WDHS Curriculum Map
Course: Introduction to Computer Science 2

Time Interval/ Content	Standards/ Strands	Essential Questions	Skills	Assessment
Unit 1: Introduction to Processing 2 weeks *Learning Processing* Lesson 1 (Chapters 1-2)	**MA.K-12.CCSS.Math.Practice.MP1** - [*Standard*] - Make sense of problems and persevere in solving them. **MA.K-12.CCSS.Math.Practice.MP2** - [*Standard*] - Reason abstractly and quantitatively. **MA.K-12.CCSS.Math.Practice.MP3** - [*Standard*] - Construct viable arguments and critique the reasoning of others. **MA.K-12.CCSS.Math.Practice.MP4** - [*Standard*] - Model with mathematics. **MA.K-12.CCSS.Math.Practice.MP5** - [*Standard*] - Use appropriate tools strategically. **MA.K-12.CCSS.Math.Practice.MP6** - [*Standard*] - Attend to precision. **MA.K-12.CCSS.Math.Practice.MP7** - [*Standard*] - Look for and make use of structure. **MA.K-12.CCSS.Math.Practice.MP8** - [*Standard*] - Look for and express regularity in repeated reasoning. **TECH.8.1.12.A.1** - [*Cumulative Progress Indicator*] - Create a personal digital portfolio which reflects personal and academic interests, achievements, and career aspirations by using a variety of digital tools and resources. **TECH.8.1.12.A.3** - [*Cumulative Progress Indicator*] - Collaborate in online courses, learning communities, social networks or virtual worlds to discuss a resolution to a problem or issue.	What are the strengths and weaknesses of Processing as a programming language? What is the value of writing elegant code with consistent indentation?	Students will understand that… ● Processing is a programming language similar to Java which emphasizes the importance of graphics and design. ● Each programming language has its strengths and limitations. Students will know… ● The basics of the Processing development environment. ● Basic function calls, assignment operations, and control structures in Processing. ● Proper indentation and structure in code. Students will be able to… ● Write and publish a basic program using the Processing language. ● Make a drawing dynamic by having it interact with an input method (mouse or keyboard).	● Homework ● Classwork ● Quiz ● Test Unit 1 Project 1. Create a static screen drawing using geometric figures. 2. Make the drawing dynamic by adding interaction with the mouse.

	TECH.8.1.12.D.1 - [*Cumulative Progress Indicator*] - Demonstrate appropriate application of copyright, fair use and/or Creative Commons to an original work. **TECH.8.1.12.D.5** - [*Cumulative Progress Indicator*] - Analyze the capabilities and limitations of current and emerging technology resources and assess their potential to address personal, social, lifelong learning, and career needs. **TECH.8.1.12.E.CS3** - [*Content Statement*] - Evaluate and select information sources and digital tools based on the appropriateness for specific tasks. **TECH.8.1.12.F.CS2** - [*Content Statement*] - Plan and manage activities to develop a solution or complete a project. **TECH.8.2.12.E.1** - [*Cumulative Progress Indicator*] - Demonstrate an understanding of the problem-solving capacity of computers in our world. **TECH.8.2.12.E.3** - [*Cumulative Progress Indicator*] - Use a programming language to solve problems or accomplish a task (e.g., robotic functions, website designs, applications, and games). **TECH.8.2.12.E.4** - [*Cumulative Progress Indicator*] - Use appropriate terms in conversation (e.g., troubleshooting, peripherals, diagnostic software, GUI, abstraction, variables, data types and conditional statements).			

