Coding

Unit Title: CS Discoveries Unit 1: Problem Solving (Grades 6)

Stage 1: Desired Results

Standards & Indicators:

2020 NJSLS – Computer Science and Design Thinking
8.2.8.ED.2: Identify the steps in the design process that could be used to solve a problem.

Computer Science and Design Thinking

Standard	Performance Expectations	Core Ideas
8.2.8.ED.2	Identify the steps in the design process that could be used to solve a problem	Engineering design is a systematic, creative, and iterative process used to address local and global problems. The process includes generating ideas, choosing the best solution, and making, testing, and redesigning models or prototypes

Career Readiness, Life Literacies and Key Skills

Standard	Performance Expectations	Core Ideas
9.4.8.CI.3	Examine challenges that may exist in the adoption of new ideas.	Gathering and evaluating knowledge and information from a variety of sources, including global perspectives, fosters creativity and innovative thinking.
9.4.8.CT.2	Develop multiple solutions to a problem and evaluate short- and long-term effects to determine the most plausible option.	Multiple solutions often exist to solve a problem.

Central Idea/Enduring Understanding:
Students will use a structured problem solving process to address problems and build a collaborative classroom environment where students view computer science as relevant, fun, and empowering.

Essential/Guiding Question:

What strategies and processes can I use to become a more effective problem solver?

Content:

Lesson 1- Intro to Problem Solving
Lesson 2- The Problem Solving Process
Lesson 3- Exploring Problem Solving

Skills(Objectives):

- Communicate and collaborate with classmates in order to solve a problem
- Identify different strategies used to solve a problem
- Iteratively improve a solution to a problem
- 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
- 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

Interdisciplinary Connections:

Science:

- MS-ETS1-1: Define the criteria and constraints of a design problem with sufficient precision to ensure a successful solution, taking into account relevant scientific principles and potential impacts on people and the natural environment that may limit the possible solutions

Coding

- MS-ETS1-2: Evaluate competing design solutions using a systematic process to determine how well they meet the criteria and constraints of the problem.

Stage 2: Assessment Evidence

Performance Task(s):

8.2.8.ED.2: Using the problem solving process
[Template] Copy of U1L03 - Activity Guide - …

Other Evidence:

- Online assignments
- Exit Tickets
- Journal Entries
- Do Nows

Stage 3: Learning Plan

Learning Opportunities/Strategies:

Lesson 1- Intro to Problem Solving: Reflect on their experience with an activity and make connections to the types of problem solving they will be doing for the rest of the course.

Lesson 2- The Problem Solving Process: Relate the steps of the problem solving process to the problem from the previous lesson, then to a problem they are good at solving, then to a problem they want to improve at solving.

Lesson 3- Exploring Problem Solving: Apply the problem solving process to two different problems that are complex and poorly defined.

Resources:

- Lesson Presentations
- Google Docs
- Google Forms
- Google Classroom
- code.org

LGBT and Disabilities Law Resources:

- GLSEN Educator Resources
- Supporting LGBTQIA Youth Resource List
- Respect Ability: Fighting Stigmas, Advancing Opportunities

Differentiation *Please note: Teachers who have students with 504 plans that require curricular accommodations are to refer to Struggling and/or Special Needs Section for differentiation

High-Achieving Students	On Grade Level Students	Struggling Students	Special Needs/ELL
Higher order thinking questions Differentiation of pacing and activities Differentiation of learning strategies: visual, auditory, kinetic and cooperative Enrichment and extension	Differentiation of learning strategies: visual, auditory, kinetic and cooperative Differentiating the lesson activities Lesson tutorials	Provide a highly structured, predictable learning environment Provide organizers Lessons designed to the style of learning that matches the student Cooperative Learning Positive reinforcement Lessons presentation available on google classroom Frequent check for understanding Break down task into manageable units One-on-one instruction Pair student with a high achieving student	Any student requiring further accommodations and/or modifications will have them individually listed in their 504 Plan or IEP. These might include, but are not limited to: breaking assignments into smaller tasks, giving directions through several channels (auditory, visual, kinesthetic, model), and/or small group instruction for reading/writing ELL supports should include, but are not limited to, the following:: Extended time Provide visual aids Repeated directions Differentiate based on proficiency Provide word banks Allow for translators, dictionaries

Coding

Unit Title: CS Discoveries - Unit 2: Interactive Animations (Grade 6)

Stage 1: Desired Results

Standards & Indicators:

2020 NJSLS – Computer Science and Design Thinking
8.1.8.AP.2: Create clearly named variables that represent different data types and perform operations on their values.
8.1.8AP.3: Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.
8.1.8.AP.4: Decompose problems and sub-problems into parts to facilitate the design, implementation, and review of programs.
8.1.8.AP.5: Create procedures with parameters to organize code and make it easier to reuse.
8.1.8.AP.8: Systematically test and refine programs using a range of test cases and users.
8.1.8.AP.9: Document programs in order to make them easier to follow, test, and debug.

