Goldberg Gator Engineering Explorers Summer Program

Table of Contents

Program Materials

Technology	Craft Supplies	Activity Supplies
• Computers • Micro:bits - 1 per student • Servo motors • Dupont Wires (male –female, male – male) • Alligator Clips • AAA Batteries • Micro:bit Stem Kits – sensors, LED lights, motors	• Paper • Tape • Markers • Pencils • Scissors • Construction paper • Paper towel/toilet paper rolls • Cardboard/Cardstock • Other craft materials for design projects	• Sticky Notes • Sticky Easel Pad (optional) • Rulers • Flashlight/cell phone for light lab • Student Rewards

Day 1: Introductions, Programming Basics, Coding Applications

Overview

Activity	Time, minutes
Summer Camp Team Introduction, Introduction to Micro:bit and Ice Breaker	90
Programming Basics (Small group Activities)	110
Lunch and Recess	60
Problem Solving Activity	15
Coding Applications (Game and Data Collection)	120
Total	395
Extra Time	25

Materials Needed for Day 1:

Technology	Craft Supplies	Activity Supplies
• Computers • Micro:bits - 1 per student • AAA Batteries	• Markers • Pencils • Construction Paper	• Consent Forms • Link to survey • Name Tags • Sticky Notes • Sticky Easel Pad • Stickers • Rulers

Activity 1: Introductions, Introduction to Micro:bit and Ice Breaker

Estimated Time*: Day 1 Paperwork/Surveys/Sign into Teams – 20 minutes, Introduction, Team Introduction – 5 minutes, Ice breaker: Name Tag – 10 minutes, Programming Languages Activity - 20 minutes, IRB Consent – 10 mins, Establish Norms – 10 minutes, Intro to Micro:bit – 15 minutes.* ***Total time = 90 minutes***

Activity Goals

- Establish norms for working together and using the technology
- Introduce Facilitator Team
- Meet groups/partners

Icebreaker Activities

- Nametag Activity
  - Students will create a nametag using construction paper.
  - Show students how to fold a piece of paper into a trifold to create a nametag that can be placed by their computer.
  - Tell students to write their name on the paper and decorate it to represent themself.
- CS Unplugged: Programming Languages Activity

- Goal: Teaches students that computers work by following a list of instructions that they have to follow even if they do not make sense
- Demonstration Example: Have students draw a picture from the instructions you give them verbally. Page 3 of linked document
- Student Example: Choose a student to come to the front of the room and give them a simple drawing. That student has to verbally provide the instructions for the rest of the class to draw the image. Repeat with 1-2 more students.
- Discussion: Make the connections between how this exercise relates to computers and programming.
- https://classic.csunplugged.org/documents/activities/programming-languages/unplugged-12-programming_languages.pdf

Establish Norms for Working together

Ask students what rules/norms they have for working with people – chart them down

Share our norms and add student norms

- Ask questions
- Be present
- Treat others with respect
- Share your thoughts
- Keep an open mind
- Do your part
- Treat technology and tools with care
- Listen with intent

Introduction to Micro:bit Hardware and Programming

***Estimated Time**: 15 minutes*

Goals:

- Establish that computing is broken into inputs, outputs, and processes
- How to use the Mirco:bits – parts of the Micro:bit and saving code
- How to use MakeCode editor to program the Micro:bit
- How to program external attachments (sensors, LEDs, etc.)
- How to use Micro:bit to collect and analyze data

How to Use a Micro:bit and MakeCode

- Introduction to Micro:bit Presentation – this is a PowerPoint Presentation to use with students
- These are reference material for the intro – *TEACHER & LEADERS should review thoroughly*
    - Micro:bit first steps: https://microbit.org/get-started/first-steps/introduction/
    - Parts of a Micro:bit: https://microbit.org/get-started/user-guide/overview/

Activity 2: Small Group Activities – Programming Basics and Computational Thinking Skills

1. Process Mapping

***Estimated Time**: 25 minutes*

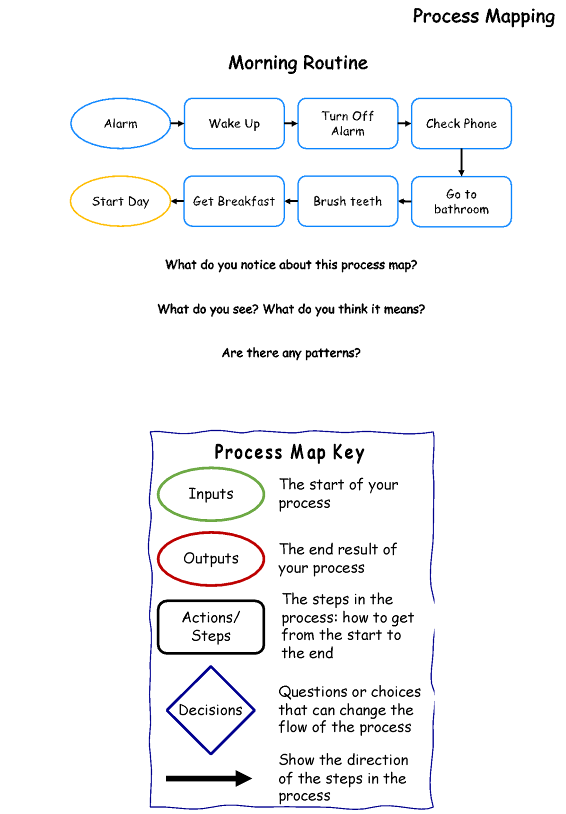
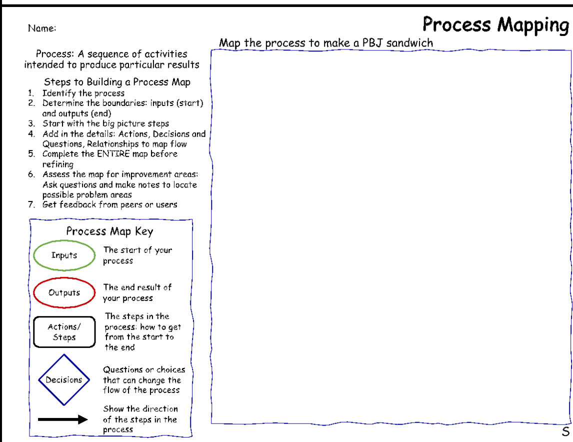
Materials: Print and cut out peanut butter jelly materials, draw morning routine process map

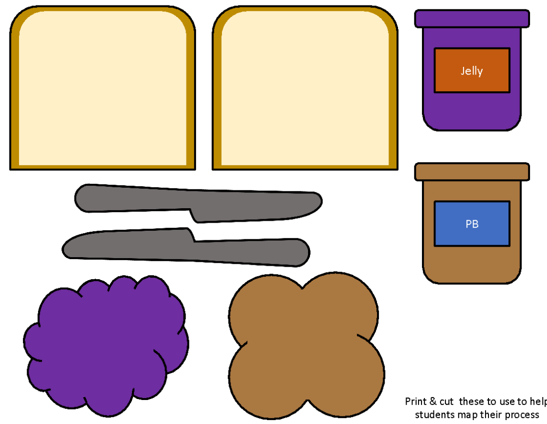
Activity Goals:

- Students learn the basics of process mapping and how it can connect to programming

Activity Procedure:

<table>
<tr>
<td>1. Introduce the Process Map Key and the specifics of what each icon is used for
a. Share morning routine process map – Small Group leader has SIMPLE morning routine process map drawn on anchor chart

Facilitator Shares Slide or directs students to slide in online activity handout workbook OR Draws on Chart Paper</td>
<td>
</td>
</tr>
<tr>
<td>2. Have students in the small group draw a process map on how to make a peanut butter jelly sandwich by identifying the step together. Each student should draw their own map.
a. Remind them of the icebreaker activity and how you had to be specific about the instructions and the actions.
b. Throughout process mapping demonstrate the steps as they are written from the student process maps –</td>
<td>
</td>
</tr>
</table>

they will see any missed steps or errors in their process c. Have students compare process maps and note differences between the steps. **Printout***	
3. Students can use the cutouts to help guide their process map **Printout***	
4. Have student groups share their process map with you and have a discussion on why sequences and clear instructions are important to programming	
5. Optional Extension: Introduce Students to draw.io software to create process maps. It is a very easy program to use and is really helpful for later projects. https://app.diagrams.net/	

3. Create a Micro:bit Name Badge

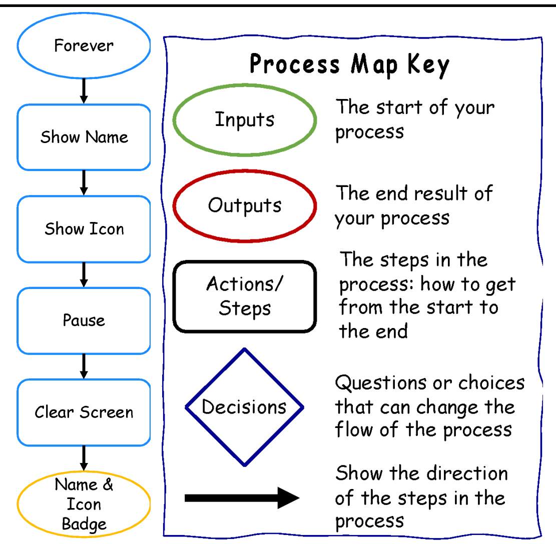
***Estimated Time**: 30 minutes*

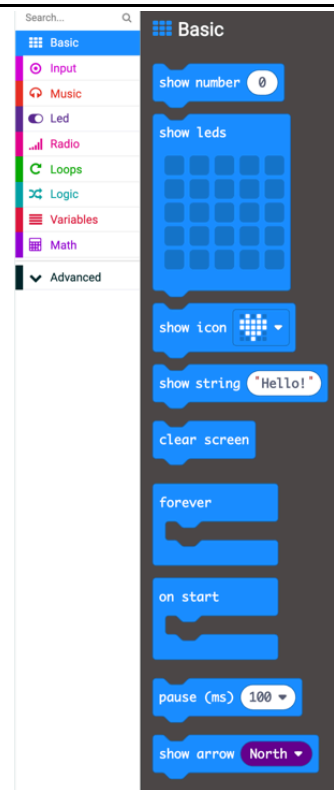
Activity Goals:

- Students learn the basics of Micro:bit: show strings, icons, pause, forever loop
- Students extend programming to include inputs, loops, and basic sounds
- Students will create or analyze process maps for programming the name badge

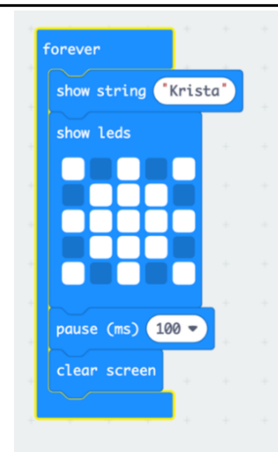
Activity Procedure:

<table>
<tr>
<td>
1. Have students analyze the process map of the name badge code
a. Remind them of the process map key
b. Walk through inputs, outputs, and steps