Unit 2: Variables, Conditionals, and Loops 3 weeks *Learning Processing* Lesson 2 (Chapters 4-6)	**MA.K-12.CCSS.Math.Practice.MP1** - [*Standard*] - Make sense of problems and persevere in solving them. **MA.K-12.CCSS.Math.Practice.MP2** - [*Standard*] - Reason abstractly and quantitatively. **MA.K-12.CCSS.Math.Practice.MP3** - [*Standard*] - Construct viable arguments and critique the reasoning of others. **MA.K-12.CCSS.Math.Practice.MP4** - [*Standard*] - Model with mathematics. **MA.K-12.CCSS.Math.Practice.MP5** - [*Standard*] - Use appropriate tools strategically. **MA.K-12.CCSS.Math.Practice.MP6** - [*Standard*] - Attend to precision. **MA.K-12.CCSS.Math.Practice.MP7** - [*Standard*] - Look for and make use of structure. **MA.K-12.CCSS.Math.Practice.MP8** - [*Standard*] - Look for and express regularity in repeated reasoning. **TECH.8.1.12.A.1** - [*Cumulative Progress Indicator*] - Create a personal digital portfolio which reflects personal and academic interests, achievements, and career aspirations by using a variety of digital tools and resources. **TECH.8.1.12.A.3** - [*Cumulative Progress Indicator*] - Collaborate in online courses, learning communities, social networks or virtual worlds to discuss a resolution to a problem or issue. **TECH.8.1.12.B.2** - [*Cumulative Progress Indicator*] - Apply previous content knowledge by creating and piloting a digital learning game or tutorial. **TECH.8.1.12.F.CS2** - [*Content Statement*] - Plan and manage activities to develop a solution or complete a project.	What decisions must be made when declaring and assigning variables? Why are conditional statements important in the creation of complex programs? When and how should a loop be implemented in a program?	Students will understand that… ● A variable is a pointer to a location in a computer's memory where data is stored. ● A variable must have a data type: integer, float, boolean, character, string, and byte. ● If a number appears three or more times, it should be a variable. ● Conditional expressions are used by a program to produce different results based on varying circumstances. ● Looping is the generative process of repeating a set of rules or steps over and over again. Students will know… ● Some variable names are reserved by Processing and are called system variables including mouseX, mouseY, width, height, frameCount, mousePressed, etc. ● These system variables are updated as a program executes. ● A boolean variable is either true or false and are useful as program flags. ● The order of operations determines which operators perform their calculations before others. ● Know how to use the conditional expressions (if, else if, else).	● Homework ● Classwork ● Quiz ● Test Unit 2 Project: *(continued from Unit 1)* 1. Create a static screen drawing using variables and a *for* loop. 2. Write assignment operations that alter the values of those variables and make the design dynamic. 3. Use conditional statements to alter the behavior of your design under certain circumstances. 4. Control a Bouncing Ball or Zoog using a combination of conditional and relational operators.

	TECH.8.2.12.C.1 - [*Cumulative Progress Indicator*] - Explain how open source technologies follow the design process. **TECH.8.2.12.E.1** - [*Cumulative Progress Indicator*] - Demonstrate an understanding of the problem-solving capacity of computers in our world. **TECH.8.2.12.E.3** - [*Cumulative Progress Indicator*] - Use a programming language to solve problems or accomplish a task (e.g., robotic functions, website designs, applications, and games). **TECH.8.2.12.E.4** - [*Cumulative Progress Indicator*] - Use appropriate terms in conversation (e.g., troubleshooting, peripherals, diagnostic software, GUI, abstraction, variables, data types and conditional statements).		● Relational expressions compare two variables by using relational operators (>, <, >=, <=, ==, !=) . A relational expression evaluates to either true or false. ● How variable scoping works, variables declared inside any block can only be accessed only inside their own block and inside any blocks enclosed within their block. ● How while, do-while, and for loops operate. Students will be able to... ● Write and publish a basic program using the Processing language. ● Use the shortcuts for expression evaluation i.e. y -=5; ● Use conditional and relational expressions to control what lines of code will be executed. ● Use looping to control program flow. ● Use nested loops to control repetition in more than one dimension.	
Unit 3: Organization -- Functions and Objects	**MA.K-12.CCSS.Math.Practice.MP1** - [*Standard*] - Make sense of problems and persevere in solving them.	What is Object Oriented Programming?	Students will understand that... ● A function is a self-contained programming module which	● Homework ● Classwork ● Quiz ● Test

