GOMPERS PREPARATORY ACADEMY A UCSD Partnership
1005 47th Street, San Diego, CA 92102 p. (619) 263-2171 f. (619) 264-4342 www.gompersprep.org

Course Title: Computer Explorations
Instructor: Mr. Bentz
Instructor Availability: 3:30pm - 5:30pm
Instructor Contact: sbentz@gomperscharter.org

Course Description:
The 6/7 Computer Explorations is a middle school elective course. Exploring Computers 5-8 is a course that meets the high school graduation requirement for Computer Literacy. It addresses the NET Standards (National Education Technology Standards) and the Performance Indicators for the NET Standards. The projects utilize various computer skills and are tied to the California State Standards. Our class meets one 90-minute period two days a week, T/Th or W/F. Every other Monday It meets one 57-minute period. We are committed to helping students learn keyboarding skills, become safe digital citizens, develop computational thinking and explore a wide variety of digital resources while using critical thinking skills to creatively solve problems. Time is provided in class to complete all assignments.

GPA Grading Guidelines:

Category	Grading Criteria	Percentage
Classwork/Participation	• Completion/Quality • Effort/Engagement	30%
Demonstrations of Learning	• Key Course Assignments (See course syllabus for Unit Key Assignments)	35%
Homework/Independent Learning	• Any work assigned to a student in which they complete on their own outside of class. (Must have a minimum of 1 weekly grade)	10%
Quarter Finals	• Quarter finals are course specific, standards based assessments that cover content from the 9 week quarter.	25%

* Classwork/Participation and Homework/Independent Learning will we updated weekly.

Late: When projects/assignments are turned in after the due date, there will be 5% lost each day the assignment is late unless the student has permission from Mr. Bentz to turn the assignment in late.

Absent: When a student is absent they are responsible to check with Mr. Bentz on missing work and class work they may have missed when they were absent. If previous project and assignments were due during their absence, they are due the day the student returns.

Prerequisites: None

Course Materials: Pencil, Pen, Google Classroom, Gompers.TypingClub.com, Code.org, KhanAcademy.org, CS-First.com, ChessKid.com. Student Workbook: K-8 Tech Curriculum Student Workbooks

Participation
Participation includes being on time for class, being prepared for class, taking care of the equipment, keeping a clean workstation area, and keeping a student notebook and student folders with all computer technology assignments.

Course Structure: Will this course be lecture based, interactive, etc

Course of Study:

Digital Tools In Class Unit *1 WEEK - 2 LESSONS*

Content Standards	Learning Objectives	Key Assignments
CCSS: WHST.6-8.7-9 NETS: 3d, 5a, 6a	• Introduce digital tools used in 6th and 7th grade • Acclimate students to the concept that tech tools enable differentiation, collaboration, sharing, and publishing • Show how to employ them in student educational endeavors	• Keyboard Color Match • Hardware-Parts of the Computer • Join GoogleClassroom, ChessKid, KhanAcademy. • Blogging Rules signed

Keyboarding Unit *12WEEKS - 64 LESSONS*

Content Standards	Learning Objectives	Key Assignments
CCSS.ELA-Literacy.W.6.6	• Minimum of three pages in a single sitting (900 words without taking a break) • Typing 25 words a minute	• 15 minutes of typing outside of class 2 times a week. • Multiple typing assessment tests.

		• Typing of 3 pages within 45 minute time frame

Problem Solving, Probability Theory and Planning Ahead Unit *24WEEKS - 64 LESSONS*

Content Standards	Learning Objectives	Key Assignments
CCSS.MATH.PRACTICE.MP1 CCSS.MATH.PRACTICE.MP2 CCSS.MATH.PRACTICE.MP3 CCSS.MATH.PRACTICE.MP4 **CCSS.MATH.PRACTICE.MP5** **CCSS.MATH.PRACTICE.MP6** **CCSS.MATH.PRACTICE.MP7** **CCSS.MATH.PRACTICE.MP8** **CCSS.7.SP.C.7**	• Increase and improve analytical thinking skills • Improve problem solving techniques • Increase self-confidence and improved organizational habits • Improve logic and reasoning skills • Increase patience and persistence • Improve decision making skills	• Chess Exam • Chess Tournaments • 3 Chess Puzzles a day

Coding/Programing Unit *12WEEKS - 64 LESSONS*

Content Standards	Learning Objectives	Key Assignments
UNIT 1 - PROBLEM SOLVING Lesson 1: Intro to Problem Solving CSTA K-12 Computer Science Standards 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.	Code.org: CS-Discoveries Unit 1 Problem Solving Students will be able to: • Communicate and collaborate with classmates in order to solve a problem • Iteratively improve a solution to a problem • Identify different strategies used to solve a problem • Identify the four steps of the problem solving process • Given a problem, identify individual actions that would	• Code.org Projects Unit 1 a. Propose an app Unit 2 b. Personal Website Unit 3 c. Interactive Card d. Design a Game Unit 4 e. Paper Prototype f. App Presentation Unit 5 g. Create a Representation h. Solve a Data Problem Unit 6 i. Board Output j. Sensor Applications

- 1B-AP-13 - Take on varying roles, with teacher guidance, when collaborating with peers during the design, implementation and review stages of program development.

Lesson 2: The Problem Solving Process

CSTA K-12 Computer Science Standards

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.

