordinate plane:** Use a coordinate system to place elements on a screen, accounting for object size and overlay. **Model two dimensional movement on a coordinate plane:** Manipulate the x and y coordinates of an object on a screen to create the illusion of smooth motion. **Model complex movement on a coordinate plane:** Combine different types of movement to create more complex behaviors such as acceleration, jumping, and bouncing.	Lesson 2: Plotting Shapes ● Students learn to how a computer processed information to come up with different styles ● Easel for computer drawing simulation Lesson 3: Drawing in Game Lab ● Students are introduced to game lab ● They need to change three colors of an existing page Lesson 15: Velocity ● Students are introduced to the properties that set velocity and rotation speed directly ● Create basic side scroller game ● Device and Internet Lesson 16: Collision Detection ● Use sprite location and size and math to determine if two sprites are touching ● Use activity sheets to draw sprites ● Create different effects when sprites collide Lesson 17: Complex Sprite Movement ● Combine the velocity property of sprites with the counter pattern to create complex sprite	Objectives to be Assessed: ● Learn and correctly implement sequencing and program flow ● Define and correctly use common programming structures: Variables, Conditionals, Functions ● Code using JavaScript ❑ **Journal Questions** ❑ **Quick-check levels** include multiple choice or short answer questions. ❑ **Programming levels** challenge students to complete a small programming task. ❑ **Activity Guides**

			movement ● Simulate gravity, make sprites jump and float ● Device and Internet Lesson 18: Collisions ● Program sprites to interact and brainstorm other ways they could interact ● Device and Internet	
Lesson 5 ● 2-AP-11 ● 2-AP-13 ● 2-AP-17 ● 2-AP-19	Variables/Storing Information Students should be able to create new variables as needed in their programs, and update and access the variable values as the program runs.	**Use variables to store and update information:** Create and assign values to variables as a program is run to store and update changing information	Lesson 5: Variables ● Learn about variables and how they are used with html ● Device and internet	Objectives to be Assessed: ● Define and correctly use common programming structures: Variables ● Code using JavaScript ❑ **Journal Questions** ❑ **Quick-check levels** include multiple choice or short answer questions. ❑ **Programming levels** challenge students to complete a small programming task. ❑ **Activity Guides**

Resources: Essential Materials, Supplementary Materials, Links to Best Practices	**Instructional Adjustments:** *Modifications/Student difficulties/Common errors/Possible misunderstandings*
Code.org platform; Journals; Activity Guides; **Game Lab** — A browser-based JavaScript programming environment designed to create sprite-based drawings, animations and games. Enables students to switch between programming in blocks or text.	Appropriate accommodations and/or modifications as determined by 504s and IEPs: shortened assignments, extended time, copy of class notes or access to notes on Chromebook, preferential seating, oral reminders, etc. Ask students to restate information, directions, and assignments.

UNIT OF STUDY: UNIT 4: The Design Process (Chapter 1, paper prototype only)

Unit 4, Chapter 1 extends the problem solving process to incorporate user centered design. By the end of the chapter, students should see the design process as a form of problem solving that prioritizes the needs of a user. They should be able to identify user needs and assess how well different designs address them. In particular they know how to develop a paper prototype, gather and respond to feedback about a prototype, and consider ways different user interfaces do or do not affect the usability of their apps. Students should leave the unit with a basic understanding of other roles in software development, such as product management, marketing, design, and testing, and to use what they have learned as a tool for social impact.

Targeted CSTA K-12 Computer Science Standards (2017):

AP - Algorithms & Programming

- **2-AP-10 - Use flowcharts and/or pseudocode to address complex problems as algorithms.**
- **2-AP-15 - Seek and incorporate feedback from team members and users to refine a solution that meets user needs.**
- **2-AP-17 - Systematically test and refine programs using a range of test cases.**

IC - Impacts of Computing

- **2-IC-20 - Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options.**
- **2-IC-21 - Discuss issues of bias and accessibility in the design of existing technologies.**
- **2-IC-22 - Collaborate with many contributors through strategies such as crowdsourcing or surveys when creating a computational artifact.**

CS - Computing Systems

- **2-CS-01 - Recommend improvements to the design of computing devices, based on an analysis of how users interact with the devices.**
- **2-CS-02 - Design projects that combine hardware and software components to collect and exchange data.**

21st Century Skills/Career Ready Practices: Critical thinking, Creativity, Collaboration, Communication, Information literacy, Media literacy, Technology literacy, Flexibility, Leadership, Initiative, Productivity, Social skills

Unit Objectives/Enduring Understandings:

Students will be able to…

- See the design process as a form of problem solving that prioritizes the needs of a user.
- Identify user needs and assess how well different designs address them:
  - know how to develop a paper prototype,
  - gather and respond to feedback about a prototype, and
  - consider ways different user interfaces do or do not affect the usability of their apps.