3 weeks *Learning Processing* Lesson 3 (Chapters 7-8)	**MA.K-12.CCSS.Math.Practice.MP2** - [*Standard*] - Reason abstractly and quantitatively. **MA.K-12.CCSS.Math.Practice.MP3** - [*Standard*] - Construct viable arguments and critique the reasoning of others. **MA.K-12.CCSS.Math.Practice.MP4** - [*Standard*] - Model with mathematics. **MA.K-12.CCSS.Math.Practice.MP5** - [*Standard*] - Use appropriate tools strategically. **MA.K-12.CCSS.Math.Practice.MP6** - [*Standard*] - Attend to precision. **MA.K-12.CCSS.Math.Practice.MP7** - [*Standard*] - Look for and make use of structure. **MA.K-12.CCSS.Math.Practice.MP8** - [*Standard*] - Look for and express regularity in repeated reasoning. **TECH.8.1.12.A.1** - [*Cumulative Progress Indicator*] - Create a personal digital portfolio which reflects personal and academic interests, achievements, and career aspirations by using a variety of digital tools and resources. **TECH.8.1.12.A.3** - [*Cumulative Progress Indicator*] - Collaborate in online courses, learning communities, social networks or virtual worlds to discuss a resolution to a problem or issue. **TECH.8.1.12.B.2** - [*Cumulative Progress Indicator*] - Apply previous content knowledge by creating and piloting a digital learning game or tutorial. **TECH.8.1.12.F.CS2** - [*Content Statement*] - Plan and manage activities to develop a solution or complete a project. **TECH.8.2.12.C.1** - [*Cumulative Progress Indicator*] - Explain how open source technologies follow the design process.	Why is it beneficial (or necessary) to use a function in a program?	may use input parameters to define its output action(s). ● Functions may be written from scratch or made available from libraries. ● It is not necessary to understand the inner workings of a function; all one needs to understand is how to use them - to know what the inputs are and how they affect the output. ● Object Oriented Programming(OOP) is a way of structuring and organizing what we have learned. The properties of an object are variables, and the stuff an object can do are called methods or functions. Students will know… ● Functions are often called by other names such as procedures, methods, or subroutines. ● In some programming languages, there is a distinction between a procedure(performs a task) and a function (calculates a value and returns it to the calling program). ● Functions improve program design, readability,and debugging by breaking a larger program into smaller parts. ● Functions can have other functions embedded into it. ● Functions allow the programmer to reuse code without retyping it.	Unit 3 Project: *(continued from Unit 2)* 1. Reorganize your code using functions. 2. Reorganize your code using a class and an object variable. 3. Add arguments to the Constructor of your class and make at least two objects with different variables. 4. Revise the ZOOG code to take advantage of functions . Namely create jiggleZoog() to change the x and y location of Zoog randomly and drawZoog() too draw Zoog's body, head, eyes and head. 5. The cookie cutter example in "Learning Processing".

	TECH.8.2.12.E.1 - [*Cumulative Progress Indicator*] - Demonstrate an understanding of the problem-solving capacity of computers in our world. **TECH.8.2.12.E.3** - [*Cumulative Progress Indicator*] - Use a programming language to solve problems or accomplish a task (e.g., robotic functions, website designs, applications, and games). **TECH.8.2.12.E.4** - [*Cumulative Progress Indicator*] - Use appropriate terms in conversation (e.g., troubleshooting, peripherals, diagnostic software, GUI, abstraction, variables, data types and conditional statements).		● A class definition in OOP refers to an object's attributes (properties) and an object's behavior (method). ● Objects are encapsulated, the values of its attributes are hidden from the outside world, only accessible via methods. ● An object's (e.g vegetable) attributes have two types: instance variables that pertain to each instance of an object, (e.g the price of a vegetable) and variables that pertain to every object and whose value is identical to is called a class variable. For example, if the tax leveled on every vegetable was the same, tax would be a class variable. ● Likewise, using the same reasoning as above, methods are of two types, instance and class. Students will be able to... ● Use drawing functions such as line(), ellipse(), and fill(). ● Create user defined functions that may return a value. ● Use objects in a variety of programming applications.	
Unit 4: Arrays	**MA.K-12.CCSS.Math.Practice.MP1** - [*Standard*] - Make sense of problems and persevere in solving them.	Why is it beneficial (or necessary) to use an array?	Students will understand that... ● An array is a structure that stores a list of variables.	● Homework ● Classwork ● Quiz