Lesson 3: Exploring Problem Solving

CSTA K-12 Computer Science Standards

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-13 - Take on

fall within each step of the problem solving process
- Identify useful strategies within each step of the problem solving process
- Apply the problem solving process to approach a variety of problems
- Assess how well-defined a problem is and use strategies to define the problem more precisely
- Identify a computer as a machine that processes information
- Provide a high level description of the different parts of the Input - Output - Store - Process model of a computer
- Identify the inputs and outputs of common computing devices
- Select the inputs and outputs used to perform common computing tasks
- Define processing as the work done (possibly by a computer) to turn an input into an output
- Define an algorithm as the series of commands a computer uses to process information
- Develop and iteratively improve an algorithm for processing information based on given constraints
- Provide examples of common types of

k. Prototype and Innovation

- CS-First Projects
  Music and Sound

- KhanAcademyProjects
  Intro to HTML/CSS
  Intro to JS/ Drawing and Animation

varying roles, with teacher guidance, when collaborating with peers during the design, implementation and review stages of program development.

Lesson 4: What is a Computer?

CSTA K-12 Computer Science Standards

CS - Computing Systems
- 1B-CS-01 - Describe how internal and external parts of computing devices function to form a system.

Lesson 5: Input and Output

CSTA K-12 Computer Science Standards

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.

Lesson 6: Processing

CSTA K-12 Computer Science Standards

AP - Algorithms & Programming
- 2-AP-10 - Use flowcharts and/or pseudocode to address complex problems as algorithms.

CS - Computing Systems
- 1B-CS-01 - Describe how

information that is stored on a computer
- Explain the need for storage as part of processing information with a computer
- Develop an algorithm that incorporates storage considerations
- Describe how information can be processed to solve a particular problem.
- Identify the information an app would need to be provided as input in order to produce a given output
- Identify and define a problem that could be solved using computing
- Design an app that inputs, outputs, stores, and processes information in order to solve a problem
- Provide and incorporate targeted peer feedback to improve a computing artifact

Unit 2: Web development

Students will be able to:

- Identify the reasons someone might visit a given website
- Identify the reasons someone might create a given website
- Identify websites as a form of personal expression
- Explain that HTML allows a programmer to communicate the way content should be structured on a web page

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. Lesson 7: Storage CSTA K-12 Computer Science Standards AP - Algorithms & Programming • 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. Lesson 8: Apps and Problem Solving CSTA K-12 Computer Science Standards AP - Algorithms & Programming • 2-AP-10 - Use flowcharts and/or pseudocode to address complex problems as algorithms. IC - Impacts of Computing • 2-IC-20 - Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options. Lesson 9: Project - Propose an App CSTA K-12 Computer Science	• Write a simple HTML document that uses opening and closing tags to structure content • Understand how to use lesson resources provided in Web Lab • Use heading tags to change the appearance of text on a web page. • Structure content into headings, subheadings, and paragraphs. • Understand and explain reasons that it is difficult to control who sees information published online. • Understand and justify guidelines for safely publishing information online. • Use the , , and tags to create ordered and unordered lists in an HTML page. • Create and name a new HTML page. • Explain the purpose of copyright. • Identify the rights and restrictions granted by various Creative Commons licenses • Add an image to a web page • Describe why using whitespace, indentation, and comments makes your code easier to maintain. • Develop a set of techniques for preventing bugs in HTML	

Standards 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-18 - Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts. CS - Computing Systems ● 2-CS-02 - Design projects that combine hardware and software components to collect and exchange data. UNIT 2 - WEB DEVELOPMENT Lesson 1: Exploring Websites CSTA K-12 Computer Science Standards IC - Impacts of Computing ● 2-IC-20 - Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options. Lesson 2: Websites for Expression CSTA K-12 Computer Science Standards IC - Impacts of Computing ● 1B-IC-18 - Discuss computing technologies	code and finding them when they occur ● Connect multiple web pages into one website using hyperlinks. ● Use CSS selectors to style HTML text elements. ● Create and link to an external style sheet. ● Explain the differences between HTML and CSS in both use and syntax. ● Use CSS properties to change the size, position, and borders of elements. ● Create a CSS rule-set for the body element that impacts all elements on the page. ● Use basic web searching techniques to find relevant information online ● Identify elements that contribute to a website's trustworthiness or untrustworthiness ● Group elements using classes in order to create more specific styles on their website. ● Apply the rgb() color function to add custom colors to their website ● Apply CSS styles across an entire website ● Explain the design choices they made on their website to other people ● Prioritize and implement incremental improvements	

that have changed the world and express how those technologies influence, and are influenced by, cultural practices. Lesson 3: Intro to HTML CSTA K-12 Computer Science Standards AP - Algorithms & Programming • 1B-AP-11 - Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process. Lesson 4: Headings CSTA K-12 Computer Science Standards AP - Algorithms & Programming • 1B-AP-11 - Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process. • 1B-AP-15 - Test and debug (identify and fix errors) a program or algorithm to insure it runs as intended. Lesson 5: Digital Footprint CSTA K-12 Computer Science Standards IC - Impacts of Computing • 2-IC-20 - Compare tradeoffs associated with computing technologies that affect people's	Unit 3 Games Students will be able to: • Identify how Computer Science is used in a field of entertainment • Reason about locations on the Game Lab coordinate grid • Communicate how to draw an image in Game Lab, accounting for shape position, color, and order • Use the Game Lab IDE to plot different colored shapes on the screen. • Sequence code correctly to overlay shapes. • Debug code written by others. • Use and reason about drawing commands with multiple parameters • Generate and use random numbers in a program • Identify a variable as a way to label and reference a value in a program • Use variables in a program to store a piece of information that is used multiple times • Reason about and fix common errors encountered when programming with variables • Assign a sprite to a variable • Use dot notation to update a sprite's properties	

everyday activities and career options.
- 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.