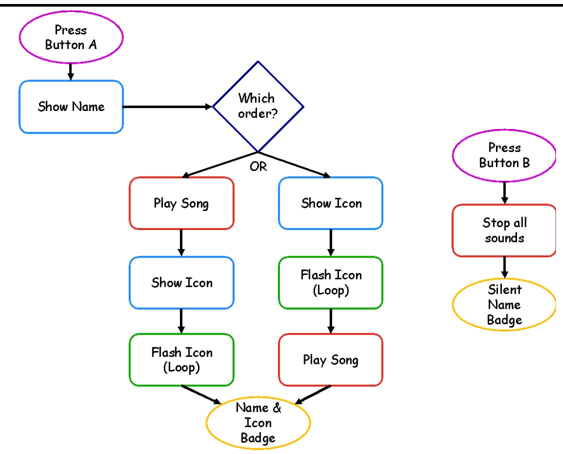
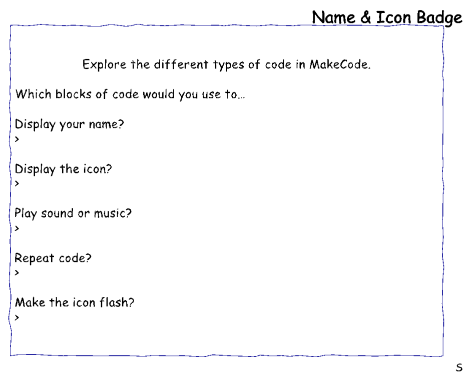
Facilitator Shares Slide or directs students to slide in online activity handout workbook
</td>
<td>


</td>
</tr>
<tr>
<td>
2. Then have students explore “Basic” code blocks on MakeCode to identify which code to use
a. Throughout this step ask students to share which code they believe is used for the name badge
</td>
<td>


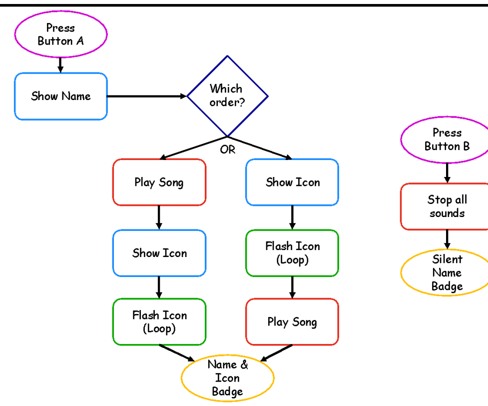
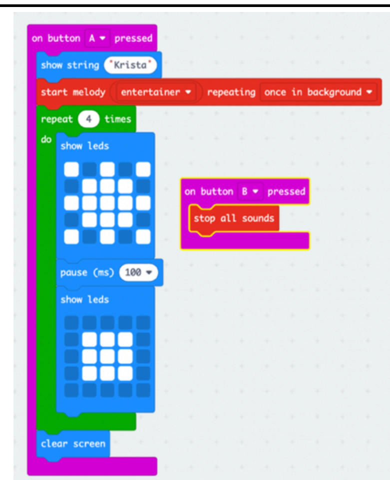
</td>
</tr>
</table>

3. Students sequence the code blocks to develop the name badge using the process map	
4. Students input their name as the "string" and they choose an icon to use in their name badge a. https://microbit.org/projects/make-it-code-it/name-badge/	
5. Show students how to test their code using the simulator in MakeCode to check for errors 6. Then walk students through importing the code onto the Micro:bit	
7. After students complete the basic name badge, challenge them to add more to their code	

<table>
<tr>
<td>8. Share the process map of the upgraded name badge – this includes inputs, loops, and sounds
a. Walk through inputs, outputs, and steps

Facilitator Shares Slide or directs students to slide in online activity handout workbook</td>
<td>
</td>
</tr>
<tr>
<td>9. Have students explore the other types of blocks in MakeCode to identify which blocks to use
a. Explain what loops are and how they function in code
i. Loops allow a portion of code to be repeated. The “Forever” block is a loop that has no end.</td>
<td>


</td>
</tr>
</table>

<table>
<tr>
<td>10. Allow students to choose which sequence of code they would like to use</td>
<td>
</td>
</tr>
<tr>
<td>11. Students program the more advanced name badge with loops to create flashing icons and sounds
12. Check for errors using the simulator
13. Load on to the Micro:bit.</td>
<td>
</td>
</tr>
</table>

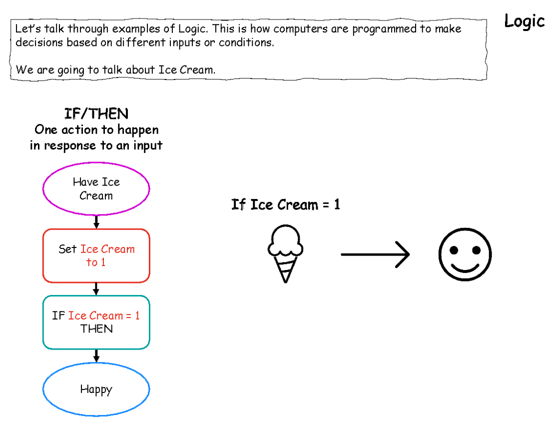
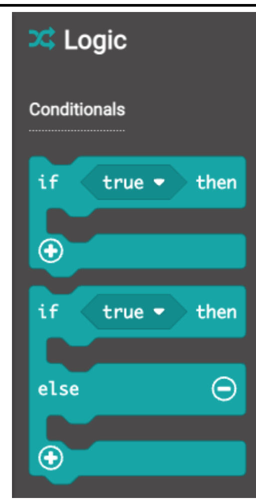
3. Understanding Logic

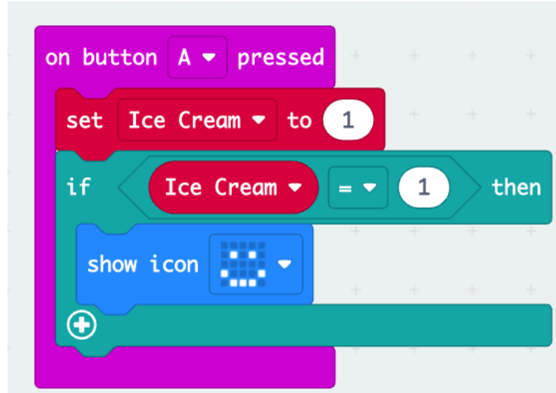
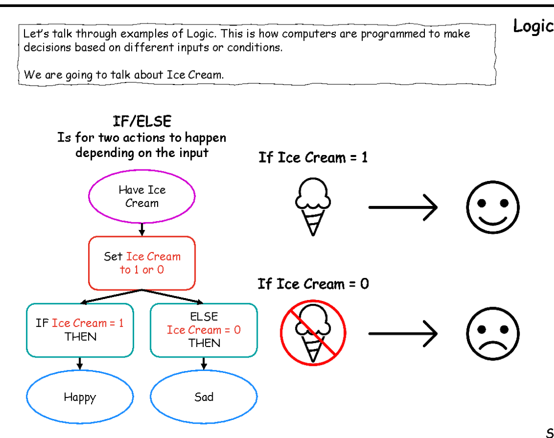
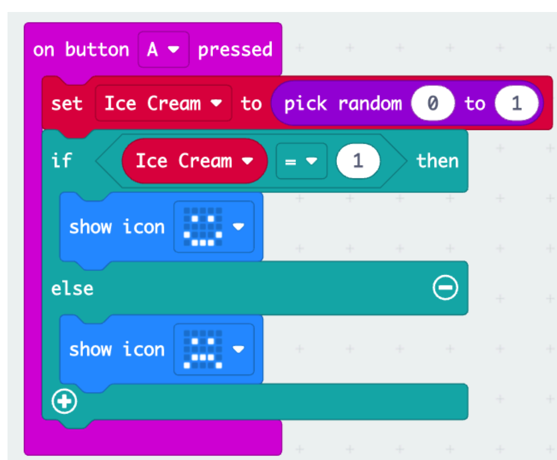
***Estimated Time**: 30 Minutes*

Activity Goals:

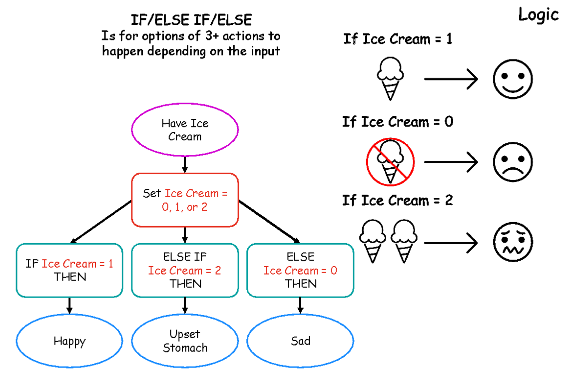
- Build understanding of logic from a programming and computational thinking lens
- Teach students the differences between If/Then statements, If/Else statements, and If/Else If/Else statement and how to use each variation.

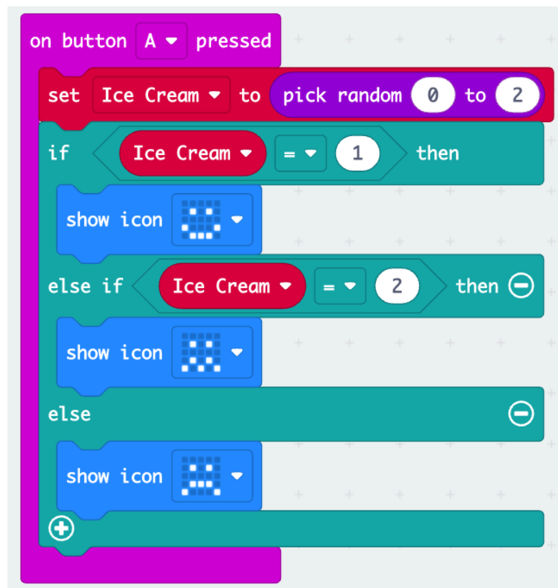
Activity Procedure:

1. Introduce the Idea of Logic and Variables a. Logic is the way a computer can be programmed to make decisions b. A Variable is a placeholder or symbol for a value	
2. Have students analyze the process map of the IF/THEN Code a. Remind them of the process map key b. Walk through inputs, outputs, and steps c. IF/THEN: Is for one action to happen in response to an input d. Talk about Variables i. Ice Cream is the variable in this example because we are setting it equal to different numbers ii. It represents how many ice creams we have **Facilitator Shares Slide and/or directs students to slide in online activity handout workbook**	
3. Have students explore "Logic" and Variable code blocks on MakeCode to identify which code to use a. Throughout this step ask students to share which code they believe is used b. Name the Variable "Ice Cream" and set it equal to 1 c. If needed, remind them that icons are in the "Basic" category	

4. Students sequence the code blocks to program an IF/THEN sequence	
5. Have students test their code using the simulator in MakeCode to check for any errors	
6. Students import the code onto the Micro:bit and see what happens. a. Does it match the process map? b. What happens when Ice Cream = 1?	
7. For IF/ELSE statements: repeat the process steps 2-6 a. Change the variable to “Pick random 0 to 1” using the **Math** blocks i. Introduce the math blocks to the students b. Ask students why the program would need to have two different numbers the variable could be equal to c. Have student talk out how is else works. d. IF/ELSE: Is for two actions to happen depending on the input	 