2 weeks *Learning Processing* Lesson 4 (Chapter 9)	**MA.K-12.CCSS.Math.Practice.MP2** - [*Standard*] - Reason abstractly and quantitatively. **MA.K-12.CCSS.Math.Practice.MP3** - [*Standard*] - Construct viable arguments and critique the reasoning of others. **MA.K-12.CCSS.Math.Practice.MP4** - [*Standard*] - Model with mathematics. **MA.K-12.CCSS.Math.Practice.MP5** - [*Standard*] - Use appropriate tools strategically. **MA.K-12.CCSS.Math.Practice.MP6** - [*Standard*] - Attend to precision. **MA.K-12.CCSS.Math.Practice.MP7** - [*Standard*] - Look for and make use of structure. **MA.K-12.CCSS.Math.Practice.MP8** - [*Standard*] - Look for and express regularity in repeated reasoning. **TECH.8.1.12.A.1** - [*Cumulative Progress Indicator*] - Create a personal digital portfolio which reflects personal and academic interests, achievements, and career aspirations by using a variety of digital tools and resources. **TECH.8.1.12.A.3** - [*Cumulative Progress Indicator*] - Collaborate in online courses, learning communities, social networks or virtual worlds to discuss a resolution to a problem or issue. **TECH.8.1.12.B.2** - [*Cumulative Progress Indicator*] - Apply previous content knowledge by creating and piloting a digital learning game or tutorial. **TECH.8.1.12.F.CS2** - [*Content Statement*] - Plan and manage activities to develop a solution or complete a project. **TECH.8.2.12.C.1** - [*Cumulative Progress Indicator*] - Explain how open source technologies follow the design process.	How care arrays used in a dynamic program?	● While a variable points to a single piece of data, an array points to multiple pieces of data. ● An array is a programming structure that allows multiple lines of code to be reduced to a few lines. ● By using an array, it is relatively easy to transition from handling 10 objects to 10 000 objects. Students will know… ● Any time a program requires multiple instances of similar data, it might be time to use an array. ● The way in which loops perform array operations. ● When it is appropriate to use multidimensional arrays. Students will be able to… ● Recognize when to use an array. ● Declare an array. ● Implement arrays when appropriate. ● Use the Processing array functions to manipulate an array - expand or shrink it. The functions are: shorten(), concat(), subset(), append(), splice(), expand(), sort(), and reverse(). ● Create an array of objects.	● Test Unit 4 Project: *(continued from Unit 3)* 1. Make an array of objects from the class you created. 2. Have the objects react to the mouse.