Computer Science and Design Thinking

Standard	Performance Expectations	Core Ideas
8.1.8.AP.2	Create clearly named variables that represent different data types and perform operations on their values.	Programmers create variables to store data values of different types and perform appropriate operations on their values
8.1.8.AP.3	Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.	Control structures are selected and combined in programs to solve more complex problems.
8.1.8.AP.4	Decompose problems and sub-problems into parts to facilitate the design, implementation, and review of programs	Programs use procedures to organize code and hide implementation details. Procedures can be repurposed in new programs. Defining parameters for procedures can generalize behavior and increase reusability.
8.1.8.AP.5	Create procedures with parameters to organize code and make it easier to reuse	
8.1.8.AP.8	Systematically test and refine programs using a range of test cases and users.	Individuals design and test solutions to identify problems taking into consideration the diverse needs of the users and the community
8.1.8.AP.9	Document programs in order to make them easier to follow, test, and debug.	

Career Readiness, Life Literacies and Key Skills

Standard	Performance Expectations	Core Ideas
9.4.8.CI.3	Examine challenges that may exist in the adoption of new ideas.	Gathering and evaluating knowledge and information from a variety of sources, including global perspectives, fosters creativity and innovative thinking.
9.4.8.CT.2	Develop multiple solutions to a problem and evaluate short- and long-term effects to determine the most plausible option.	Multiple solutions often exist to solve a problem.

Central Idea/Enduring Understanding:
Students should be able to create an interactive animation that includes basic programming concepts such as control structures, variables, user input, and randomness. Students should leave this unit viewing themselves as computer

Essential/Guiding Question:

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?

Coding

programmers, and see programming as a fun and creative form of expression.	
Content: Lesson 1- Programming for Entertainment Lesson 2- Plotting Shapes Lesson 3- Drawing in Game Lab Lesson 4- Shapes and Parameters Lesson 5- Variables Lesson 6- Random Numbers Lesson 7- Sprites Lesson 8- Sprite Properties Lesson 9- Text Lesson 10- Mini-Project- Captioned Scene Lesson 11- The Draw Loop Lesson 12- Sprite Movement Lesson 13- Mini- Project- Animation Lesson 14- Conditionals Lesson 15- Keyboard Inputs Lesson 16- Mouse Input Lesson 17- Project- Interactive Card	**Skills(Objectives):** ● Identify how computer science is used in a field of entertainment ● Communicate how to draw an image in Game Lab, accounting for shape position, color, and order ● Reason about locations on the Game Lab coordinate grid ● Sequence code correctly to overlay shapes. ● Use a coordinate system to place elements on the screen. ● Use and reason about drawing commands with multiple parameters ● 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 ● Generate and use random numbers in a program ● Update a value stored in a variable ● Create and use a sprite ● Use dot notation to update a sprite's properties ● Place text on the screen using a coordinate plane. ● Use arguments to control how text is displayed on a screen. ● Use a structured process to plan and develop a program. ● 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 ● Identify which sprite properties need to be changed, and in what way, to achieve a specific movement ● Use the counter pattern to increment or decrement sprite properties ● Use conditionals to react to changes in variables and sprite properties ● Move sprites in response to keyboard input ● Use conditionals to react to keyboard input ● Respond to a variety of types of user input. ● Use an if-else statement to control the flow of a program. ● Apply an iterator pattern to variables or properties in a loop ● Sequence commands to draw in the proper order ● Use conditionals to react to keyboard input or changes in variables / properties
Interdisciplinary Connections: **Visual and Performing Arts:** ● 1.5.8.Cr1a: Conceptualize early stages of the creative process, including applying methods to overcome creative blocks or take creative risks, and document the processes in traditional or new media.	

Coding

- 1.5.8.Cr1b: Develop criteria, identify goals and collaboratively investigate an aspect of present-day life, using contemporary practice of art or design.
- 1.5.8.Cr2a: Demonstrate persistence and willingness to experiment and take risks during the artistic process.
- 1.5.8.Cr2b: Demonstrate an awareness of ethical responsibility as applied to artmaking including environmental implications, responsibility in sharing images online, appropriation, and intellectual property ethics.
- 1.5.8.Cr2c: Apply, organize and strategize methods for design and redesign of objects, places, systems, images and words to clearly communicate information to a diverse audience.
- 1.5.8.Cr3a: Use criteria to examine, reflect on and plan revisions for a work of art, and create an artistic statement.

Stage 2: Assessment Evidence

Performance Task(s):

Lesson 10: Mini-Project- Captioned Scene
https://studio.code.org/s/csd3-2021/lessons/10

Lesson 13: Mini-Project- Animation
https://studio.code.org/s/csd3-2021/lessons/13

Lesson 17: Project- Interactive Card
https://studio.code.org/s/csd3-2021/lessons/17

Other Evidence:

- Online assignments
- Exit Tickets
- Journal Entries
- Do Nows
- Reflection on assignments

Stage 3: Learning Plan

Learning Opportunities/Strategies:

Lesson 1- Programming for Entertainment: Explore how Computer Science and programming play a role in either a specific form of entertainment or as a vehicle of self expression.

Lesson 2- Plotting Shapes: Learn the difficulties of communicating how to draw with shapes and use tools in Game Lab.

Lesson 3- Drawing in Game Lab: Learn the basics of sequencing and debugging, and program images.