Lesson 6: Lists

CSTA K-12 Computer Science Standards

AP - Algorithms & Programming
- 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 insure it runs as intended.

Lesson 7: Intellectual Property and Images

CSTA K-12 Computer Science Standards

AP - Algorithms & Programming
- 2-AP-16 - Incorporate existing code, media, and libraries into original programs, and give attribution.
- 3A-AP-21 - Evaluate

- Create a static scene combining sprites, shapes, and text
- Explain what an animation is and how it creates the illusion of smooth motion
- Explain how the draw loop allows for the creation of animations in Game Lab
- Use the draw loop in combination with the randomNumber() command, shapes, and sprites to make simple animations
- Describe the connection between updating a sprite's location properties and sprite movement on the screen.
- Read and follow the steps of a short program written in pseudo code that manipulates variable values.
- Use the counter pattern to increment or decrement sprite properties
- Identify which sprite properties need to be change, and in what way, to achieve a specific movement
- Organize objects based on simple and compound boolean statements
- Describe the properties of an object using boolean statements
- Predict the output of simple boolean statements
- Use conditionals to react to changes in variables and sprite properties

licenses that limit or restrict use of computational artifacts when using resources such as libraries. IC - Impacts of Computing ● 1B-IC-21 - Use public domain or creative commons media and refrain from copying or using material created by others without permission. ● 2-IC-23 - Describe tradeoffs between allowing information to be public and keeping information private and secure. Lesson 8: Clean Code and Debugging CSTA K-12 Computer Science Standards AP - Algorithms & Programming ● 2-AP-19 - Document programs in order to make them easier to follow, test, and debug. Lesson 9: Project - Multi-Page Websites CSTA K-12 Computer Science Standards AP - Algorithms & Programming ● 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	● Use conditionals to react to keyboard input ● Move sprites in response to keyboard input ● Use an else statement as the fallback case to an if statement ● Differentiate between conditions that are true once per interaction, and those that remain true through the duration of an interaction. ● Use conditionals to react to keyboard input or changes in variables / properties ● Sequence commands to draw in the proper order ● Apply an iterator pattern to variables or properties in a loop ● Use the velocity and rotation Speed blocks to create and change sprite movements ● Describe the advantages of simplifying code by using higher level blocks ● Use the isTouching block to determine when two sprites are touching ● Describe how abstractions help to manage the complexity of code ● Use sprite velocity with the counter pattern to create different types of sprite movement ● Explain how individual programming constructs can be combined to create more complex behavior	

attribution. • 2-AP-17 - Systematically test and refine programs using a range of test cases. • 2-AP-19 - Document programs in order to make them easier to follow, test, and debug. IC - Impacts of Computing • 1B-IC-21 - Use public domain or creative commons media and refrain from copying or using material created by others without permission. Lesson 10: Styling Text with CSS CSTA K-12 Computer Science Standards AP - Algorithms & Programming • 2-AP-16 - Incorporate existing code, media, and libraries into original programs, and give attribution. • 2-AP-19 - Document programs in order to make them easier to follow, test, and debug. Lesson 11: Styling Elements with CSS CSTA K-12 Computer Science Standards AP - Algorithms & Programming • 2-AP-16 - Incorporate existing code, media, and libraries into original programs, and give attribution. • 2-AP-19 - Document	• Use the `displace`, `collide`, `bounce`, and `bounceOff` blocks to produce sprite interactions • Describe how abstractions can be built upon to develop even further abstractions • Create and use functions for blocks of code that perform a single high-level task within a program • Create and use functions to remove repeated blocks of code from their programs • Create and use functions to improve the readability of their programs • Explain how abstractions allow programmers to reason about a program at a higher level • Identify core programming constructs necessary to build different components of a game • Create and use multi frame animations in a program • Implement different features of a program by following a structured project guide • identify core programming constructs necessary to build different components of a game • Implement different features of a program by following a structured project guide • Independently scope the features of a piece of software	

programs in order to make them easier to follow, test, and debug.

Lesson 12: Sources and Search Engines

CSTA K-12 Computer Science Standards

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-23 - Describe tradeoffs between allowing information to be public and keeping information private and secure.

Lesson 13: RGB Colors and Classes

CSTA K-12 Computer Science Standards

AP - Algorithms & Programming
- 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-19 - Document programs in order to make them easier to follow, test, and debug.