8. For IF/ELSE IF/ELSE statements: repeat the process steps 2-6
    a. Change the variable to **“Pick random 0 to 2”** using the math blocks
    b. Ask students why the program would need to have three different numbers the variable could be equal to
    c. Ask students to explain how if, else if, and else work – have them talk it out
    d. IF/ELSE IF/ELSE: Is for the option of 3+ actions to happen depending on the input





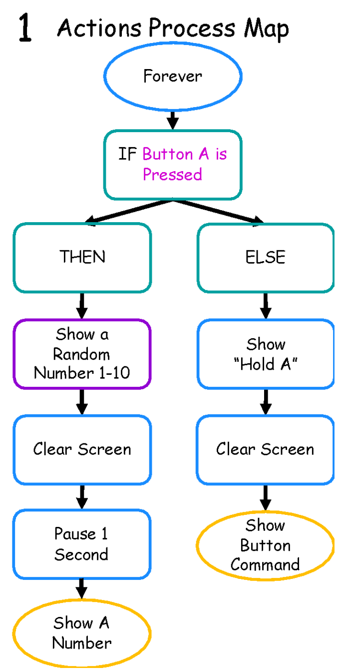
4. Troubleshooting and Debugging

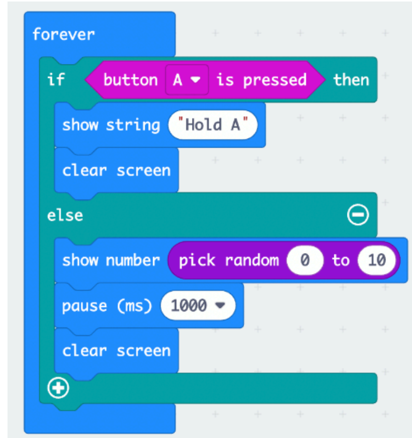
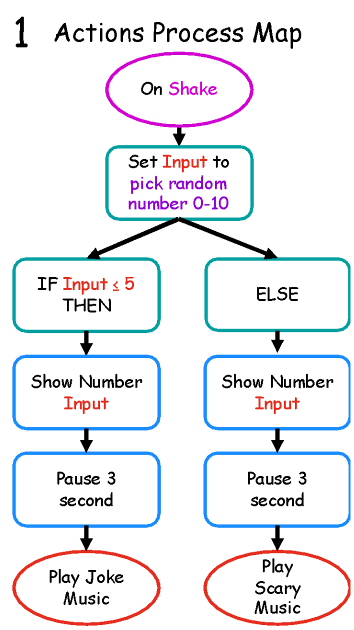
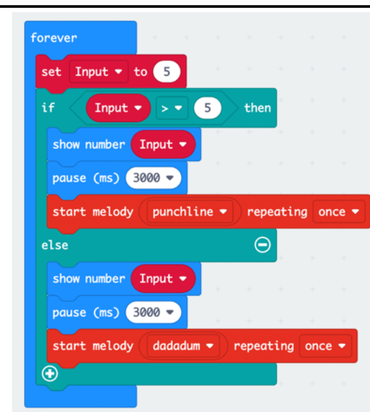
Estimated Time: *25 Minutes*

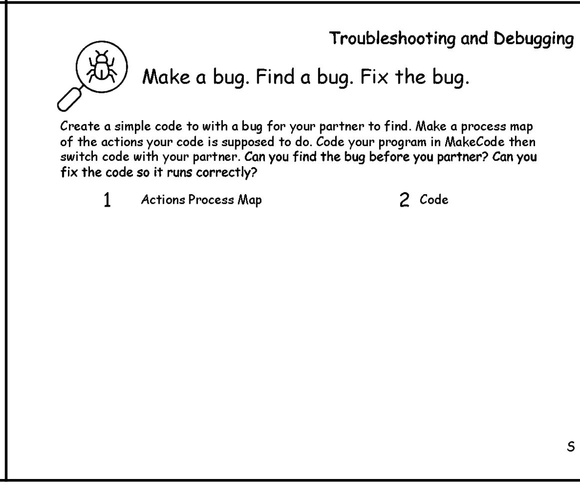
Activity Goals:

- Introduce the concept of troubleshooting "bugs" – errors in the code
- Use computational thinking (decomposition) to deduce errors in code
- Students can identify errors in code that prevent the Micro:bit to perform its intended function
- Students create a process map and simple code which they will intentionally introduce a bug to
- Students will identify bugs in their partner's code

Activity Procedure:

1. Talk about what happens when there is an error in the code a. Did the program provide correct instructions to the computer? b. Errors in code are called "Bugs"	
2. **Bug Challenge 1:** Students are provided with a process map of what the code should do (actions). a. Identify the inputs, outputs, and steps of the process **Facilitator Shares Slide and/or directs students to slide in online activity handout workbook**	

3. Then, students assess the code in on the slide that contains a "bug" a. Have them talk through the sequence of the code that they see b. Have students copy the "bugged" code into Makecode and load it on a Micro:bit	
4. Have students identify the bug, provide reasoning on why they know it is a bug, and a solution to fix the code.	
5. **Bug Challenge 2**: Students are provided with a more **complex** process map of what the code should do (actions). a. Identify the inputs, outputs, and steps of the process **Facilitator Shares Slide and/or directs students to slide in online activity handout workbook**	
6. Then, show students the code in Makecode with **multiple** "bug" a. Have them talk through the sequence of the code that they see b. Have students copy the "bugged" code into Makecode and load it on a Micro:bit	

7. Have students identify the bugs, provide reasoning on why they know it is a bug, and a solution to fix the code.	
8. **Bug Challenge 3**: Create a simple code to with a bug for your partner to find. a. Make a process map of the actions your code is supposed to do. b. Code your program in MakeCode c. Switch code with your partner d. Can you find the bug before you partner? e. Can you fix the code so it runs correctly? **Facilitator Shares Slide and/or directs students to slide in online activity handout workbook**	

Lunch and Recess

60 minutes

5. Problem Solving

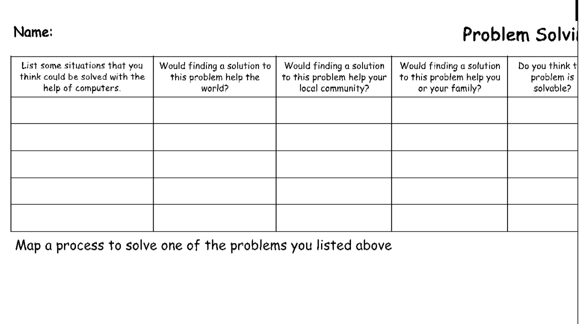
***Estimated Time**: 15 Minutes*

Activity Goals:

- Introduce students to problem solving starting with how to identify a problem, create a plan, and how to find a plausible solution to the problem.
- To help students understand that for some problems they need tools like a computer
- Establish connection between problem solving, computational thinking, and the real-world

Activity Procedure:

<table>
<tr><td>1. Introduce what it means to problem solve as an engineer or scientist in the real-world
a. Connect it to de-bugging done in the previous activity
b. Computational Thinking: describe a problem, identify the important details needed to solve this problem, break the problem down into small, logical steps, use these steps to create a process (algorithm) that solves the problem, and then evaluate this process.</td><td></td></tr>
<tr><td>2. Students will brainstorm with themselves and their small group ideas of problems or situations that could be solved with the help of a computer.
a. The camp has a design focus, so it is important to emphasize why problem solving skills are essential in the workforce

Printout*</td><td>
Name: Problem Solvi
<table><tr><td>List some situations that you think could be solved with the help of computers.</td><td>Would finding a solution to this problem help the world?</td><td>Would finding a solution to this problem help your local community?</td><td>Would finding a solution to this problem help you or your family?</td><td>Do you think t problem is solvable?</td></tr><tr><td></td><td></td><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td><td></td><td></td></tr></table>Map a process to solve one of the problems you listed above</td></tr>
<tr><td>3. Students will then identify the types of problems they listed
a. Global
b. Community
c. Personal/Family</td><td></td></tr>
<tr><td>4. Students will create a process map around a possible way to solve one of the problems they identified.</td><td></td></tr>
</table>

a. Share their process with the small group and evaluate the process	

Activity 3: Coding Applications

Coding Application 1: Creating a Rock, Paper, Scissors Game

https://microbit.org/projects/make-it-code-it/rock-paper-scissors/

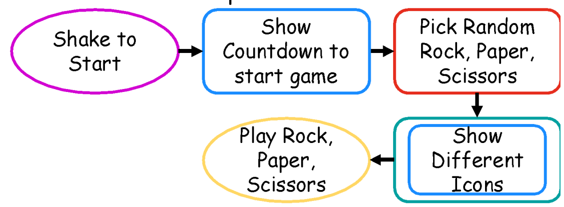
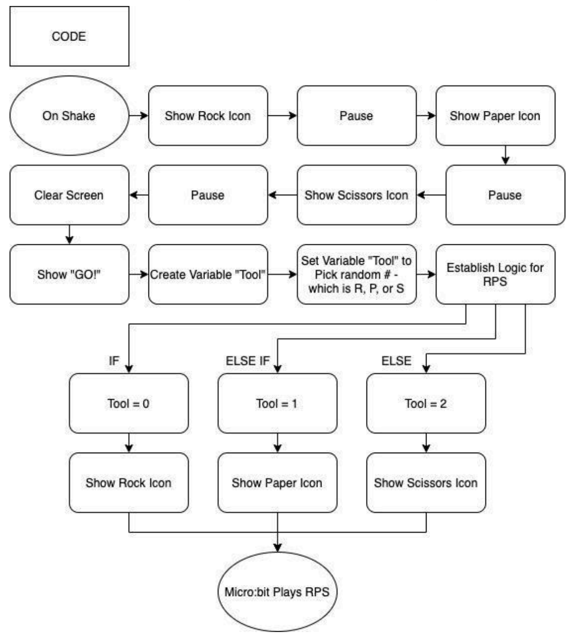
***Estimated Time**: 60 Minutes*

Activity Goals:

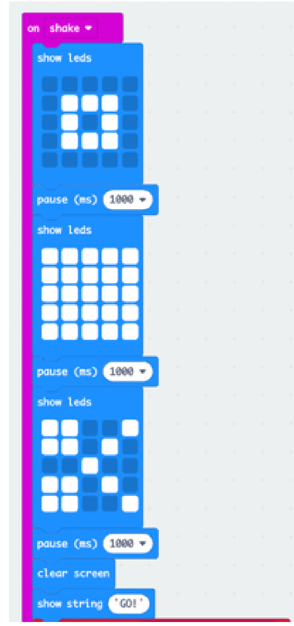
- Students will be able to program a Micro:bit to be able to play a game against it
- Students will apply code they previously learned such as Loops, Logic, Sounds, Variables

Activity Procedure:

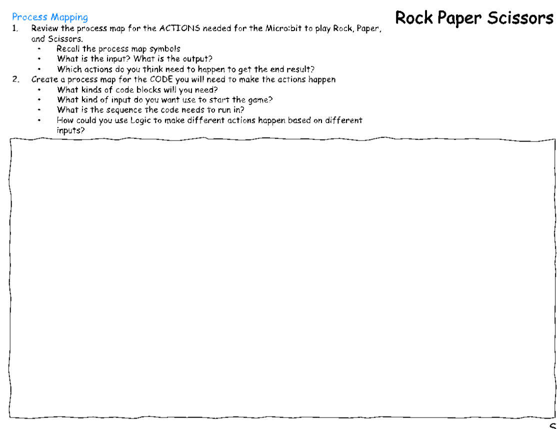
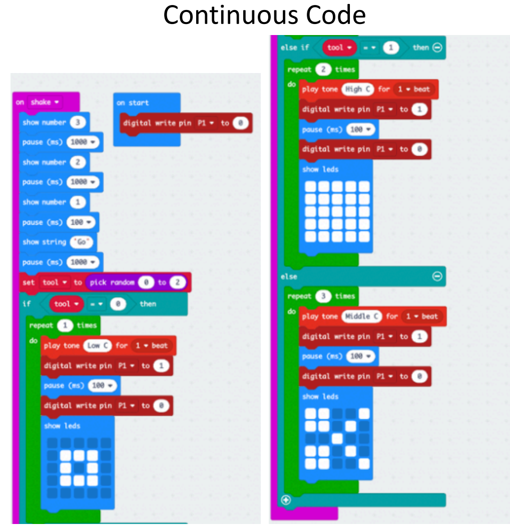
1. Have students play rock, paper, scissors against a partner for 2 rounds	
2. Introduce the activity: Your challenge is to design a rock, paper, scissors game using the Micro:bit. Provide the Program Requirements Below: a. Micro:bot needs a countdown to start playing the game – just like you say "Rock, Paper, Scissors, Shoot!" b. Micro:bit needs to randomly choose between Rock, Paper, or Scissors c. Micro:bit needs to display an icon for Rock, Paper, or Scissors d. Micro:bit needs an input to know when to start the game **Facilitator Shares Slide and/or directs students to slide in online activity handout workbook**	**Program Requirements** 1. Micro:bot needs a countdown to start playing the game - just like you say "Rock, Paper, Scissors, Shoot!" 2. Micro:bit needs to randomly choose between Rock, Paper, or Scissors 3. Micro:bit needs to display an icon for Rock, Paper, or Scissors 4. Micro:bit needs an input to know when to start the game
3. Have the students identify the inputs and outputs for the game a. What will start the game? What is the input? b. What is the end result of the game or the output? 4. Have students think about the actions that have to happen to get from the start to the end result a. Which actions needs to happen for the game to run? b. Actions Process Map is developed *Example Process Map Shown* **Printout***	**Process Mapping** 1. Review the process map for the ACTIONS needed for the Micro:bit to play Rock, Paper, and Scissors. • Recall the process map symbols • What is the input? What is the output? • Which actions do you think need to happen to get the end result? 2. Create a process map for the CODE you will need to make the actions happen • What kinds of code blocks will you need? • What kind of input do you want use to start the game? • What is the sequence the code needs to run in? • How could you use Logic to make different actions happen based on different inputs? Example Process Map

<table>
<tr>
<td></td>
<td>Actions Process Map

</td>
</tr>
<tr>
<td>5. Guide students to take the Actions Process map they built and identify the code blocks they believe they will need. Students will create a CODE process map to match the actions they listed.
a. What kinds of code blocks will you need?
b. What kind of input do you want use to start the game?
c. What is the sequence the code needs to run in?
d. How could you use Logic to make different actions happen based on different inputs?</td>
<td>Example Code Process Map

</td>
</tr>
<tr>
<td>6. Students move to MakeCode editor to build out their code the process mapped
a. Use the MakeCode editor to program the Micro:bit to play rock, paper, scissors
b. Test your code periodically using the simulator or the Mirco:bit to identify any bugs or test your sequence.

Example of Code Shown</td>
<td>Continuous Code</td>
</tr>
</table>

<table>
<tr>
<td></td>
<td>


</td>
</tr>
<tr>
<td>
7. Check each group's programming when they are finished to make sure it follows their process maps and results in a complete game of Rock, Paper, Scissors.
    a. Students then load the program onto their Micro:bits
</td>
<td></td>
</tr>
</table>

8. Each students goes 2 rounds of Rock, Paper, Scissors against their Micro:bit	
9. **Rock, Paper, Scissors Challenge:** Challenge the students to add another element to the code of their game. Have them choose from these options: a. Different number of **flashes** of the Rock, Paper, or Scissor **icons** b. Play **different tones** for Rock, Paper, or Scissors c. Display **different lights** or **flashes of lights** for Rock, Paper, or Scissors using an LED	
10. Have students mark up or add to their process maps to show where their changes will be added in both Actions and Code	
11. Students move to MakeCode editor to build out their code a. Use the MakeCode editor to map out code to play rock, paper, and scissors. b. Test your code periodically using the simulator or the Mirco:bit to identify any bugs or test your sequence. *Example code shown with both LED and sounds/tones* LIGHTS NEED DIGITAL WRITE PIN	Continuous Code 
12. Check each group's programming when they are finished to make sure it follows their process maps and results in a complete game of Rock, Paper, Scissors.	

13. Students can then load it onto their Micro:bit if they have not done so already	
14. Each students goes 2 rounds of Rock, Paper, Scissors against their Micro:bit	

Coding Application 2: Collect Light Intensity Data and Create an Automatic Light

***Estimated Time**: 60 Minutes*

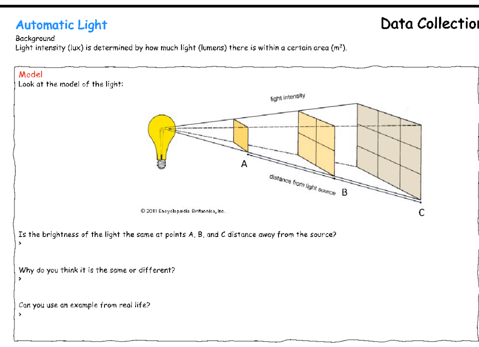
Materials: Flashlights for each group, batteries, it helps to use a toilet paper roll or paper towel roll to concentrate the light from flashlight to microbit, tape to map out distances (2 in, 4 in, 6 in, 8 in, 10 in), measuring tape to mark distance, GROUPS of 3

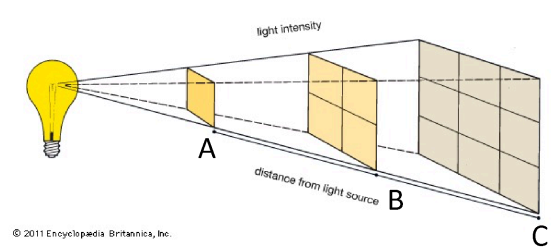
Activity Goals:

- Students will program the Micro:bits to collect light intensity data using the internal photoresist in the Micro:bit and/or program an external photoresist sensor to collect light intensity values.
- Students will learn to collect, plot, and analyze data for trends using a spreadsheet software (google sheets, excel, etc.)
- Student will establish relationships between light intensity and distance using the data collected and the Inverse Square Law for light intensity. $I = 1/d^2$ I= light intensity d = distance from the light source

Activity Procedure:

<table>
<tr><td>1. Introduce the challenge of creating a light that automatically turns on when the brightness reaches a certain level</td><td></td></tr>
<tr><td>2. Elicit student understanding of the concept by showing them the diagram. Have them think about and respond to the following prompts.
a. Is the brightness of the light the same at points A, B, and C distance away from the source?
b. Why do you think it is the same or different?
c. Can you use an example from real life?

Facilitator Shares Slide and/or directs students to slide in online activity handout workbook</td><td>


S


</td></tr>
</table>

<table>
<tr>
<td>

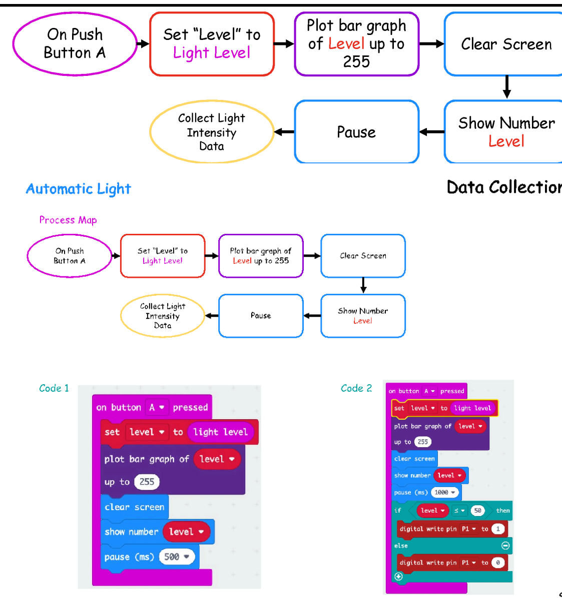
3. Show students the photoresist sensor and explain how it works to measure light intensity
    a. When a photoresistor sensors is exposed to light, the resistance decreases so it become more conductive. We can use this change in resistance to measure the intensity of light.

</td>
<td></td>
</tr>
<tr>
<td>

4. Have students think of the inputs, outputs, and possible steps (actions) that would need to take place in the program

</td>
<td></td>
</tr>
<tr>
<td>

5. Show students the ACTIONS process map to collect and graph data from a light sensor
    a. Have them identify what the inputs, outputs, and steps are in the code provided

</td>
<td>


</td>
</tr>
<tr>
<td>

6. Show students the code to collect and graph data from a light sensor
    a. Have them identify and match the inputs, outputs, and steps from the process map
    b. What does the variable level equal?
    c. What number is going to display on the screen?

</td>
<td>


</td>
</tr>
<tr>
<td>

7. Have students experiment with data collection using the light sensor in the Micro:bit.

</td>
<td></td>
</tr>
</table>

They will EACH take 5 measurements from different distances while aiming directly at the center of the Micro:bit.

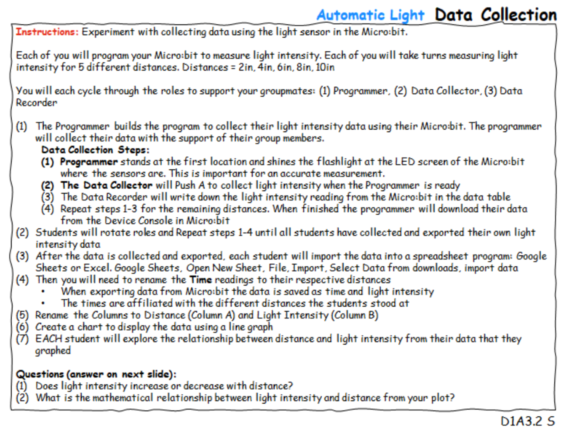
Distances = 2 in, 4 in, 6 in, 8in, 10 in

Each student will cycle through each of the roles to support their groupmates: (1) Programmer, (2) Data Collector, (3) Data Recorder

a. The Programmer builds the program to collect their light intensity data.
b. The programmer will then collect their data with the support of their group members.
c. Programmer stands at the first location and shines the flashlight at the LED screen of the Micro:bit where the sensors are. This is important for an accurate measurement.
d. The Data Collector will Push A to collect light intensity when the Programmer is ready
e. The Data Recorder will write down the light intensity reading from the Micro:bit in the data table
f. Repeat steps C-E for the remaining distances away from the light source
g. When finished the programmer will download their data from the Device Console in Micro:bit
h. Students will rotate roles and Repeat steps A-G until all students have collected and exported their own light intensity data

Facilitator Shares Slide and/or directs students to slide in online activity handout workbook

Printout* Data Table



Automatic Light Data Collection

Instructions: Experiment with collecting data using the light sensor in the Micro:bit.