Lesson 4- Shapes and Parameters: Learn to draw with versions of ellipse() and rect() that include width and height parameters and to use the background() block.

Lesson 5- Variables: Introduce variables as a way to label a number in a program or save a randomly generated value.

Lesson 6- Random Numbers: Learn the randomNumber() block and how it can be used to create new behaviors in programs.

Lesson 7- Sprites: Discuss the various information that programs must keep track of and how sprites are a way to keep track of that

Resources:

- Lesson Presentations
- Google Docs
- Google Forms
- Google Classroom
- Code.org
- Game Lab - A browser-based JavaScript programming environment designed to create sprite-based drawings, animations and games, Enables students to switch between programming in blocks or text.

LGBT and Disabilities Law Resources:

- GLSEN Educator Resources
- Supporting LGBTQIA Youth Resource List
- Respect Ability: Fighting Stigmas, Advancing Opportunities

Coding

information. Learn how to assign each sprite an image. **Lesson 8- Sprite Properties:** Extend understanding of sprites by interacting with sprite properties. Reflect on the connections between properties and variables. **Lesson 9- Text:** Practice adding and placing text on the screen and controlling other text properties, such as size. **Lesson 10- Mini-Project- Captioned Scene:** Use the problem solving process as a model, define the scene to create, prepare by thinking of all of the code needed, try a plan in Game Lab, then reflect on what was created. **Lesson 11- The Draw Loop:** Learn how to combine the draw loop with random numbers to manipulate some simple animations. **Lesson 12- Sprite Movement:** Learn how to control sprite movement using a construct called the counter pattern and use the counter pattern to create various types of sprite movement. **Lesson 13- Mini- Project- Animation:** Combine different methods from previous lessons to create an animated scene.	

Differentiation *Please note: Teachers who have students with 504 plans that require curricular accommodations are to refer to Struggling and/or Special Needs Section for differentiation

High-Achieving Students	On Grade Level Students	Struggling Students	Special Needs/ELL
Higher order thinking questions Differentiation of pacing and activities Differentiation of learning strategies: visual, auditory, kinetic and cooperative Enrichment and extension Code.org challenge levels	Differentiation of learning strategies: visual, auditory, kinetic and cooperative Differentiating the lesson activities Lesson tutorials	Provide a highly structured, predictable learning environment Provide organizers Lessons designed to the style of learning that matches the student Cooperative Learning Positive reinforcement Lessons presentation available on google classroom Frequent check for understanding Break down task into manageable units One-on-one instruction	Any student requiring further accommodations and/or modifications will have them individually listed in their 504 Plan or IEP. These might include, but are not limited to: breaking assignments into smaller tasks, giving directions through several channels (auditory, visual, kinesthetic, model), and/or small group instruction for reading/writing ELL supports should include, but are not limited to, the following:: Extended time Provide visual aids Repeated directions Differentiate based on proficiency Provide word banks Allow for translators, dictionaries

Coding

		Pair student with a high achieving student Provide video tutorials	

Pacing Guide - Grade 6

Coding (CS Discoveries)	Resource	Standards
UNIT 1 Problem Solving and Computing (4 days)	**Lessons:** Lesson 1- Intro to Problem Solving: 2 days Lesson 2- The Problem Solving Process: 1 day Lesson 3- Exploring Problem Solving: 1 day	8.2.8.ED.2
UNIT 3 Interactive Animations (18 days)	**Lessons:** Lesson 1- Programming for Entertainment: 1 day Lesson 2- Plotting Shapes: 1 day Lesson 3- Drawing in Game Lab: 1 day Lesson 4- Shapes and Parameters: 1 day Lesson 5- Variables: 1 day Lesson 6- Random Numbers: 1 day Lesson 7- Sprites: 1 day Lesson 8- Sprite Properties: 1 day Lesson 9- Text: 1 day Lesson 10- Mini-Project- Captioned Scene: 1 day Lesson 11- The Draw Loop: 1 day Lesson 12- Sprite Movement: 1 day Lesson 13- Mini- Project- Animation: 1 day	8.1.8.AP.2 8.1.8AP.3 8.1.8.AP.4 8.1.8.AP.5 8.1.8.AP.8 8.1.8.AP.9

Coding

Unit Title: Unit 1: Microsoft MakeCode with Micro:Bits (Grade 7)

Stage 1: Desired Results

Standards & Indicators:

2020 NJSLS – Computer Science and Design Thinking
8.1.8.AP.2: Create clearly named variables that represent different data types and perform operations on their values.
8.1.8AP.3: Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.
8.1.8.AP.4: Decompose problems and sub-problems into parts to facilitate the design, implementation, and review of programs.
8.1.8.AP.5: Create procedures with parameters to organize code and make it easier to reuse.
8.1.8.AP.8: Systematically test and refine programs using a range of test cases and users.
8.1.8.AP.9: Document programs in order to make them easier to follow, test, and debug.