- Create a plan for building a piece of software by describing its major components
- Implement a plan for creating a piece of software

Unit 4 - The Design Process

Students will be able to:

- Express opinions respectfully and effectively
- Critically evaluate an object for how well its design meets a given set of needs
- Identify empathy for the user as an important component of the design process
- Distinguish between their own needs and the needs of their users
- Critique a design through the perspective of a user profile
- Design improvements to a product based on a user profile (not personal opinions)
- Empathize with a user's needs to design an object
- Create meaningful categories from a collection of ideas, specifically in the context of a brainstorm
- Use a paper prototype to test out an app before programming it
- Identify the user needs a prototype was designed to address
- Translate user needs into changes and improvements in the user interface of an app
- Categorize and prioritize user feedback for an app
- Create a paper prototype for the screens of an app

Lesson 14: Project - Personal Portfolio Website CSTA K-12 Computer Science Standards AP - Algorithms & Programming • 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. UNIT 3 - INTERACTIVE GAMES AND ANIMATIONS Lesson 1: Programming for Entertainment CSTA K-12 Computer Science Standards IC - Impacts of Computing • 2-IC-21 - Discuss issues of bias and accessibility in the design of existing technologies. Lesson 2: Plotting Shapes	• Interview a peer to learn about their needs • Analyze interview notes to develop follow-up questions • Brainstorm potential solutions to a specific problem • 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 • Identify ways in which apps can effect social change • Identify ways in which apps can effect social change • Document user interactions with a prototype of an app's User Interface using paper and pen • Be able to articulate and demonstrate the user flow through an app's design • Test a prototype with a user, recording the results • Analysing a user test to identify potential issues or improvements • Translate a paper prototype into a digital format • Select the appropriate input element for a given type of information • Write programs that respond to user input • Describe the difference between event driven programming and the draw loop model • Collaborate with others to develop an interactive prototype • Apply the event driven programming model to new events	

CSTA K-12 Computer Science Standards

AP - Algorithms & Programming
- 2-AP-10 - Use flowcharts and/or pseudocode to address complex problems as algorithms.
- 2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.
- 2-AP-17 - Systematically test and refine programs using a range of test cases.

Lesson 3: Drawing in Game Lab

CSTA K-12 Computer Science Standards

AP - Algorithms & Programming
- 2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.
- 2-AP-19 - Document programs in order to make them easier to follow, test, and debug.

Lesson 4: Shapes and Randomization

CSTA K-12 Computer Science Standards

AP - Algorithms & Programming
- 2-AP-13 - Decompose problems and

- Use code to modify statically created design elements
- Write programs that take user input in the form of strings and display that information elsewhere
- Write out a detailed plan for how they will test their low fidelity prototype with other people
- Run a user test on an app and record what users say about their minimum viable product
- Analyze the user feedback from the previous lesson and determine a list of bugs (flaws) that need to be fixed and features that could be added to the app
- Prioritize the bugs and features according to impact and ease of implementation
- Present technical information clearly to non-technical users
- Reflect on the development of an ongoing project

Unit 5 - Data and Society

Students will be able to:

- Define data as information collected from the world to help make a recommendation or solve a problem.
- Provide examples of how representing data in different ways can affect its ability to solve different problems.
- Choose the best way to represent some information based on how it will be used.
- Define data as information collected from the world to help make a

subproblems into parts to facilitate the design, implementation, and review of programs. • 2-AP-17 - Systematically test and refine programs using a range of test cases. • 2-AP-19 - Document programs in order to make them easier to follow, test, and debug. Lesson 5: Variables CSTA K-12 Computer Science Standards AP - Algorithms & Programming • 2-AP-11 - Create clearly named variables that represent different data types and perform operations on their values. • 2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs. • 2-AP-17 - Systematically test and refine programs using a range of test cases. • 2-AP-19 - Document programs in order to make them easier to follow, test, and debug. Lesson 6: Sprites CSTA K-12 Computer Science Standards AP - Algorithms & Programming • 2-AP-11 - Create clearly	recommendation or solve a problem. • Provide examples of how representing data in different ways can affect its ability to solve different problems. • Choose the best way to represent some information based on how it will be used. • Define data as information collected from the world to help make a recommendation or solve a problem. • Provide examples of how representing data in different ways can affect its ability to solve different problems. • Choose the best way to represent some information based on how it will be used. • Create and manipulate binary patterns to represent black and white images • Describe common features of systems used to represent information in binary • Use a binary system to represent numbers. • Extend a representation system based on patterns. • Structure binary information into a fixed number of bits. • Convert between binary and decimal numbers up to 255 • Explain why computers standardize the lengths of bit patterns. • Use multiple binary systems to decode information. • Determine the most appropriate encoding system for a given piece of information. • Choose and justify the use of different binary	

named variables that represent different data types and perform operations on their values.
- 2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.
- 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-19 - Document programs in order to make them easier to follow, test, and debug.

Lesson 7: The Draw Loop

CSTA K-12 Computer Science Standards

AP - Algorithms & Programming
- 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,

representation systems depending on the information being represented
- Encode and decode information represented in binary numbers and ASCII text
- Create a generalized representation system for many instances of a complex type of information
- Use the problem solving process to answer a question using data
- Use tables and visualizations summarizing data to support a decision
- Present and critique interpretations of tables and visualizations
- Identify additional data that could be collected to improve a decision
- Identify and remove irrelevant data from a data set.
- Create a bar chart based on a set of data.
- Explain why a set of data must be cleaned before a computer can use it.
- Design an algorithm for making decisions using data as inputs
- Explain the benefits and drawbacks of using computers for automated decision making
- Interpret collected data to identify patterns
- Give examples of how data is collected from sensors and tracking user behavior.
- Determine data that would be helpful in solving a problem, and how that data could be collected.

implementation, and review of programs.
- 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-19 - Document programs in order to make them easier to follow, test, and debug.

Lesson 8: Counter Pattern Unplugged

CSTA K-12 Computer Science Standards

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.