Each of you will program your Micro:bit to measure light intensity. Each of you will take turns measuring light intensity for 5 different distances. Distances = 2in, 4in, 6in, 8in, 10in

You will each cycle through the roles to support your groupmates: (1) Programmer, (2) Data Collector, (3) Data Recorder

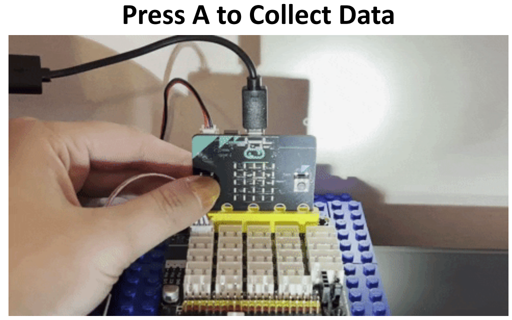
(1) The Programmer builds the program to collect their light intensity data using their Micro:bit. The programmer will collect their data with the support of their group members.
   Data Collection Steps:
   (1) Programmer stands at the first location and shines the flashlight at the LED screen of the Micro:bit where the sensors are. This is important for an accurate measurement.
   (2) The Data Collector will Push A to collect light intensity when the Programmer is ready
   (3) The Data Recorder will write down the light intensity reading from the Micro:bit in the data table
   (4) Repeat steps 1-3 for the remaining distances. When finished the programmer will download their data from the Device Console in Micro:bit
(2) Students will rotate roles and Repeat steps 1-4 until all students have collected and exported their own light intensity data
(3) After the data is collected and exported, each student will import the data into a spreadsheet program: Google Sheets or Excel. Google Sheets, Open New Sheet, File, Import, Select Data from downloads, import data
(4) Then you will need to rename the **Time** readings to their respective distances
   - When exporting data from Micro:bit the data is saved as time and light intensity
   - The times are affiliated with the different distances the students stood at
(5) Rename the Columns to Distance (Column A) and Light Intensity (Column B)
(6) Create a chart to display the data using a line graph
(7) EACH student will explore the relationship between distance and light intensity from their data that they graphed

Questions (answer on next slide):
(1) Does light intensity increase or decrease with distance?
(2) What is the mathematical relationship between light intensity and distance from your plot?

D1A3.2 S

Programmer Builds the Program

Press A to Collect Data



Data Collector Records Values for Programmer

Automatic Light Data Collection

Programmer 1

Distance	Time	Light Intensity
2 inches		
4 inches		
6 inches		
8 inches		
10 inches		

Programmer 2

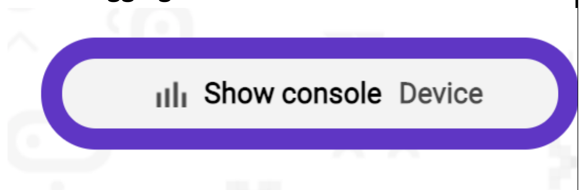
Distance	Time	Light Intensity
2 inches		
4 inches		
6 inches		
8 inches		
10 inches		

Programmer 3

Distance	Time	Light Intensity
2 inches		
4 inches		
6 inches		
8 inches		
10 inches		

D1A3.2 S

Click "Show console Device" to view live data logging. Located under simulator



Use these icons to pause or download data files from "show console device"

8. Each Student will import the data into a spreadsheet program like Google Sheets or Excel
    a. Google Sheets, Open New Sheet, File, Import, Select Data from downloads, import data
9. Rename the Time readings to their respective distances from the Micro:bit
    a. When exporting data from Micro:bit the data is saved as time and light intensity
    b. The times are affiliated with the different distances the students stood at
10. Rename the Columns to Distance (Column A) and Light Intensity (Column B)
11. Create a chart to display the data using a line graph
12. EACH student will explore the relationship between distance and light intensity from their data that they graphed

Raw Data

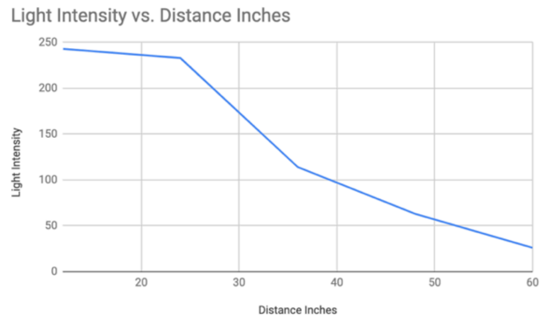
sep=	
time (source1)	Light Intesity
75.44	243
117.65	233
164.575	114
210.051	63
266.122	26

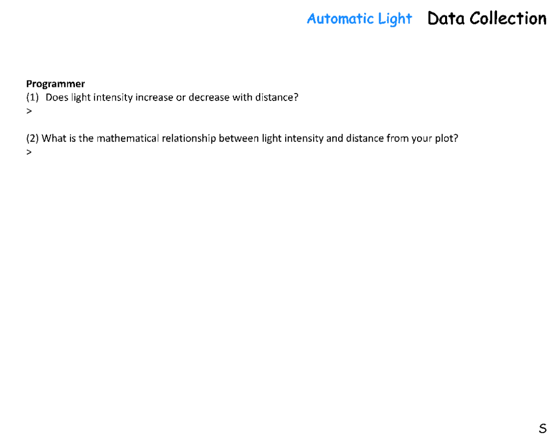
Change Time to related distance

sep=	
Distance Inches	Light Intensity
12	243
24	233
36	114
48	63
60	26

Graph Distance vs Light Intensity

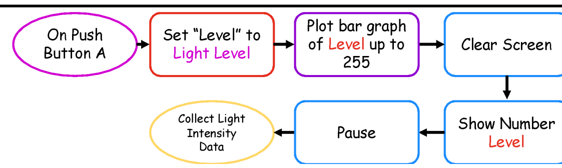
a. Guide students to look at the relationship between the light intensity and the distance away
b. Does light intensity increase or decrease with distance?
c. What is the mathematical relationship between light intensity and distance from your plot?

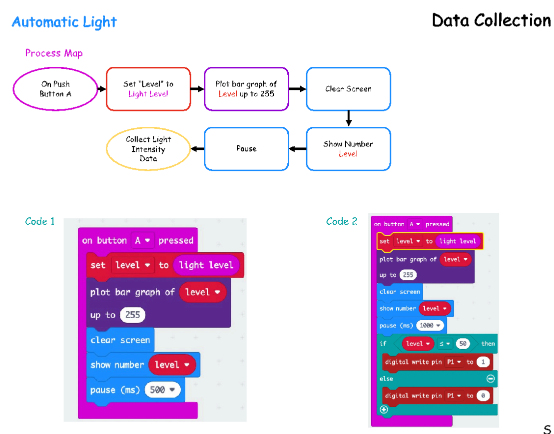


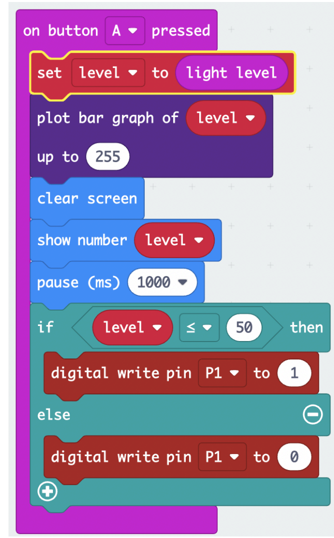


13. NOW create an automatic light that turns on at a certain light intensity or distance from a light source

14. Look at the original process map for the code. What actions would you add and where would you add them to create an automatic light?
15. What could the code and sequence look like for that?





16. Show the students the updated code that includes an automatic light that turns on at a certain threshold. a. Ask the students where they see the changes in ACTIONS and in CODE b. What actions does the new code show when run? c. What tool is Pin P1? d. What do you think happens when P1 = 1? What happens when P1 = 0?	
17. Using your data from the first experiment, estimate what distance you think the light will turn on at: a. What is your light intensity threshold? b. What distance do you think the light will turn on at? 18. Have students repeat the experiment from above with the new code. a. Have them note at what distance and light intensities the LED light turns on at b. Make sure students download the data and graph the results accordingly	
Wrap Up and Sign Out Surveys	

Day 2: Design a Micro:bit Pet, Micro:bit Pet Extension, and Design Challenge

Overview

Activity	Time, minutes
Welcome	30
Design a Micro:bit Pet	280
Lunch and Recess	60
Micro:bit Pet Extension	30
Total	400
Extra Time	20

Materials Needed for Day 2:

Technology	Craft Supplies	Activity Supplies
● Computers ● Micro:bits - 1 per student ● Servo motors ● Dupont Wires (male –female, male – male) ● Alligator Clips ● AAA Batteries ● Micro:bit Stem Kits – sensors, LED lights, motors	● Paper ● Tape ● Markers ● Pencils ● Scissors ● Construction paper ● Paper towel/toilet paper rolls ● Cardboard/Cardstock ● Other craft materials for design projects	● Sticky Notes ● Sticky Easel Pad (optional) ● Rulers ● Student Rewards

Activity 1: Welcome

Estimated Time: 30 Minutes

Activity 2: Design a Micro:bit Pet

https://mgraffin.edublogs.org/2020/06/21/designing-microbit-virtual-pets-monsters-so-many-possibilities/#.YWXHMBDMI56

Estimated Time*: Intro Design Pet Challenge – 10 minutes, Find Partner – 10 minutes, Empathize – 20 minutes, Define – 20 minutes, Ideate/Brainstorm – 30 minutes, Ideate Feedback – 20 minutes, Pet Prototype 1 – 30 minutes, Pet Prototype 1 Feedback – 10 minutes, Pet Prototype 2 – 30 minutes, Pet Prototype 2 Feedback – 10 minutes, Pet Final Design – 30 minutes, Create Adoption Flyer and Flipgrid Advertisement – 60 minutes* ***Total Time = 280 minutes***

Activity Goals:

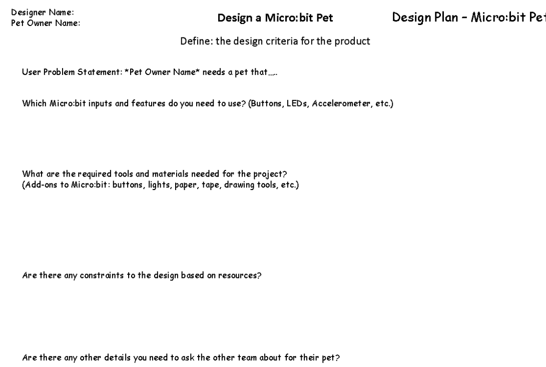
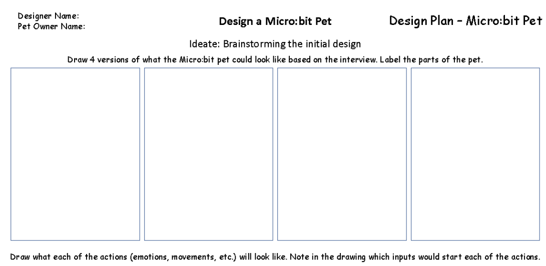
- Students will learn about and experience a design challenge and how the engineering design process is an essential component in completing design challenges
- Students will use the design thinking process: Empathize, Define, Ideate, Prototype, Test, Refine
- Students will learn to identify user needs and how to transform those needs into a design plan
- Students will use process mapping to map out the actions, purpose, and functions of the design
- Students will use and apply previous programming skills

Activity Procedure:

<table>
<tr><td>DAY 1
1. Micro:bit Pet Design Introduction (10 minutes)

Introduce the goal of designing a Micro:bit pet for your partner. There will be an adoption contest at the end of the project.