Computer Science and Design Thinking

Standard	Performance Expectations	Core Ideas
8.1.8.AP.2	Create clearly named variables that represent different data types and perform operations on their values.	Programmers create variables to store data values of different types and perform appropriate operations on their values
8.1.8.AP.3	Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.	Control structures are selected and combined in programs to solve more complex problems.
8.1.8.AP.4	Decompose problems and sub-problems into parts to facilitate the design, implementation, and review of programs	Programs use procedures to organize code and hide implementation details. Procedures can be repurposed in new programs. Defining parameters for procedures can generalize behavior and increase reusability.
8.1.8.AP.5	Create procedures with parameters to organize code and make it easier to reuse	
8.1.8.AP.8	Systematically test and refine programs using a range of test cases and users.	Individuals design and test solutions to identify problems taking into consideration the diverse needs of the users and the community
8.1.8.AP.9	Document programs in order to make them easier to follow, test, and debug.	

Career Readiness, Life Literacies and Key Skills

Standard	Performance Expectations	Core Ideas
9.4.8.CI.3	Examine challenges that may exist in the adoption of new ideas.	Gathering and evaluating knowledge and information from a variety of sources, including global perspectives, fosters creativity and innovative thinking.
9.4.8.CT.2	Develop multiple solutions to a problem and evaluate short- and long-term effects to determine the most plausible option.	Multiple solutions often exist to solve a problem.

Central Idea/Enduring Understanding:	Essential/Guiding Question:
Students should be able to create multiple programs for a Micro:Bit processor that includes basic programming concepts such as control structures, variables, user input, and randomness. Students should leave this unit viewing themselves	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?

Coding

as computer programmers, and see programming as a fun and creative form of expression.	
Content: Lesson 1- Intro to Micro:Bits and Course Lesson 2- Happy & Sad Face Buttons Lesson 3- Pet Hamster Lesson 4- Dice Lesson 5- Rock, Paper, Scissors Lesson 6- 7 Seconds Game Lesson 7- Tug of LED Lesson 8- Reaction Time Lesson 9- Snap the Dot Lesson 10- Magic 8 Ball Assessment	**Skills(Objectives):** ● Use a coordinate system to place elements ● 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 ● Generate and use random numbers in a program ● Update a value stored in a variable ● Create and use a sprite ● Use a structured process to plan and develop a program. ● Use conditionals to react to changes in variables ● Use conditionals to react to various inputs ● Respond to a variety of types of user input. ● Use an if-else statement to control the flow of a program. ● Sequence code to run in a certain order ● Use conditionals to react to input or changes in variables / properties
Interdisciplinary Connections: **Visual and Performing Arts:** ● 1.5.8.Cr1a: Conceptualize early stages of the creative process, including applying methods to overcome creative blocks or take creative risks, and document the processes in traditional or new media. ● 1.5.8.Cr2a: Demonstrate persistence and willingness to experiment and take risks during the artistic process. ● 1.5.8.Cr2c: Apply, organize and strategize methods for design and redesign of objects, places, systems, images and words to clearly communicate information to a diverse audience. ● 1.5.8.Cr3a: Use criteria to examine, reflect on and plan revisions for a work of art, and create an artistic statement. **Mathematics:** ● 7.NS.A.1 Apply and extend previous understandings of addition and subtraction to add and subtract rational numbers; represent addition and subtraction on a horizontal or vertical number line diagram.	
Stage 2: Assessment Evidence	
Performance Task(s): Lesson 10- Magic 8 Ball	**Other Evidence:** ● Online assignments ● Exit Tickets ● Journal Entries ● Do Nows ● Reflection on assignments
Stage 3: Learning Plan	
Learning Opportunities/Strategies: **Lesson 1- Intro to Micro:Bits and Course**- Introduction to the Micro:Bit and MakeCode programming **Lesson 2- Happy & Sad Face Buttons**- Gain an understanding of the input buttons and outputs of the Micro:Bit and how to program them.	**Resources:** ● Lesson Presentations ● Google Docs ● Google Forms ● Google Classroom ● makecode.microbit.org LGBT and Disabilities Law Resources: ● GLSEN Educator Resources ● Supporting LGBTQIA Youth Resource List

Coding

Lesson 3- Pet Hamster- Gain an understanding of the additional input options and outputs of the Micro:Bit and how to program them. **Lesson 4- Dice**- Learn how to incorporate a random number picker into code. **Lesson 5- Rock, Paper, Scissors**- Learn how to incorporate if/then logic statements into a code tied to the random number picker. **Lesson 6- 7 Seconds Game**- Learn how to incorporate game based code blocks into code. **Lesson 7- Tug of LED**- Learn how to create and manipulate a variable within a game. **Lesson 8- Reaction Time**- Learn how to utilize game based code with inputs and outputs. **Lesson 9- Snap the Dot**- Learn how to utilize game based code with input and reaction time, learn how to adjust sprite movement speed **Lesson 10- Magic 8 Ball Assessment**- Create a Magic 8 Ball program utilizing code learned throughout the unit.	• Respect Ability: Fighting Stigmas, Advancing Opportunities

Differentiation *Please note: Teachers who have students with 504 plans that require curricular accommodations are to refer to Struggling and/or Special Needs Section for differentiation