Lesson 9: Sprite Movement

CSTA K-12 Computer Science Standards

AP - Algorithms & Programming
- 2-AP-11 - Create clearly named variables that represent different data types and perform operations on their values.
- 2-AP-12 - Design and

- Distinguish between data that users intentionally and unintentionally produce.
- Apply the data problem solving process to a personally relevant topic
- Determine appropriate sources of data needed to solve a problem

Unit 6 - Physical Computing

Students should be able to:

- Identify computing innovations within a given field
- For a given device, articulate the likely inputs and outputs
- Suggest improvements to help a device better solve a specific problem
- Compare and contrast multiple ways to take input
- Describe the elements of an event handler
- Model different methods of taking user input
- Create a user interface composed of multiple design elements
- Use event handlers to respond to user input
- Explain the difference between the "click", "change", and "input" events, and identify scenarios for each
- Use a getter to get the current content of a UI element
- Use a setter to change the content or properties of a UI element
- Write programs that change multiple elements on a single screen instead of changing screens

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-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-19 - Document programs in order to make them easier to follow, test, and debug. Lesson 10: Booleans Unplugged CSTA K-12 Computer Science Standards AP - Algorithms & Programming • 2-AP-10 - Use flowcharts and/or pseudocode to address complex problems as algorithms. Lesson 11: Booleans and Conditionals CSTA K-12 Computer Science Standards AP - Algorithms & Programming • 2-AP-12 - Design and iteratively develop	• Connect and troubleshoot external devices • Turn on and off an LED with code • Access a specific location in a list using its index • Articulate the difference between the length of a list and the index of its last value • Access items in an array by index • Apply the RGB color model to LEDs • Understand how to use a for loop as a way to repeat a set of code a certain number of times • Trace the execution of a for loop • Given a for loop, predict how many times it will repeat • Use for loops to process through the colorLeds and do something to all the colorLeds • Use for loops to process a list • Use a timed loop to write a non-blocking infinite loop • Replicated a for loop with a timed loop • Use event handlers to take user input • Output simple information on a physical device • Use a loop to repeat instructions • Prototype a program that integrates software and hardware • Attach an event handler to a hardware input • Choose the appropriate event for a given scenario • Develop programs that respond to analog input • Scale a range of numbers to meet a specific need	

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-17 - Systematically test and refine programs using a range of test cases. • 2-AP-19 - Document programs in order to make them easier to follow, test, and debug. Lesson 12: Conditionals and User Input CSTA K-12 Computer Science Standards AP - Algorithms & Programming • 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-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	• Represent a sensor value in a variety of ways • Develop apps that take input through analog sensors • Independently scope the features of a piece of software • Prototype a physical computing device • Implement a plan for developing a piece of software that integrates hardware inputs and outputs	

cases. ● 2-AP-19 - Document programs in order to make them easier to follow, test, and debug. Lesson 13: Other Forms of Input CSTA K-12 Computer Science Standards AP - Algorithms & Programming ● 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-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-19 - Document programs in order to make them easier to follow, test, and debug. Lesson 14: Project -		

Interactive Card CSTA K-12 Computer Science Standards AP - Algorithms & Programming • 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.

Lesson 15: Velocity

CSTA K-12 Computer Science Standards

AP - Algorithms & Programming

- 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-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-19 - Document programs in order to make them easier to follow, test, and debug.

Lesson 16: Collision Detection

CSTA K-12 Computer Science Standards

AP - Algorithms &

Programming • 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-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. Lesson 17: Complex Sprite Movement CSTA K-12 Computer Science Standards AP - Algorithms & Programming • 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-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-19 - Document programs in order to make them easier to follow, test, and debug.

Lesson 18: Collisions

CSTA K-12 Computer Science Standards

AP - Algorithms & Programming
- 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-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-19 - Document programs in order to make them easier to follow, test, and debug.

Lesson 19: Functions

CSTA K-12 Computer Science Standards

AP - Algorithms & Programming
- 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-14 - Create procedures with parameters to organize code and make it easier to reuse.
- 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-19 - Document programs in order to make them easier to follow, test, and debug. Lesson 20: The Game Design Process CSTA K-12 Computer Science Standards AP - Algorithms & Programming • 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-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-19 - Document programs in order to make them easier to		

follow, test, and debug.

Lesson 21: Using the Game Design Process

CSTA K-12 Computer Science Standards

AP - Algorithms & Programming

- 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-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-19 - Document programs in order to make them easier to follow, test, and debug.

Lesson 22: Project - Design a Game

CSTA K-12 Computer Science Standards

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.

UNIT 4 - THE DESIGN PROCESS

Lesson 1: Analysis of Design

CSTA K-12 Computer Science Standards

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.

Lesson 2: Understanding Your User

CSTA K-12 Computer Science Standards

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.

Lesson 3: User-Centered Design Micro Activity

CSTA K-12 Computer Science Standards

CS - Computing Systems
- 2-CS-02 - Design projects that combine hardware and software components to collect and exchange data.

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.

Lesson 4: User Interfaces

CSTA K-12 Computer Science Standards

AP - Algorithms & Programming
- 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.

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.