Facilitator directs students to slide in online activity handout workbook
Printout* Ideate Page, Refinements Page</td><td></td></tr>
<tr><td>2. Find a Partner (10 minutes)
First students need to find a partner that they are going to design a Micro:bit pet for
a. Someone they did not work with yesterday
b. Someone who is on the other side of the room
c. With partner play 2 truths and a lie icebreaker and introduce their partner to the class</td><td></td></tr>
<tr><td>3. Empathize Interview Partner (20 minutes)
Students will introduce themselves to their partners. They will proceed to Interview each other about what they would like in a Micro:bit pet. Then they will switch who gets interviewed and who is interviewing. (15 mins)
a. What kind of pets does your partner want? (an Animal? Insect?)
b. What are the attributes they would like in their pet?
i. Looks:
ii. Size:
iii. Emotions:
iv. Movements:</td><td>
Designer Name:
Pet Owner Name:
Design a Micro:bit Pet
Design Plan – Micro:bit Pet
Empathize: Get to know your product users
Before you start designing a Micro:bit pet, you need to learn what its future owner would like in a robot pet. Interview your partner to learn about the kid of pet they would like to have designed for them.
What kind of pets does your partner want? (an Animal? Insect?)
What are the attributes they would like in their pet?
Looks:
Size:
Emotions:
Movements:
Sounds:
How does the owner want to interact with their pet?</td></tr>
</table>

<table>
<tr>
<td>v. Sounds:
c. How does the owner want to interact with their pet?</td>
<td></td>
</tr>
<tr>
<td>4. Define Design Criteria (20 minutes)
After the interview, each student will define the design criteria: the wants and needs of the user.
a. Create a problem statement of user’s needs. Finish the sentence “Pet owner needs a pet that…”
b. Which Micro:bit inputs and features do you need to use? (Buttons, LEDs, Accelerometer, etc.)
c. What are the required tools and materials needed for the project?
d. (Add-ons to Micro:bit: buttons, lights, paper, tape, drawing tools, etc.)
e. Are there any constraints to the design based on resources?
f. Are there any other details you need to ask the other team about for their pet?</td>
<td>
Designer Name:
Pet Owner Name:
Design a Micro:bit Pet
Design Plan – Micro:bit Pet
Define: the design criteria for the product
User Problem Statement: *Pet Owner Name* needs a pet that.....
Which Micro:bit inputs and features do you need to use? (Buttons, LEDs, Accelerometer, etc.)
What are the required tools and materials needed for the project?
(Add-ons to Micro:bit: buttons, lights, paper, tape, drawing tools, etc.)
Are there any constraints to the design based on resources?
Are there any other details you need to ask the other team about for their pet?</td>
</tr>
<tr>
<td>5. Ideate Micro:bit Pet Designs (30 minutes)
Students then move into the Ideation/Brainstorming stage where they will draw 4 versions of the Micro:bit pet and label the parts of the pet. They will also map out the actions the pet will do – emotions, icons, sounds etc.</td>
<td>
Designer Name:
Pet Owner Name:
Design a Micro:bit Pet
Design Plan – Micro:bit Pet
Ideate: Brainstorming the initial design
Draw 4 versions of what the Micro:bit pet could look like based on the interview. Label the parts of the pet.
Draw what each of the actions (emotions, movements, etc.) will look like. Note in the drawing which inputs would start each of the actions.</td>
</tr>
</table>

6. **Ideate Feedback** (20 minutes)

   Students will approach a group leader and another student for feedback on their design ideas.

   a. Students will approach another group leader to ask for feedback about their four design ideas. They will use the four-square framework as a guide:
      i. What I like about the design?
      ii. What I do not like about the design?
      iii. Questions I have about the design.
      iv. I want the creator to know.
   b. After talking with the group leader, the students will brainstorm on how to refine their design.
   c. Students will approach their partner (the user) to ask for feedback about their four design ideas. They will use the four-square framework as a guide:
      i. What I like about the design?
      ii. What I do not like about the design?
      iii. Questions I have about the design.
      iv. I want the creator to know.
   d. After talking to their partner (the user), the students will brainstorm on how to refine their design.

Designer Name:
Pet Owner Name:
Group Leader Name:

Design a Micro:bit Pet Design Plan – Micro:bit Pet

Test and Refine: Feedback from users to make refinements

Group Leader Checkpoint: Get feedback from another group leader in the room. Make any design changes after the feedback.

What I like about the design	Questions I have about the design
>	>
What I do not like about the design	I want the creator to know…
>	>

For Design Group - Identify Refinements

What I agree with about the feedback:
>

What I disagree with about the feedback:
>

What is going to change about the design in the next version:
>

Designer Name:
Pet Owner Name:
Feedback Partner Name:

Design a Micro:bit Pet Design Plan – Micro:bit Pet

Test and Refine: Feedback from users to make refinements

Ideate User Feedback: Get feedback from your partner (the user). Make any design changes after the feedback.

What I like about the design	Questions I have about the design
>	>
What I do not like about the design	I want the creator to know…
>	>

For Design Group - Identify Refinements

What I agree with about the feedback:
>

What I disagree with about the feedback:
>

What is going to change about the design in the next version:
>

7. **Prototype 1** (30 minutes)

   The students will review the feedback from the group leader and their partner from the previous and use it to start building their first prototype for their Micro:bit pet.

   a. The students will start by creating a process map of the actions for their Micro:bit.
   b. The students will identify the codes they need for the actions and the sequence of codes.

Design a Micro:bit Pet Design Plan – Micro:bit Pet

Prototype: Building and testing different designs

PROTOTYPE 1

Instructions:
1. Grab a blank sheet of paper and label it: **PROTOTYPE 1** with your name at the top
2. Create 2 process maps for the (1) ACTIONS of the Micro:bit pet and the (2) CODE you are going to use to create the pet. Label the process maps.
3. Begin building a prototype of your Micro:bit pet based on the design ideas you came up with in the ideate stage.
4. Add any additional resources you may have used to the DEFINE page of the design plan.
5. Answer the questions on the **REFINEMENT** page to track the changes in your design

Example Paper

Name Prototype 1

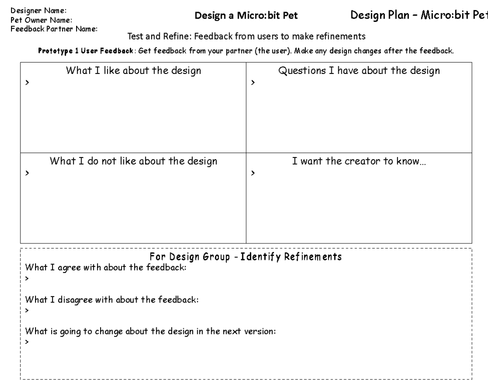
Actions Process Map

Code Process Map

c. Once they have identified the codes and sequence, they will use MakeCode editor to program their Micro:bit.

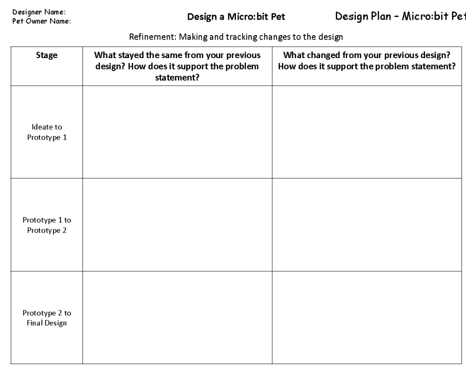
8. **Prototype 1 Feedback** (10 minutes)
Students will approach their partner (the user) to ask for feedback about their Micro:bit pet prototype -1. They will provide feedback simultaneously to each other. They will use the four-square framework as a guide:
    a. What I like about the design?
    b. What I do not like about the design?
    c. Questions I have about the design.
    d. I want the creator to know.
    e. After talking to their partner (the user), the students will brainstorm on how to refine their design.

Designer will reflect on the refinements they made from the ideate to the prototype stage. They will capture their thoughts on using the refinement chart.


Designer Name:
Pet Owner Name:
Feedback Partner Name:
Design a Micro:bit Pet Design Plan – Micro:bit Pet
Test and Refine: Feedback from users to make refinements
Prototype 1 User Feedback: Get feedback from your partner (the user). Make any design changes after the feedback.

What I like about the design	Questions I have about the design
›	›
What I do not like about the design	I want the creator to know…
›	›

For Design Group - Identify Refinements
What I agree with about the feedback:
›
What I disagree with about the feedback:
›
What is going to change about the design in the next version:
›

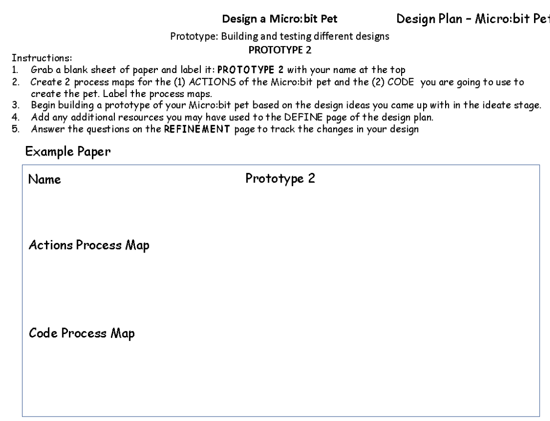

Designer Name:
Pet Owner Name:
Design a Micro:bit Pet Design Plan – Micro:bit Pet
Refinement: Making and tracking changes to the design

Stage	What stayed the same from your previous design? How does it support the problem statement?	What changed from your previous design? How does it support the problem statement?
Ideate to Prototype 1		
Prototype 1 to Prototype 2		
Prototype 2 to Final Design		

9. **Prototype 2** (30 minutes)
The students will review the feedback from their partner and use it to start building their second prototype for their Micro:bit pet.
    a. The students will modify their process map of the actions for their Micro:bit based on the feedback they received.
    b. The students will modify the codes they need for the actions and the sequence of codes.
    c. Once they have identified the codes and sequence, they will use MakeCode editor to program their Micro:bit.

Lunch and Recess

Lunch and Recess (60 minutes)
Time can be adjusted as needed.


Design a Micro:bit Pet Design Plan – Micro:bit Pet
Prototype: Building and testing different designs
PROTOTYPE 2
Instructions:
1. Grab a blank sheet of paper and label it: **PROTOTYPE 2** with your name at the top
2. Create 2 process maps for the (1) ACTIONS of the Micro:bit pet and the (2) CODE you are going to use to create the pet. Label the process maps.
3. Begin building a prototype of your Micro:bit pet based on the design ideas you came up with in the ideate stage.
4. Add any additional resources you may have used to the DEFINE page of the design plan.
5. Answer the questions on the **REFINEMENT** page to track the changes in your design

Example Paper
Name Prototype 2
Actions Process Map
Code Process Map

10. **Prototype 2 Feedback** (10 minutes)
    Students will approach their partner (the user) to ask for feedback about their Micro:bit pet prototype -1. They will provide feedback simultaneously to each other. They will use the four-square framework as a guide:
    a. What I like about the design?
    b. What I do not like about the design?
    c. Questions I have about the design.
    d. I want the creator to know.
    e. After talking to their partner (the user), the students will brainstorm on how to refine their design.