High-Achieving Students	On Grade Level Students	Struggling Students	Special Needs/ELL
Higher order thinking questions Differentiation of pacing and activities Differentiation of learning strategies: visual, auditory, kinetic and cooperative Enrichment and extension Advanced Microsoft MakeCode Activities	Differentiation of learning strategies: visual, auditory, kinetic and cooperative Differentiating the lesson activities Lesson tutorials	Provide a highly structured, predictable learning environment Provide organizers Lessons designed to the style of learning that matches the student Cooperative Learning Positive reinforcement Lessons presentation available on google classroom Frequent check for understanding Break down task into manageable units One-on-one instruction Pair student with a high achieving student	Any student requiring further accommodations and/or modifications will have them individually listed in their 504 Plan or IEP. These might include, but are not limited to: breaking assignments into smaller tasks, giving directions through several channels (auditory, visual, kinesthetic, model), and/or small group instruction for reading/writing ELL supports should include, but are not limited to, the following:: Extended time Provide visual aids Repeated directions Differentiate based on proficiency Provide word banks Allow for translators, dictionaries

		Provide video tutorials	

Unit Title: Unit 2: Microsoft MakeCode with Finch Robots (Grade 7)

Stage 1: Desired Results

Standards & Indicators:

2020 NJSLS – Computer Science and Design Thinking
8.1.8.AP.2: Create clearly named variables that represent different data types and perform operations on their values.
8.1.8AP.3: Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.
8.1.8.AP.4: Decompose problems and sub-problems into parts to facilitate the design, implementation, and review of programs.
8.1.8.AP.5: Create procedures with parameters to organize code and make it easier to reuse.
8.1.8.AP.8: Systematically test and refine programs using a range of test cases and users.
8.1.8.AP.9: Document programs in order to make them easier to follow, test, and debug.

Computer Science and Design Thinking

Standard	Performance Expectations	Core Ideas
8.1.8.AP.2	Create clearly named variables that represent different data types and perform operations on their values.	Programmers create variables to store data values of different types and perform appropriate operations on their values
8.1.8.AP.3	Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.	Control structures are selected and combined in programs to solve more complex problems.
8.1.8.AP.4	Decompose problems and sub-problems into parts to facilitate the design, implementation, and review of programs	Programs use procedures to organize code and hide implementation details. Procedures can be repurposed in new programs. Defining parameters for procedures can generalize behavior and increase reusability.
8.1.8.AP.5	Create procedures with parameters to organize code and make it easier to reuse	
8.1.8.AP.8	Systematically test and refine programs using a range of test cases and users.	Individuals design and test solutions to identify problems taking into consideration the diverse needs of the users and the community
8.1.8.AP.9	Document programs in order to make them easier to follow, test, and debug.	

Career Readiness, Life Literacies and Key Skills

Standard	Performance Expectations	Core Ideas
9.4.8.CI.3	Examine challenges that may exist in the adoption of new ideas.	Gathering and evaluating knowledge and information from a variety of sources, including global perspectives, fosters creativity and innovative thinking.
9.4.8.CT.2	Develop multiple solutions to a problem and evaluate short- and long-term effects to determine the most plausible option.	Multiple solutions often exist to solve a problem.

Central Idea/Enduring Understanding:	Essential/Guiding Question:
Students should be able to create multiple programs for a Finch Robot that includes basic programming concepts such as control structures,	What is a computer program? What are the core features of most programming languages?

Coding

variables, user input, and randomness from the previous unit. Students should leave this unit viewing themselves as computer programmers, and see programming as a fun and creative form of expression.	How does programming enable creativity and individual expression? What practices and strategies will help me as I write programs?
Content: Lesson 1- Intro to Finch Robots Lesson 2- Figure 8 Lesson 3- Socially Distanced Finch Lesson 4- Obstacle Detection Lesson 5- Robot Maze Assessment Lesson 6- Robot Shapes	**Skills(Objectives):** ● 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 ● Generate and use random numbers in a program ● Update a value stored in a variable ● Create and use a sprite ● Use a structured process to plan and develop a program. ● Use conditionals to react to changes in variables ● Use conditionals to react to various inputs ● Respond to a variety of types of user input. ● Use an if-else statement to control the flow of a program. ● Sequence code to run in a certain order ● Use conditionals to react to input or changes in variables / properties

Interdisciplinary Connections:

Visual and Performing Arts:

- 1.5.8.Cr1a: Conceptualize early stages of the creative process, including applying methods to overcome creative blocks or take creative risks, and document the processes in traditional or new media.
- 1.5.8.Cr2a: Demonstrate persistence and willingness to experiment and take risks during the artistic process.
- 1.5.8.Cr2c: Apply, organize and strategize methods for design and redesign of objects, places, systems, images and words to clearly communicate information to a diverse audience.
- 1.5.8.Cr3a: Use criteria to examine, reflect on and plan revisions for a work of art, and create an artistic statement.

Mathematics:

- 7.G.B.4 Know the formulas for the area and circumference of a circle and use them to solve problems
- 7.G.A.2 Draw (with technology, with ruler and protractor, as well as freehand) geometric shapes with given conditions.