	TECH.8.2.12.E.1 - [*Cumulative Progress Indicator*] - Demonstrate an understanding of the problem-solving capacity of computers in our world. **TECH.8.2.12.E.3** - [*Cumulative Progress Indicator*] - Use a programming language to solve problems or accomplish a task (e.g., robotic functions, website designs, applications, and games). **TECH.8.2.12.E.4** - [*Cumulative Progress Indicator*] - Use appropriate terms in conversation (e.g., troubleshooting, peripherals, diagnostic software, GUI, abstraction, variables, data types and conditional statements).			
Unit 5: Algorithms, Debugging, and Libraries 2 weeks *Learning Processing* Lesson 5 (Chapters 10-12)	**MA.K-12.CCSS.Math.Practice.MP1** - [*Standard*] - Make sense of problems and persevere in solving them. **MA.K-12.CCSS.Math.Practice.MP2** - [*Standard*] - Reason abstractly and quantitatively. **MA.K-12.CCSS.Math.Practice.MP3** - [*Standard*] - Construct viable arguments and critique the reasoning of others. **MA.K-12.CCSS.Math.Practice.MP4** - [*Standard*] - Model with mathematics. **MA.K-12.CCSS.Math.Practice.MP5** - [*Standard*] - Use appropriate tools strategically. **MA.K-12.CCSS.Math.Practice.MP6** - [*Standard*] - Attend to precision. **MA.K-12.CCSS.Math.Practice.MP7** - [*Standard*] - Look for and make use of structure. **MA.K-12.CCSS.Math.Practice.MP8** - [*Standard*] - Look for and express regularity in repeated reasoning. **TECH.8.1.12.A.1** - [*Cumulative Progress Indicator*] - Create a personal digital portfolio which reflects personal and academic interests, achievements,	What is the importance of using an algorithm when programming? What are some of the most basic strategies for debugging? Why are libraries an essential part of programming?	Students will understand that... ● Programming your own software is terrifically exciting because of the immeasurable array of possibilities for creation. ● Going from ideas to code requires an algorithm - how to solve the problem that needs to take into consideration the tools and features found in your programming environment. ● Debugging is an important part of writing a program. Students will know... ● A library consists of functions, variables, and objects that your program needs to use. ● A bug is a defect or error in a program. ● Once the algorithm is designed, it is evaluated to determine how it can be implemented using	● Homework ● Classwork ● Quiz ● Test Unit 5 Project: 1. Design a program you'd like to create that makes use of one or more libraries. 2. Create an algorithm (including a flowchart) with the design of your program. 3. Document at least one example of debugging for your program.

	and career aspirations by using a variety of digital tools and resources. **TECH.8.1.12.A.3** - [*Cumulative Progress Indicator*] - Collaborate in online courses, learning communities, social networks or virtual worlds to discuss a resolution to a problem or issue. **TECH.8.1.12.B.2** - [*Cumulative Progress Indicator*] - Apply previous content knowledge by creating and piloting a digital learning game or tutorial. **TECH.8.1.12.F.CS2** - [*Content Statement*] - Plan and manage activities to develop a solution or complete a project. **TECH.8.2.12.C.1** - [*Cumulative Progress Indicator*] - Explain how open source technologies follow the design process. **TECH.8.2.12.E.1** - [*Cumulative Progress Indicator*] - Demonstrate an understanding of the problem-solving capacity of computers in our world. **TECH.8.2.12.E.3** - [*Cumulative Progress Indicator*] - Use a programming language to solve problems or accomplish a task (e.g., robotic functions, website designs, applications, and games). **TECH.8.2.12.E.4** - [*Cumulative Progress Indicator*] - Use appropriate terms in conversation (e.g., troubleshooting, peripherals, diagnostic software, GUI, abstraction, variables, data types and conditional statements).		object oriented programming via the Processing IDE. ● The design should be robust enough so that other programmers can participate. ● Talking the debugging issue through with a friend forces you to organize the problem and might help you to realize the solution. ● Libraries are listed at the top of the program using the import function. ● The more you break your algorithm into smaller pieces and test those pieces, you'll have fewer errors and bugs, this is the philosophy of incremental design. Students will be able to… ● Use paragraphs, flow charting and/or some other graphical representation to show how the algorithm is implemented. ● Use pseudocode to work out the details of the algorithm. ● Break an algorithm down into smaller parts. ● Use one or more representations to "code" the algorithm. ● Implement the algorithm with code. ● Take the data and functionality associated with the algorithm and build it into a class.	