Lesson 5: Feedback and Testing

CSTA K-12 Computer Science Standards

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-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. Lesson 6: Identifying User Needs CSTA K-12 Computer Science Standards 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-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. Lesson 7: Project - Paper		

Prototype CSTA K-12 Computer Science Standards 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. Lesson 8: Designing Apps for Good CSTA K-12 Computer Science Standards 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. Lesson 9: Market Research CSTA K-12 Computer Science Standards 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.

Lesson 10: Paper Prototypes

CSTA K-12 Computer Science Standards

AP - Algorithms & Programming
- 2-AP-10 - Use flowcharts and/or pseudocode to address complex problems as algorithms.
- 2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.
- 2-AP-19 - Document programs in order to make them easier to follow, test, and debug.

Lesson 11: Prototype Testing

CSTA K-12 Computer Science Standards

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.

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. DA - Data & Analysis • 2-DA-08 - Collect data using computational tools and transform the data to make it more useful and reliable. • 2-DA-09 - Refine computational models based on the data they have generated. IC - Impacts of Computing • 2-IC-22 - Collaborate with many contributors through strategies such as crowdsourcing or surveys when creating a computational artifact. Lesson 12: Digital Design CSTA K-12 Computer Science Standards AP - Algorithms & Programming • 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-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. 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. IC - Impacts of Computing • 2-IC-22 - Collaborate with many contributors through strategies such as crowdsourcing or surveys when creating a computational artifact. Lesson 13: Event Driven Programming CSTA K-12 Computer Science Standards AP - Algorithms & Programming • 2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs. • 2-AP-14 - Create procedures with parameters to organize code and make it easier to reuse. • 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-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-22 - Collaborate with many contributors through strategies such as crowdsourcing or surveys when creating a computational artifact. Lesson 14: Basic App Functionality CSTA K-12 Computer Science Standards AP - Algorithms & Programming • 2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs. • 2-AP-14 - Create procedures with parameters to organize code and make it easier to reuse. • 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-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-22 - Collaborate with many contributors through strategies such as crowdsourcing or surveys when creating a computational artifact. Lesson 15: Testing the App CSTA K-12 Computer Science Standards AP - Algorithms & Programming ● 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. ● 2-AP-18 - Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts. 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.

DA - Data & Analysis
- 2-DA-08 - Collect data using computational tools and transform the data to make it more useful and reliable.

IC - Impacts of Computing
- 2-IC-22 - Collaborate with many contributors through strategies such as crowdsourcing or surveys when creating a computational artifact.

Lesson 16: Improving and Iterating

CSTA K-12 Computer Science Standards

AP - Algorithms & Programming
- 2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.
- 2-AP-14 - Create procedures with parameters to organize code and make it easier to reuse.
- 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. 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. DA - Data & Analysis • 2-DA-09 - Refine computational models based on the data they have generated. IC - Impacts of Computing • 2-IC-22 - Collaborate with many contributors through strategies such as crowdsourcing or surveys when creating a computational artifact. Lesson 17: Project - App Presentation CSTA K-12 Computer Science Standards AP - Algorithms & Programming • 2-AP-19 - Document programs in order to make them easier to follow, test, and debug. 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.

DA - Data & Analysis
- 2-DA-09 - Refine computational models based on the data they have generated.

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.

UNIT 5 - DATA AND SOCIETY

Lesson 1: Representation Matters

CSTA K-12 Computer Science Standards

DA - Data & Analysis
- 2-DA-07 - Represent data using multiple encoding schemes.

Lesson 2: Patterns and Representation

CSTA K-12 Computer Science Standards

DA - Data & Analysis
- 2-DA-07 - Represent data using multiple encoding schemes.

Lesson 3: ASCII and Binary Representation CSTA K-12 Computer Science Standards DA - Data & Analysis ● 2-DA-07 - Represent data using multiple encoding schemes. Lesson 4: Representing Images CSTA K-12 Computer Science Standards DA - Data & Analysis ● 2-DA-07 - Represent data using multiple encoding schemes. Lesson 5: Representing Numbers CSTA K-12 Computer Science Standards DA - Data & Analysis ● 2-DA-07 - Represent data using multiple encoding schemes. Lesson 6: Eight Bit Numbers CSTA K-12 Computer Science Standards DA - Data & Analysis ● 2-DA-07 - Represent data using multiple encoding schemes. Lesson 7: Combining Representations Lesson 8: Create a Representation CSTA K-12 Computer Science		

Standards AP - Algorithms & Programming ● 2-AP-10 - Use flowcharts and/or pseudocode to address complex problems as algorithms. ● 2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs. DA - Data & Analysis ● 2-DA-07 - Represent data using multiple encoding schemes. Lesson 9: Problem Solving and Data CSTA K-12 Computer Science Standards DA - Data & Analysis ● 2-DA-08 - Collect data using computational tools and transform the data to make it more useful and reliable. Lesson 10: Making Decisions with Data CSTA K-12 Computer Science Standards DA - Data & Analysis ● 2-DA-08 - Collect data using computational tools and transform the data to make it more useful and reliable. Lesson 11: Interpreting Data CSTA K-12 Computer Science		

Standards DA - Data & Analysis ● 2-DA-08 - Collect data using computational tools and transform the data to make it more useful and reliable. Lesson 12: Automating Data Decisions CSTA K-12 Computer Science Standards AP - Algorithms & Programming ● 2-AP-10 - Use flowcharts and/or pseudocode to address complex problems as algorithms. DA - Data & Analysis ● 2-DA-08 - Collect data using computational tools and transform the data to make it more useful and reliable. IC - Impacts of Computing ● 2-IC-22 - Collaborate with many contributors through strategies such as crowdsourcing or surveys when creating a computational artifact. Lesson 13: Problem Solving with Big Data CSTA K-12 Computer Science Standards IC - Impacts of Computing ● 2-IC-20 - Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options.		