Stage 2: Assessment Evidence

Performance Task(s): Lesson 5- Robot Maze	**Other Evidence:** ● Online assignments ● Exit Tickets ● Journal Entries ● Do Nows ● Reflection on assignments

Stage 3: Learning Plan

Learning Opportunities/Strategies: **Lesson 1- Intro to Finch Robots**- Introduce Finch robots and additional coding skills needed for operation. Learn how to code basic movement, distance and speed.	**Resources:** ● Lesson Presentations ● Google Docs ● Google Forms ● Google Classroom ● makecode.microbit.org

Coding

Lesson 2- Figure 8- Learn how to use code to turn the robot and create a loop that can be followed to draw or follow a figure 8. **Lesson 3- Socially Distanced Finch**- Learn how to use the proximity sensor and have the Finch react to the change in proximity accordingly. **Lesson 4- Obstacle Detection**- Continue using proximity sensors and learn to create a set of code to follow once an obstacle is detected a certain distance away from the robot. **Lesson 5- Robot Maze Assessment**- Students will use knowledge to create a code that will guide the robot successfully through the maze. **Lesson 6- Robot Shapes**- Create code to have the robots draw basic shapes to create a picture.	LGBT and Disabilities Law Resources: ● GLSEN Educator Resources ● Supporting LGBTQIA Youth Resource List ● Respect Ability: Fighting Stigmas, Advancing Opportunities

Differentiation *Please note: Teachers who have students with 504 plans that require curricular accommodations are to refer to Struggling and/or Special Needs Section for differentiation

High-Achieving Students	On Grade Level Students	Struggling Students	Special Needs/ELL
Higher order thinking questions Differentiation of pacing and activities Differentiation of learning strategies: visual, auditory, kinetic and cooperative Enrichment and extension Advanced Microsoft MakeCode Activities	Differentiation of learning strategies: visual, auditory, kinetic and cooperative Differentiating the lesson activities Lesson tutorials	Provide a highly structured, predictable learning environment Provide organizers Lessons designed to the style of learning that matches the student Cooperative Learning Positive reinforcement Lessons presentation available on google classroom Frequent check for understanding Break down task into manageable units One-on-one instruction Pair student with a high achieving student Provide video tutorials	Any student requiring further accommodations and/or modifications will have them individually listed in their 504 Plan or IEP. These might include, but are not limited to: breaking assignments into smaller tasks, giving directions through several channels (auditory, visual, kinesthetic, model), and/or small group instruction for reading/writing ELL supports should include, but are not limited to, the following:: Extended time Provide visual aids Repeated directions Differentiate based on proficiency Provide word banks Allow for translators, dictionaries

Coding

Pacing Guide - Grade 7

Coding (MakeCode)	Resource	Standards
UNIT 1 Unit 1: Microsoft MakeCode with Micro:Bits (13 days)	**Lessons:** Lesson 1- Intro to Micro:Bits and Course: 1 day Lesson 2- Happy & Sad Face Buttons: 1 day Lesson 3- Pet Hamster : 1 day Lesson 4- Dice: 1 day Lesson 5- Rock, Paper, Scissors: 1 day Lesson 6- 7 Seconds Game: 1 day Lesson 7- Tug of LED: 2 days Lesson 8- Reaction Time: 1 day Lesson 9- Snap the Dot: 1 day Lesson 10- Magic 8 Ball Assessment: 2 days	8.1.8.AP.2 8.1.8AP.3 8.1.8.AP.4 8.1.8.AP.5 8.1.8.AP.8 8.1.8.AP.9
UNIT 2 Microsoft MakeCode with Finch Robots (9 days)	**Lessons:** Lesson 1- Intro to Finch Robots: 1 day Lesson 2- Figure 8: 1 day Lesson 3- Socially Distanced Finch: 1 day Lesson 4- Obstacle Detection: 2 days Lesson 5- Robot Maze Assessment: 2 days Lesson 6- Robot Shapes: 2 days	8.1.8.AP.2 8.1.8AP.3 8.1.8.AP.4 8.1.8.AP.5 8.1.8.AP.8 8.1.8.AP.9

Unit Title: Creating Apps with Devices (Micro:Bit) (Grade 8)

Stage 1: Desired Results

Standards & Indicators:

2020 NJSLS – Computer Science and Design Thinking
8.1.8.AP.2: Create clearly named variables that represent different data types and perform operations on their values.

Coding

8.1.8AP.3: Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.
8.1.8.AP.4: Decompose problems and sub-problems into parts to facilitate the design, implementation, and review of programs.
8.1.8.AP.5: Create procedures with parameters to organize code and make it easier to reuse.
8.1.8.AP.8: Systematically test and refine programs using a range of test cases and users.
8.1.8.AP.9: Document programs in order to make them easier to follow, test, and debug.

Computer Science and Design Thinking

Standard	Performance Expectations	Core Ideas
8.1.8.AP.2	Create clearly named variables that represent different data types and perform operations on their values.	Programmers create variables to store data values of different types and perform appropriate operations on their values
8.1.8.AP.3	Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.	Control structures are selected and combined in programs to solve more complex problems.
8.1.8.AP.4	Decompose problems and sub-problems into parts to facilitate the design, implementation, and review of programs	Programs use procedures to organize code and hide implementation details. Procedures can be repurposed in new programs. Defining parameters for procedures can generalize behavior and increase reusability.
8.1.8.AP.5	Create procedures with parameters to organize code and make it easier to reuse	
8.1.8.AP.8	Systematically test and refine programs using a range of test cases and users.	Individuals design and test solutions to identify problems taking into consideration the diverse needs of the users and the community
8.1.8.AP.9	Document programs in order to make them easier to follow, test, and debug.	