			● Take all the classes and integrate them into one larger algorithm. ● Use println() to display the value of variables as a program runs so you can observe the program running in real time. ● Instead of creating new functions, see if libraries are available.	
Unit 6: Applications of Processing 5 weeks *Learning Processing* Lesson 6 (Chapters 13-14) *The Nature of Code* (Various selections)	**MA.K-12.CCSS.Math.Practice.MP1** - [*Standard*] - Make sense of problems and persevere in solving them. **MA.K-12.CCSS.Math.Practice.MP2** - [*Standard*] - Reason abstractly and quantitatively. **MA.K-12.CCSS.Math.Practice.MP3** - [*Standard*] - Construct viable arguments and critique the reasoning of others. **MA.K-12.CCSS.Math.Practice.MP4** - [*Standard*] - Model with mathematics. **MA.K-12.CCSS.Math.Practice.MP5** - [*Standard*] - Use appropriate tools strategically. **MA.K-12.CCSS.Math.Practice.MP6** - [*Standard*] - Attend to precision. **MA.K-12.CCSS.Math.Practice.MP7** - [*Standard*] - Look for and make use of structure. **MA.K-12.CCSS.Math.Practice.MP8** - [*Standard*] - Look for and express regularity in repeated reasoning. **TECH.8.1.12.A.1** - [*Cumulative Progress Indicator*] - Create a personal digital portfolio which reflects personal and academic interests, achievements, and career aspirations by using a variety of digital tools and resources.	What is Object Oriented Programming? What is the importance of using an algorithm when programming? What is the value of writing elegant code with consistent indentation?	Students will know... ● Object oriented programming and other advanced programming techniques can be used to simulate real world systems. Students will understand that... ● A model of the system must be defined. ● The assumptions of the model need to be clear to everyone writing software. ● Once the model is defined, an algorithm for simulating the model can be created. ● The algorithm should be crafted using object-oriented programming structures and other advanced programming techniques. ● The user experience for interacting with the program needs to be simple. ● Model assumptions need to be realistic, so students must research their model using their	● Homework ● Classwork ● Quiz ● Test Unit 6 Project: Students will choose one or more chapters from *The Nature of Code* and explore it independently, creating a standalone program that reflect what they've learned. Students will present their program project in a presentation at the end of the semester.

	TECH.8.1.12.A.3 - [*Cumulative Progress Indicator*] - Collaborate in online courses, learning communities, social networks or virtual worlds to discuss a resolution to a problem or issue. **TECH.8.1.12.B.2** - [*Cumulative Progress Indicator*] - Apply previous content knowledge by creating and piloting a digital learning game or tutorial. **TECH.8.1.12.C.1** - [*Cumulative Progress Indicator*] - Develop an innovative solution to a real world problem or issue in collaboration with peers and experts, and present ideas for feedback through social media or in an online community. **TECH.8.1.12.F.CS2** - [*Content Statement*] - Plan and manage activities to develop a solution or complete a project. **TECH.8.2.12.C.1** - [*Cumulative Progress Indicator*] - Explain how open source technologies follow the design process. **TECH.8.2.12.E.1** - [*Cumulative Progress Indicator*] - Demonstrate an understanding of the problem-solving capacity of computers in our world. **TECH.8.2.12.E.2** - [*Cumulative Progress Indicator*] - Analyze the relationships between internal and external computer components. **TECH.8.2.12.E.3** - [*Cumulative Progress Indicator*] - Use a programming language to solve problems or accomplish a task (e.g., robotic functions, website designs, applications, and games). **TECH.8.2.12.E.4** - [*Cumulative Progress Indicator*] - Use appropriate terms in conversation (e.g., troubleshooting, peripherals, diagnostic software, GUI, abstraction, variables, data types and conditional statements).		STEM knowledge, for example consulting their math teacher about fractals, their biology teacher about genetic algorithms, or their physics teacher about what happens when objects collide. Students will be able to… ● Prioritize the areas in which you will spend the most programming effort: user input/output or algorithm development. ● Specify classes and functions to operate on instances of the class. ● Develop a testing strategy for individual routines, test from the bottom up and test often. ● Use a software control system for building releases if working on a multi-programmer project. ● Create a project presentation that includes a model description, the model assumptions, the algorithm for implementing the model, the model's software design, how the user interface was implemented, the testing strategy for the software, and finally whether the simulation output matches the output as predicted by the model.	