Designer will reflect on the refinements they made from the ideate to the prototype stage. They will capture their thoughts on using the refinement chart.

Designer Name:
Pet Owner Name:
Feedback Partner Name:

Design a Micro:bit Pet Design Plan – Micro:bit Pet

Test and Refine: Feedback from users to make refinements

Prototype 2 User Feedback: Get feedback from your partner (the user). Make any design changes after the feedback.

What I like about the design	Questions I have about the design
›	›
What I do not like about the design	**I want the creator to know…**
›	›

For Design Group - Identify Refinements
What I agree with about the feedback:
›
What I disagree with about the feedback:
›
What is going to change about the design in the next version:
›

Designer Name:
Pet Owner Name:

Design a Micro:bit Pet Design Plan – Micro:bit Pet

Refinement: Making and tracking changes to the design

Stage	What stayed the same from your previous design? How does it support the problem statement?	What changed from your previous design? How does it support the problem statement?
Ideate to Prototype 1		
Prototype 1 to Prototype 2		
Prototype 2 to Final Design		

11. **Final Design** (30 minutes)
    The students will review the feedback from their partner and use it to start building their final design for their Micro:bit pet.
    a. The students will modify their process map of the actions for their Micro:bit based on the feedback they received.
    b. The students will modify the codes they need for the actions and the sequence of codes.
    c. Once they have identified the codes and sequence, they will use MakeCode editor to program their Micro:bit.

Design a Micro:bit Pet Design Plan – Micro:bit Pet

Prototype: Building and testing different designs
Final Design

Instructions:
1. Grab a blank sheet of paper and label it: **FINAL DESIGN** with your name at the top
2. Create 2 process maps for the (1) ACTIONS of the Micro:bit pet and the (2) CODE you are going to use to create the pet. Label the process maps.
3. Begin building a prototype of your Micro:bit pet based on the design ideas you came up with in the ideate stage.
4. Add any additional resources you may have used to the DEFINE page of the design plan.
5. Answer the questions on the **REFINEMENT** page to track the changes in your design

Example Paper

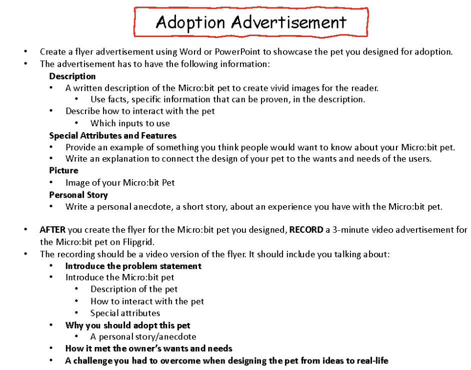
Name Final Design

Actions Process Map

Code Process Map

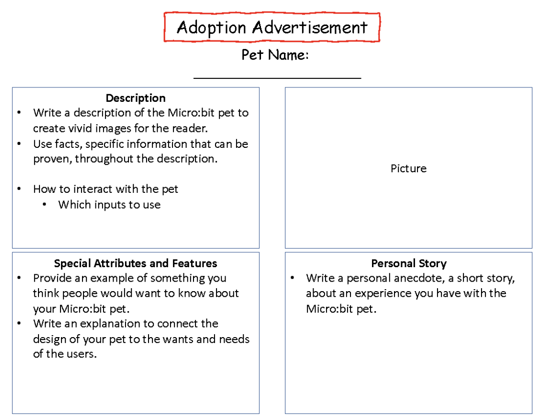
12. Presentation: Adoption Advertisement
(60 minutes)
Students will create a flyer using Word or PowerPoint to get their pet adopted. They will use the four-point frame as a guide in creating the flyer. In the flyer, the students will include the following:

a. Description of their pet and instructions on people can interact with the pet.
b. Special attributes and features pf the pet they want to showcase.
c. Picture of their pet.
d. Personal story about their experience with their pet.



13. Students will then present their advertisement to the class for the Micro:bit Pet they designed.

a. The presentation should include you talking about:
b. **Introduce the problem statement**
c. Introduce the Micro:bit pet
d. Description of the pet
e. How to interact with the pet
f. Special attributes
g. **Why you should adopt this pet**
h. A personal story/anecdote
i. **How it met the owner's wants and needs**
j. **A challenge you had to overcome when designing the pet from ideas to real-life**



14. Students will listen to student presentations.

a. At the end students will have a "pet parade" and put their pets at the front of the class.
b. Students will each get 3 sticky notes to vote for their 3 favorite pets.

If available, Rewards will be given for the top 3 pets.

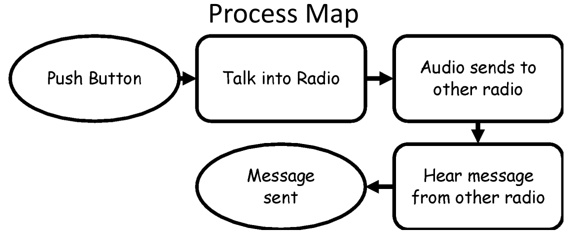
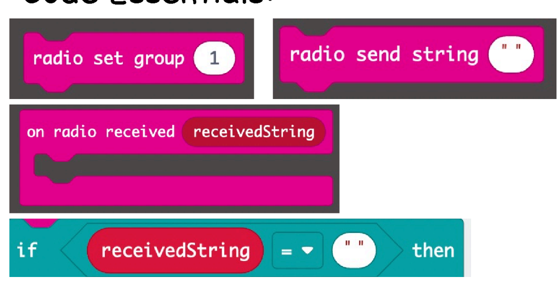
Activity 3: Design a Micro:bit Pet – Extension

***Estimated Time**: Introduction – 5 minutes, Choose and do extension - 25 mins. **Total time = 30 minutes**.*

Activity Goals:

- Allow students more time to work with each other before going into design teams
- Provide students the opportunity to explore more advanced programming in Micro:bit: Bluetooth communication, movement with servo motors, and reactions to sensors

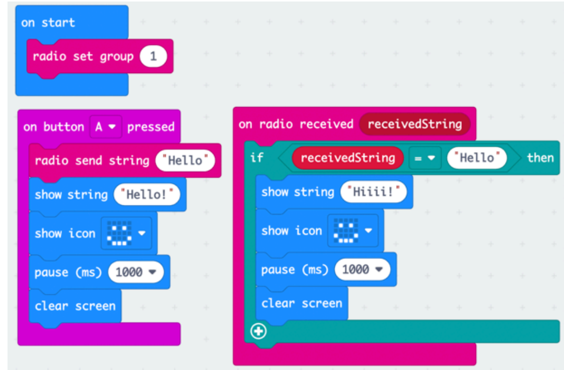
Activity Procedure:

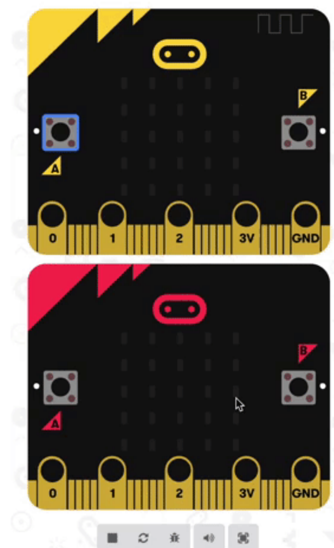
Extension Activity Introduction (5 minutes) Welcome students to the project and share with them that we will be learning 4 new tools for Micro:bit. The goal for this mini project is to explore new programs and bring different skills to their future design teams. The new programs are: a. Bluetooth communication b. Movement with servo motors c. Reactions to sensors (2 types) Students identify which out of the three makes most sense with their Micro:bit pet design. They will then learn how to program the new element. (20 mins)	
Bluetooth Communication ● Have students brainstorm about the inputs, outputs, and steps a walkie talkie uses. ○ This should look like: Push button to talk, sends audio, hear message ● Guide students to think about how the Micro:bits could talk to each other through signals like the walkie talkies ○ Create a process map of the actions for the program ● Show students the essential code blocks under "Radio" and let them talk through what each block may do ● Have students set up the basic code sequence in MakeCode and have them use it on the simulator and between each other. ● ***Make sure each pair of students is on a different radio number*** ○ Have them talk through what expected to see and what they saw between the two Micro:bits	Process Map  Code Essentials: 

- Invite the students to expand on that code with a partner to have their pets send and receive different emotions when different inputs are triggered.
    - Need to have matching radio numbers
    - Need to send “strings” as inputs for different actions
    - Process map your actions and codes
    - Add loops and logic to support your actions

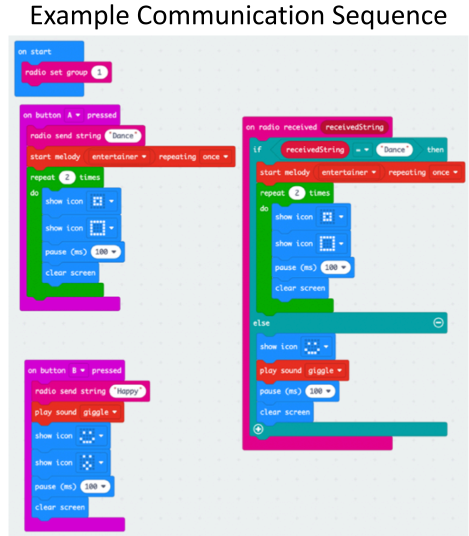
*Example code on next page

Basic Radio Sequence





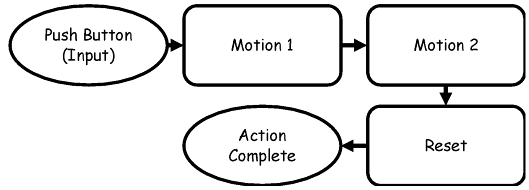
Example Communication Sequence



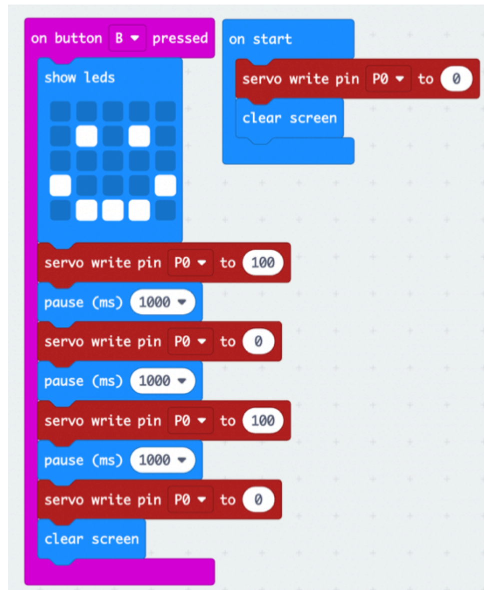
Movement with Servo Motors

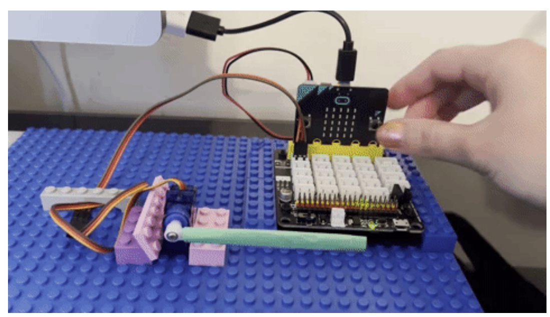
- Have students brainstorm about the inputs, outputs, and steps would be to program movement
  - This should look like: input, motion 1, motion 2, reset, output
- Show students the essential “Servo Write” code blocks under “Pins” and let them talk through what each block may do
  - Before we have only used 0 and 1 to control a device on a pin
  - What could the numbers in “Servo Write to ###” mean?
    - Location to move to
- Have students set up the basic code sequence in MakeCode and have them try it out on the Micro:bit
  - Ensure the pin the connect the Servo to is the correct pin
  - Show students how to connect the servo to the board using the male to female cables. Explain that they need to match the connection to the connection on the board – ie. ground to ground
- Then invite the students to expand on that code to have their pets do different movements with different triggers
- Code could look like something below
  - When a button on P1 is pressed, the show surprised face, wave and blink the LED