- 2-IC-23 - Describe tradeoffs between allowing information to be public and keeping information private and secure.

Lesson 14: Project - Solve a Data Problem

CSTA K-12 Computer Science Standards

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-18 - Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts.

DA - Data & Analysis
- 2-DA-08 - Collect data using computational tools and transform the data to make it more useful and reliable.

IC - Impacts of Computing
- 2-IC-23 - Describe tradeoffs between allowing information to be public and keeping information private and secure.

UNIT 6 - PHYSICAL COMPUTING

Lesson 1: Innovations in

Computing CSTA K-12 Computer Science Standards IC - Impacts of Computing ● 2-IC-20 - Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options. Lesson 2: Input Unplugged CSTA K-12 Computer Science Standards AP - Algorithms & Programming ● 2-AP-10 - Use flowcharts and/or pseudocode to address complex problems as algorithms. Lesson 3: Event Types CSTA K-12 Computer Science Standards AP - Algorithms & Programming ● 2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs. ● 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. Lesson 4: Getters and Setters		

CSTA K-12 Computer Science Standards AP - Algorithms & Programming • 2-AP-11 - Create clearly named variables that represent different data types and perform operations on their values. • 2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs. • 2-AP-16 - Incorporate existing code, media, and libraries into original programs, and give attribution. • 2-AP-19 - Document programs in order to make them easier to follow, test, and debug. Lesson 5: The Circuit Playground CSTA K-12 Computer Science Standards AP - Algorithms & Programming • 2-AP-11 - Create clearly named variables that represent different data types and perform operations on their values. • 2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs. • 2-AP-16 - Incorporate existing code, media, and		

libraries into original programs, and give attribution. ● 2-AP-19 - Document programs in order to make them easier to follow, test, and debug. CS - Computing Systems ● 2-CS-02 - Design projects that combine hardware and software components to collect and exchange data. ● 2-CS-03 - Systematically identify and fix problems with computing devices and their components. Lesson 6: Lists CSTA K-12 Computer Science Standards AP - Algorithms & Programming ● 2-AP-11 - Create clearly named variables that represent different data types and perform operations on their values. ● 2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs. ● 2-AP-16 - Incorporate existing code, media, and libraries into original programs, and give attribution. ● 2-AP-19 - Document programs in order to make them easier to follow, test, and debug. CS - Computing Systems		

- 2-CS-02 - Design projects that combine hardware and software components to collect and exchange data.

Lesson 7: Color LEDs

CSTA K-12 Computer Science Standards

AP - Algorithms & Programming
- 2-AP-11 - Create clearly named variables that represent different data types and perform operations on their values.
- 2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.
- 2-AP-16 - Incorporate existing code, media, and libraries into original programs, and give attribution.
- 2-AP-19 - Document programs in order to make them easier to follow, test, and debug.

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.
- 2-CS-03 - Systematically identify and fix problems

with computing devices and their components.

Lesson 8: For Loops

CSTA K-12 Computer Science Standards

AP - Algorithms & Programming

- 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-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-19 - Document programs in order to make them easier to follow, test, and debug.

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.
- 2-CS-03 - Systematically identify and fix problems with computing devices and their components.

Lesson 9: Lists and For Loops

CSTA K-12 Computer Science Standards

AP - Algorithms & Programming
- 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-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-19 - Document programs in order to make them easier to

follow, test, and debug.

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.
- 2-CS-03 - Systematically identify and fix problems with computing devices and their components.

Lesson 10: Timed Loops

CSTA K-12 Computer Science Standards

AP - Algorithms & Programming

- 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-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-19 - Document programs in order to make them easier to follow, test, and debug. 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. ● 2-CS-03 - Systematically identify and fix problems with computing devices and their components. Lesson 11: Project - Board Output CSTA K-12 Computer Science Standards 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. 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. • 2-CS-03 - Systematically identify and fix problems with computing devices		

and their components.

Lesson 12: Physical Input

CSTA K-12 Computer Science Standards

AP - Algorithms & Programming

- 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-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-19 - Document programs in order to make them easier to follow, test, and debug.

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. • 2-CS-03 - Systematically identify and fix problems with computing devices and their components. Lesson 13: Analog Input CSTA K-12 Computer Science Standards AP - Algorithms & Programming • 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-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-19 - Document programs in order to make them easier to		

follow, test, and debug.

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.
- 2-CS-03 - Systematically identify and fix problems with computing devices and their components.