Career Readiness, Life Literacies and Key Skills

Standard	Performance Expectations	Core Ideas
9.4.8.CI.3	Examine challenges that may exist in the adoption of new ideas.	Gathering and evaluating knowledge and information from a variety of sources, including global perspectives, fosters creativity and innovative thinking.
9.4.8.CT.2	Develop multiple solutions to a problem and evaluate short- and long-term effects to determine the most plausible option.	Multiple solutions often exist to solve a problem.

Central Idea/Enduring Understanding:
Students should be able to create multiple programs for a Micro:Bit processor that includes basic programming concepts such as control structures, variables, user input, and randomness. Students should leave this unit viewing themselves as computer programmers, and see programming as a fun and creative form of expression.

Essential/Guiding Question:
What inputs and outputs are available on a physical device?

What inputs and outputs are available on an app?

How can we create apps that use a physical device to control a digital app?

How can a physical device use sensors to react to a physical environment?

How can simple hardware be used to develop innovative new products?

Content:
Lesson 1- Intro to App Lab
Lesson 2- Physical Designs

Skills(Objectives):
- Build and share their own apps in App Lab using features like buttons, text, images, sound, and screens.

Coding

Lesson 3- Introducing the Micro:Bit Lesson 4- Updating Screen Elements Lesson 5- Board Events Lesson 6- Variables and If Statements Lesson 7- Mini-Project: Field Collector App Lesson 8- Getting Screen Inputs Lesson 9- Combining Inputs and Outputs Lesson 10- Project: Human Device Interaction Lesson 11- Board Sensors Lesson 12- Accelerometer Lesson 13- Functions Lesson 14- Mini-Project: Interactive Pet Lesson 15- Physical Outputs and LEDs Lesson 16- Physical Inputs and Buttons Lesson 17- Project: Prototype an Innovation	● Create a prototype of a physical design to meet the needs of a user using the problem-solving process ● Identify features of a physical design that match the needs of users ● Understand the steps of the problem-solving process ● Connect and troubleshoot external micro:bit devices ● Control the micro:bit's LED display with code: turn individual LEDs on and off, scroll words and numbers ● Change text on the screen using code ● Respond to user input using event handlers ● Set the properties of UI elements using code ● Use a board event handler to control buttons on the micro:bit ● Use if-statements to make decisions when creating apps ● Use the counter pattern to update variables when creating apps ● Use variables to store information when creating apps ● Create a project that incorporates inputs from the micro:bit ● Create a project that incorporates outputs from the app screen ● Create a project that incorporates variables and if-statements ● Design apps using the text input, dropdown, and slider design elements ● Use getProperty and getText to access user input on the screen ● Use user input to update elements on the screen using code ● Use inputs from app design elements to update screen elements or produce output on the micro:bit ● Use inputs from the micro:bit to update screen elements or produce output on the micro:bit ● Use variables and if-statements for complex user interaction ● Create a project that incorporates physical materials into its design ● Create a project that uses a variety of inputs and outputs from the micro:bit and app screen ● Develop programs that respond to sensor input ● Scale a range of numbers to meet a specific need ● Use accelerometer orientation (pitch and roll) when creating apps ● Use the data event to continually update an element's properties. ● Use functions to organize repeated blocks of code ● Use parameters to generalize the purpose of a function ● Create a project that combines output from the micro:bit with physical materials ● Create a project that seamlessly integrates the micro:bit with physical materials ● Create and debug a circuit with an LED

Coding

	• Create prototype devices that control an LED with code • Create and debug a circuit with a button • Create prototype devices that use an external button for input • Implement a plan for developing a piece of software that integrates hardware inputs and outputs • Independently scope the features of a piece of software • Prototype a physical computing device

Interdisciplinary Connections:

Visual and Performing Arts: 1.5.8.Cr1a: Conceptualize early stages of the creative process, including applying methods to overcome creative blocks or take creative risks, and document the processes in traditional or new media.

- 1.5.8.Cr2a: Demonstrate persistence and willingness to experiment and take risks during the artistic process.
- 1.5.8.Cr2c: Apply, organize and strategize methods for design and redesign of objects, places, systems, images and words to clearly communicate information to a diverse audience.
- 1.5.8.Cr3a: Use criteria to examine, reflect on and plan revisions for a work of art, and create an artistic statement.