Essential Questions:

- How do designers identify the needs of their user?
- How can we ensure that a user's needs are met by our designs?
- What processes will best allow us to efficiently create, test, and iterate upon our designs?

Unit Assessment: Students create a Paper Prototype project; Students take a summative Unit Assessment

UNIT 4	Core Content		Instructional Actions	
Cumulative Progress Indicators	**Concepts** What students will know.	**Skills** What students will be able to do.	**Activities/Strategies** Technology Implementation/ Interdisciplinary Connections	**Assessment Check Points**
Lessons 1, 2, 3, 6 ● 2-CS-01 ● 2-CS-02 ● 2-IC-20 ● 2-IC-21 ● 2-IC-22 ● 2-AP-10 ● 2-AP-15 ● 2-AP-17	Understanding the user Students should actively consider the needs of others while developing a solution to a problem. They should identify user needs as central criteria for the development of a computational artifact and distinguish their own needs from those of the user, taking steps to identify those needs through a variety of strategies.	**Identify user needs as distinct from the needs of the developer:** Describe the target users for a computational artifact and identify those user's needs, distinguishing between the needs of the student and those of the user. **Collect and analyze user information to better understand user needs:** Collect information on users, either through research, interviews, or surveys, and analyze that information to create a profile for the user, including the needs that the user may have in interacting with the intended computational artifact.	Lesson 1: Analysis of Design ● Examine real world objects to see how the problem solving process can be used to help others ● Tea pot activity handout Lesson 2: Understanding Your User ● Handouts for user profiles ● Students role-play to determine the wants and needs of potential customers. ● Helps to address marketing issues Lesson 3: User-Centered Design Micro Activity ● Brainstorm a list of smart clothing users ● Put them into potential groups and identify needs or concerns of the user ● Design clothing for the user based on information Lesson 6: Identifying User Needs ● Activity guides user interface and paper prototype. ● Conduct an interview to collect information about user needs ● Analyze interview	Objectives to be Assessed: ● Identify user needs and assess how well different designs address them: ● consider ways different user interfaces do or do not affect the usability of their apps ❑ **Journal Questions** ❑ **Quick-check levels** include multiple choice or short answer questions. ❑ **Programming levels** challenge students to complete a small programming task. ❑ **Activity Guides**

			notes to identify specific user needs	
Lessons 4, 5, 7 ● 2-CS-01 ● 2-AP-10 ● 2-AP-15 ● 2-AP-17 ● 2-IC-21 ● 2-IC-22	Designing to criteria Students should be able to use set criteria to guide their design of a computational artifact, both in the overall design and purpose of the artifact and in the specific ways that users can interact with it.	**Generate and evaluate ideas for meeting specific criteria:** Generate multiple strategies for meeting user needs, then organize them into meaningful categories so that they can be analyzed. Select the most appropriate strategies from the list. **Design according to specific user needs:** Design an artifact, including core functionality and user interface, that meets the needs of a specific user, or design improvements to an existing artifact in order to meet those needs.	Lesson 4: User Interfaces ● Students use the mini design process to develop a prototype to meet a user's needs ● Activity sheets for user interface, testing computer, and testing user Lesson 5: Feedback and Testing ● Use feedback to create a development plan to improve an app ● Activity guides for feedback, improve a screen, use interface screens Lesson 7: Project - Paper Prototype ● Activity guide computer science best practices ● Design the functionality of an app to address the specific needs of a user ● Identify improvements to an app based on user testing ● Design the user interface of an app	Objectives to be Assessed: ● See the design process as a form of problem solving that prioritizes the needs of a user. ● Identify user needs and assess how well different designs address them: ● know how to develop a paper prototype ● gather and respond to feedback about a prototype ❑ **Journal Questions** ❑ **Quick-check levels** include multiple choice or short answer questions. ❑ **Programming levels** challenge students to complete a small programming task. ❑ **Activity Guides** **Chapter Level Assessments:** ❑ **End of chapter project** ❑ **Student-facing rubric** ❑ **Practice reflections** ❑ **Post-project test**

Resources: Essential Materials, Supplementary Materials, Links to Best Practices Code.org platform; Journals; Activity Guides**;**	**Instructional Adjustments:** *Modifications/Student difficulties/Common errors/Possible misunderstandings* Appropriate accommodations and/or modifications as determined by 504s and IEPs: shortened assignments, extended time, copy of class notes or access to notes on Chromebook, preferential seating, oral reminders, etc. Ask students to restate information, directions, and assignments.