Lesson 14: Sensor Applications

CSTA K-12 Computer Science Standards

AP - Algorithms & Programming

- 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-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-19 - Document programs in order to make them easier to follow, test, and debug. 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. ● 2-CS-03 - Systematically identify and fix problems with computing devices and their components. Lesson 15: Project - Prototype an Innovation CSTA K-12 Computer Science Standards 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. 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. • 2-CS-03 - Systematically identify and fix problems		

with computing devices and their components. CCSS.ELA-Literacy.RST.6-8.3, CCSS.ELA-Literacy.RST.6-8.4, CCSS.ELA-Literacy.RST.6-8.7, CCSS.ELA-Literacy.RST.9-10.3, CCSS.ELA-Literacy.RST.9-10.4, CCSS.ELA-Literacy.RST.9-10.5, CCSS.ELA-Literacy.RST.9-10.7, CCSS.Math.Content.5.OA.B.3, CCSS.Math.Content.6.NS.C.6, CCSS.Math.Content.7.RP.A.2b, CCSS.Math.Content.8.F.A.1, CCSS.Math.Content.HSN-Q.A.2 CCSS.ELA-Literacy.RI.5.10, CCSS.ELA-Literacy.RI.5.3, CCSS.ELA-Literacy.RST.11-12.3, CCSS.ELA-Literacy.RST.11-12.4, CCSS.ELA-Literacy.RST.11-12.7, CCSS.ELA-Literacy.RST.11-12.9		

Course Specific Student Expectations:

Computer and internet access after school hours.

Working Together:

Students are encouraged to discuss assignments with fellow classmates, but each student is responsible for writing an individual answer and/or his/her own projects and assignments. Cheating is: sharing written or electronic work either by copying, retyping, looking at, or supplying a copy. Cheating is not: discussing concepts, answering questions about concepts or clarifying ambiguities, or helping someone understand how to use the class tools and software.

Accommodations/Modification and Supports:

Any student who requires accommodations, modifications or additional supports should contact me as early as possible so that we may arrange accommodations, modifications and supports.

GPA Student Expectations:

School-wide Attendance: All students are expected to be punctual and in their classroom seat, ready to learn for each day. Under California law (Ed. Code 48200) all children between the ages of six and eighteen are required to be enrolled and in regular attendance at school. GPA families know that school attendance is the critical first step to make sure that each student receives an education that will help them on their path to college. Students cannot learn what they need to be prepared for the next grade level, if they are not in school. The more absences from school a student has, the more they fall behind in their classes and the more difficult it will be to make it to college.

Planner Use: All students are expected to write all assignments in their GPA planner daily. Your first GPA planner will be provided by the school to support organization and time management.

Homework Completion: As a school working toward college preparation, all GPA students are expected to complete their daily/weekly assignments. Students who fail to complete their homework assignments on time, and are unexcused, will be required to attend lunch and after school tutoring support daily until completed. Until all assignments are completed, students may not be eligible for athletics, clubs, and other extracurricular activities.

Electronic Device Policy: Cell phones, smart watches, and other electronic communication devices that can send and/or receive data are not permitted to be visible, heard, or used in any manner during school hours except by approval of school authorities. Any violation and/or disruption of the learning process will result in the confiscation of the item. The parent/guardian must pick up the confiscated item from the Office of Student Conduct or the teacher.

Computer/Internet Usage Policy: Students may not use computers and/or the GPA network without proper adult supervision. The teacher/staff will choose resources on the Internet that are appropriate for classroom instruction and/or research for the needs, maturity, and ability of their students.

Acceptable Use-

- Access to any site that provides information relevant to current class assignments
- Access to college or university websites
- Use of teacher approved educational software (games, instructional tools, etc.)

Academic Integrity: Honest behavior is an expectation for all students at Gompers Preparatory Academy. Our goal is to create and maintain an ethical academic atmosphere. Acts of academic dishonesty that will not be tolerated at GPA are listed below:

- Cheating on any classroom assignment, test, or quiz
- Plagiarism - copying or representing another's ideas, words, or work as one's own, without properly citing the source. Plagiarism includes the misuse of published

material, electronic material, and/or the work of other students. The original writer who intentionally shares his/her work for another to copy, without the permission of the teacher, is also engaged in plagiarism.

- Fabrication (any falsification or invention of date, citation, or other authority in an assignment); theft or alteration of materials
- Unauthorized collaboration
- Unauthorized use of electronic devices

Students found in violation of GPA's Academic Integrity Policy will be disciplined appropriately which may lead to formal suspension. Consequences for offenses may include, but are not limited to, detention, *lowering of academic and citizenship grade and/or suspensions/exclusion from extracurricular activities*.

Standards/Format for Writing Papers - MLA Format:

The standard format for all papers follows the MLA formatting rules:

1. Typed, double-spaced: TIMES NEW ROMAN, 12 font, including title
2. Heading: 4 lines

Student name:	"Sammy Gompers"
Teacher name:	Ms. Teacher
Course name, period:	English I, Period 3
Date	06 February 2009

3. All pages numbered: upper right corner, last name and page number; no punctuation, no "p." or "pg."
4. Title: centered, upper and lower case
5. Work Cited/ Documentation Format: It is necessary to credit any source that is used in a paper or project. Plagiarism is considered cheating. All sources must be documented. Citing sources in a paper must be thorough and accurate. MLA formatting for in text citations and works cited is mandatory

Important Dates:

Quarter 1:

- Finals Week: October 23rd and 27th
- Parent Conferences: October 23rd - 27th
- End Date: October 30th

Quarter 2:

- Q2 Finals Week: January 22nd - 26th
- Parent Conferences: January 16th - 22nd
- End Date: January 31st

Quarter 3:

- Q3 Finals Week: April 9th - 13th
- Parent Conferences: April 16th - 20th
- End Date: April 23rd

Quarter 4:

- Q4 Finals Week: May 29th - June 1st
- End Date: June 26th

Student Signature : _____ Parent/Guardian Signature: _____

Date: _____