Stage 2: Assessment Evidence

Performance Task(s):	Other Evidence:
Lesson 7- Mini-Project: Field Collector App Lesson 10- Project: Human Device Interaction Lesson 14- Mini-Project: Interactive Pet Lesson 17- Project: Prototype an Innovation	• Online assignments • Exit Tickets • Journal Entries • Do Nows • Reflection on assignments

Stage 3: Learning Plan

Learning Opportunities/Strategies:	Resources:
Lesson 1- Intro to App Lab- Introduce the App Lab programming environment, learn to create and control buttons, text, images, sounds and screens in JavaScript. **Lesson 2- Physical Designs**- Investigate the design of different physical devices and their apps. **Lesson 3- Introducing the Micro:Bit**- Learn to create applications that use App Lab inputs to control the micro:bit's main output, its LED display, to show simple pictures, words and numbers. **Lesson 4- Updating Screen Elements**- Learn how to interact with apps, we introduce the setProperty() and setText() blocks that allow users to change the properties and content of various UI elements. **Lesson 5- Board Events**- Learn to use **onBoardEvent()**, analogously to **onEvent()**, in order to take input from their micro:bit.	• Lesson Presentations • Google Docs • Google Forms • Google Classroom • code.org LGBT and Disabilities Law Resources: • GLSEN Educator Resources • Supporting LGBTQIA Youth Resource List • Respect Ability: Fighting Stigmas, Advancing Opportunities

Coding

Lesson 6- Variables and If Statements- Learn to use variables, the counter pattern and if-statements to create more complex input/output behaviors. **Lesson 7- Mini-Project: Field Collector App**- Use the micro:bit to collect data, then use App Lab to analyze the data that was collected. **Lesson 8- Getting Screen Inputs**- Learn to use several new design elements - text inputs, dropdowns, and sliders - so they can get user input from the screen of their apps. **Lesson 9- Combining Inputs and Outputs**- Learn to combine inputs and outputs across both the micro:bit and the app screen. **Lesson 10- Project: Human Device Interaction**- Create an app that controls the micro:bit so it interacts with the physical environment around it. **Lesson 11- Board Sensors**- Learn how the two sensors (light and temperature) can be used to write programs that respond to changes in the environment. **Lesson 12- Accelerometer**- Explore the accelerometer and its capabilities, events and properties. **Lesson 13- Functions**- Learn to use functions, and their parameters, as way to organize and group repeated blocks of code together **Lesson 14- Mini-Project: Interactive Pet**- Create an interactive "pet" using the sensors on the micro:bit, the LED Display to represent expressions, and physical materials. **Lesson 15- Physical Outputs and LEDs**- Learn how to attach external LEDs to their micro:bit and use code to light up these LEDs. **Lesson 16- Physical Inputs and Buttons**- Learn how to connect external wires to create input events when the wires touch, simulating a button press. **Lesson 17- Project: Prototype an Innovation**- Develop and test a prototype for an innovative computing device based on the micro:bit.	

Coding

Differentiation *Please note: Teachers who have students with 504 plans that require curricular accommodations are to refer to Struggling and/or Special Needs Section for differentiation			
High-Achieving Students	**On Grade Level Students**	**Struggling Students**	**Special Needs/ELL**
Higher order thinking questions Differentiation of pacing and activities Differentiation of learning strategies: visual, auditory, kinetic and cooperative Enrichment and extension Advanced Microsoft MakeCode Activities	Differentiation of learning strategies: visual, auditory, kinetic and cooperative Differentiating the lesson activities Lesson tutorials	Provide a highly structured, predictable learning environment Provide organizers Lessons designed to the style of learning that matches the student Cooperative Learning Positive reinforcement Lessons presentation available on google classroom Frequent check for understanding Break down task into manageable units One-on-one instruction Pair student with a high achieving student Provide video tutorials	Any student requiring further accommodations and/or modifications will have them individually listed in their 504 Plan or IEP. These might include, but are not limited to: breaking assignments into smaller tasks, giving directions through several channels (auditory, visual, kinesthetic, model), and/or small group instruction for reading/writing ELL supports should include, but are not limited to, the following:: Extended time Provide visual aids Repeated directions Differentiate based on proficiency Provide word banks Allow for translators, dictionaries

Pacing Guide - Grade 8

Coding (Apps with Micro:bit)	Resource	Standards
UNIT 1 Creating Apps with Devices (Micro:Bit) (23 days)	**Lessons:** Lesson 1- Intro to App Lab: 1 day Lesson 2- Physical Designs: 1day Lesson 3- Introducing the Micro:Bit: 1 day Lesson 4- Updating Screen Elements: 1 day Lesson 5- Board Events: 1 day Lesson 6- Variables and If Statements: 1 day	8.1.8.AP.2 8.1.8AP.3 8.1.8.AP.4 8.1.8.AP.5 8.1.8.AP.8 8.1.8.AP.9

Coding

	Lesson 7- Mini-Project: Field Collector App: 1 day Lesson 8- Getting Screen Inputs: 1 day Lesson 9- Combining Inputs and Outputs: 2 days Lesson 10- Project: Human Device Interaction: 2 days Lesson 11- Board Sensors: 1 day Lesson 12- Accelerometer: 1 day Lesson 13- Functions: 1 day Lesson 14- Mini-Project: Interactive Pet: 2 days Lesson 15- Physical Outputs and LEDs: 2 days Lesson 16- Physical Inputs and Buttons: 2 days Lesson 17- Project: Prototype an Innovation: 2 days	