COURSE: CS DISCOVERIES PHASE 2

Susie McCatty FALL 2018

Phase I: Course Essential Question

How can computer science help solve real world problems?

Phase II Curriculum

Unit: 1 PROBLEM SOLVING

CHAPTER 1 The Problem Solving Process

Essential Questions:

- What strategies and processes can I use to become a more effective problem solver?
- What is the problem solving process?
- What actions can I take to solve problems?

Essential Understanding:

- Following a problem solving model will aide in successful outcomes.
- A model is a tool to help reach desired outcomes.
- Define, prepare, try, & reflect are key parts to the problem solving model.

Curriculum Standards- DOK noted where applicable with 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-16 - Take on varying roles, with teacher guidance, when collaborating with peers during the design, implementation and review stages of program development.

Knowledge/Content	Skills/Processes
I Know ...(includes academic vocabulary)	I Can ...
● Define ● Prepare ● Try ● Reflect	● 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 fall within each step of the problem solving process ● Identify useful strategies within each step of the problem solving process ● Identify the four steps of the problem solving process ● Given a problem, identify individual actions that would fall within each step of the problem solving process ● Identify useful strategies within each step of the problem solving process

Phase II Curriculum

Unit: 1 PROBLEM SOLVING

CHAPTER 2 Computers and Problem Solving

Essential Questions:

- 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

Essential Understanding:

- Computers make tasks easier, faster, and solve problems.
- Computers can systematically run with algorithms
- Computers require input to process and output
- Humans must develop an algorithm whereas a computer runs the algorithm.
- Processing components must be set by humans for algorithms to be effective.

Curriculum Standards- DOK noted where applicable with 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.

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.

IC - Impacts of Computing

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

Knowledge/Content I Know ...(includes academic vocabulary)	Skills/Processes I Can ...

• Input • Output • Store • Process • Algorithm	• 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 • 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 • Describe how information can be processed to solve a particular problem. • Identify a possible source of a given input. • Determine what information should be stored on a device for later. • 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

Phase II Curriculum

Unit: 2 WEB DEVELOPMENT

CHAPTER 1 Web Content and HTML

Essential Questions:

- Why do people create websites?
- How can text communicate content and structure on a web page?
- How can I incorporate content I find online into my own webpage?
- What strategies can I use when coding to find and fix issues?

Essential Understanding:

- Websites are created for a variety of purposes including self expressions,
- HTML communicates text structure on a webpage
- Use of style conventions makes reading/writing HTML easier.
- Problems/subproblems can be decomposed by checking the algorithm HTML with use of codes.
- Use of copyrighted materials require permissions when adding content to web pages.

Curriculum Standards- DOK noted where applicable with Standards

C - 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.

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

- 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.

NI - Networks & the Internet

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

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-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.

- 3A-AP-20 - Evaluate licenses that limit or restrict use of computational artifacts when using resources such as libraries

Knowledge/Content I Know …(includes academic vocabulary)	Skills/Processes I Can …
● Website ● Website Content ● HTML ● HTML Element ● HTML Tag ● Website Structure ● Heading ● Digital Footprint ● Citation ● Copyright ● Creative Commons ● Intellectual Property ● Bug ● Comment ● Debugging ● Indentation ● Whitespace	● 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 ● 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 <ol>, <ul>, and <li> 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 code and finding them when they occur • Connect multiple web pages into one website using hyperlinks.

Phase II Curriculum
Unit: 2 WEB DEVELOPMENT
CHAPTER 2 Styling and CSS

Essential Questions:	Essential Understanding:
• 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?	• CSS add style to HTML text elements. • A variety of elements contribute to a website's trustworthiness.

Curriculum Standards- DOK noted where applicable with 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.

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.

Knowledge/Content I Know ...(includes academic vocabulary)	Skills/Processes I Can ...
• CSS • CSS Selector • CSS Class • Relevant • Search Engine	• 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

Phase II Curriculum

Unit: 3 INTERACTIVE GAMES AND ANIMATIONS

CHAPTER 1 Interactive Games and Animations

Essential Questions:	Essential Understanding:
● What is a computer program? ● What are the core features of most programming languages? ● How does programming enable creativity and individual expression? ● What practices and strategies will help me as I write programs?	● A program is a collection of instructions that performs a specific task when executed by a computer. ● Variables will help group/store pieces of information used multiple times. ● Programming is very personal and allows for self expression in a variety of formats. ● Iterator patterns, counting, sequence commands, and variables all contribute to programming

language.

Curriculum Standards- DOK noted where applicable with Standards

IC - Impacts of Computing

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

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-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.

Knowledge/Content I Know ...(includes academic vocabulary)	Skills/Processes I Can ...
● Bug ● Debugging ● Program	● 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,

• Parameter • Variable • Property • Sprite • Animation • Frame • Frame rate • Expression • Variable • Boolean • Conditionals • Expression • Boolean expression • If-statement	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 • 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

	pseudocode that manipulates variable values. ● Use the counter pattern to increment or decrement sprite properties ● Identify which sprite properties need to be changed, 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 ● 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

Phase II Curriculum

Unit: 3 INTERACTIVE GAMES AND ANIMATIONS

CHAPTER 2 Building Games

Essential Questions:

- How do software developers manage complexity and scale?
- How can programs be organized so that common problems only need to be solved once?
- How can I build on previous solutions to create even more complex behavior?

Essential Understanding:

- By reducing the amount of code, complexity and scale is more easily managed.
- Use of functions help to organize code.
- Procedures are established to organize code.

Curriculum Standards- DOK noted where applicable with 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-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-19 - Document programs in order to make them easier to follow, test, and debug

Knowledge/Content I Know ...(includes academic vocabulary)	Skills/Processes I Can ...

● Abstraction ● Function	● Use the velocity and rotationSpeed 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 ● 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 multiframe 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 ● Create a plan for building a piece of software by describing its major components ● Implement a plan for creating a piece of software