Process Map

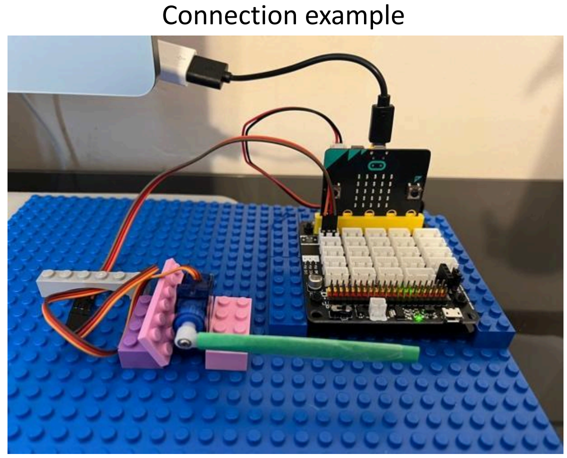


Servo Wave

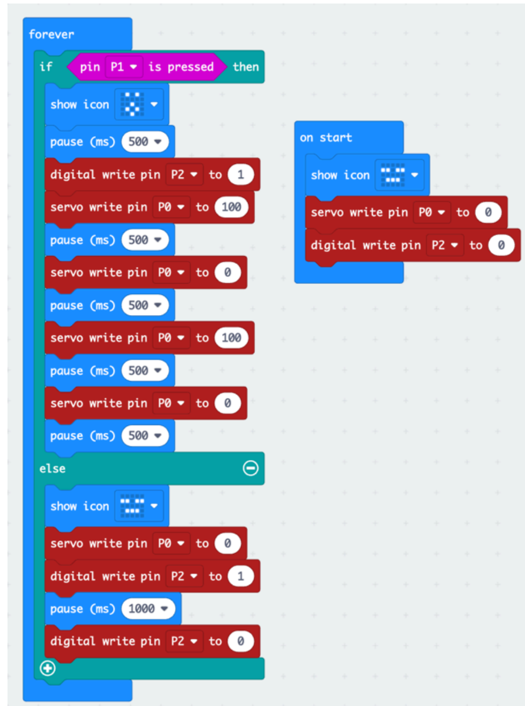




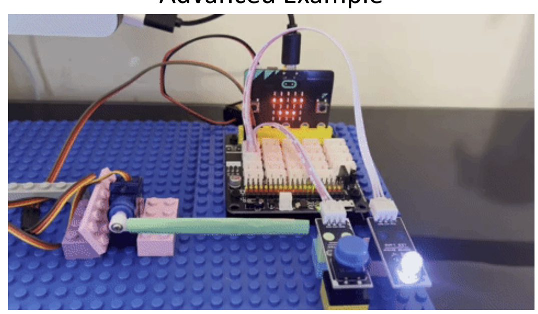
Connection example



Example Advanced Code



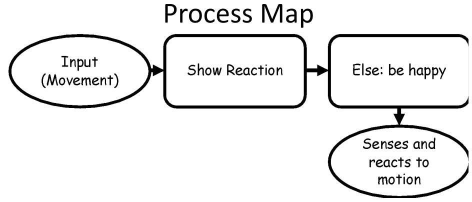
Advanced Example



Reaction to Sensors - Accelerometer

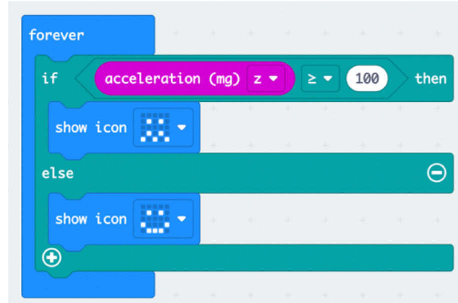
- Have students think about what they do when they hear a loud noise or how they feel on a drop when they ride a rollercoaster
- Mirco:bits can react to the world around them like we can react to things
    - What are some types of sensors that we can use with Micro:bit?
    - What senses do we as humans have?
- Students analyze the process map about using a sensor.
    - How is this like or different than the codes they have worked with today?

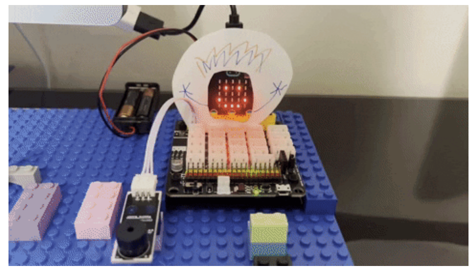
Process Map



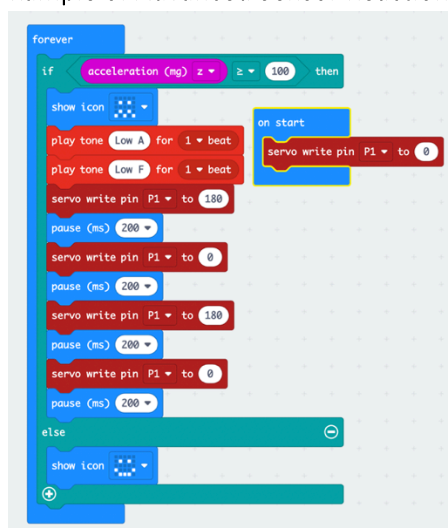
Example Basic Code for Accelerometer Sensor

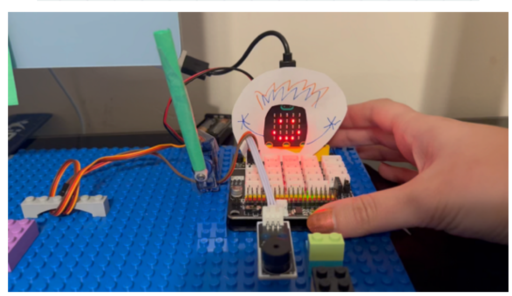
- What is the input? What is the output?
- This sensor is built into the Micro:bit and detects when the Micro:bit experiences different types of motion.
- Have students replicate the example of Basic code in their MakeCode editors.
- Have them break down the actions and reactions they see in the Micro:bit.
- Invite students to expand on the code to make a unique reaction to the Micro:bit pet and the accelerometer.
    - So many options on using the accelerometer: can set acceleration threshold, can use shake, or direction of movement, can use direction Micro:bit is facing
    - Have students add in sounds and other features to the motion. They can use more logic to create different reactions





Example of Advanced Sensor Reactions



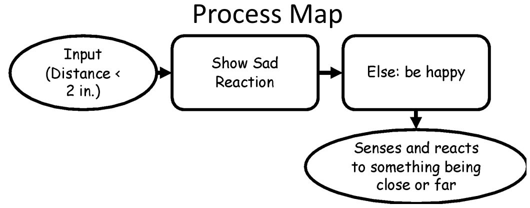


Reaction to Sensors – Ultrasonic Detector

ONLY IF YOU HAVE THE STEM KIT BOARD

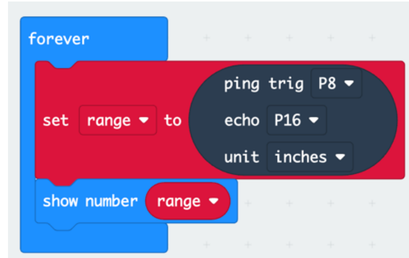
- Have students think about what they do when they hear a loud noise or how they feel on a drop when they ride a rollercoaster

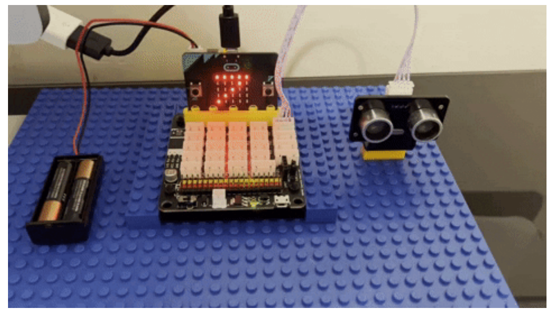
Process Map



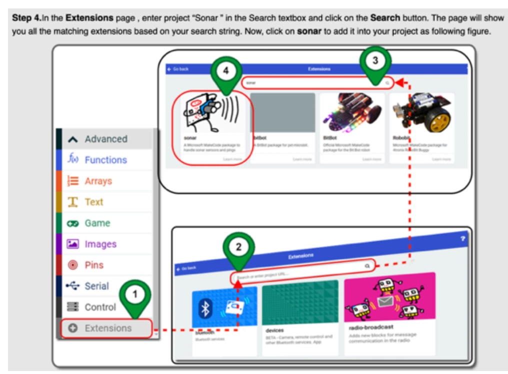
- Mirco:bits can react to the world around them like we can react to things
    - What are some types of sensors that we can use with Micro:bit?
    - What senses do we as humans have?
- Students analyze the process map about using a sensor.
    - How is this like or different than the codes they have worked with today?
    - What is the input? What is the output?
- This sensor is external to the Micro:bit and detects distance from the sensor.
- Have students replicate the example of Basic code in their MakeCode editors.
    - They will need to add the "Sonar" extension from Micro:bit. Click extensions and search sonar. Then choose the first option.
- Have them break down the actions and reactions they see in the Micro:bit.
- Invite students to expand on the code to make a unique reaction to the Micro:bit pet and the ultrasonic detector.
    - So many options on using the ultrasonic detector: can set distance threshold to trigger different emotions, sounds, or movements.
    - Have students add in sounds and other features to the motion. They can use more logic to create different reactions

Basic Ultrasonic Detector Code

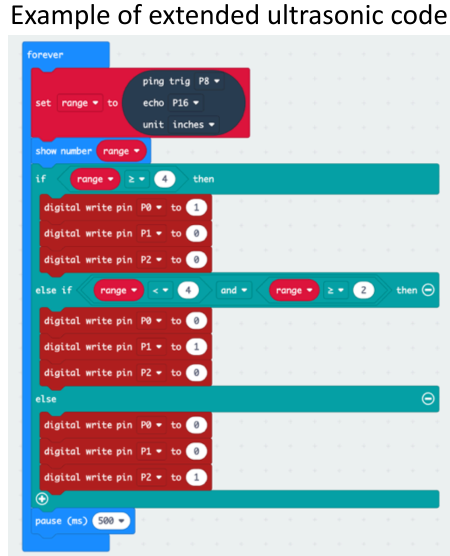


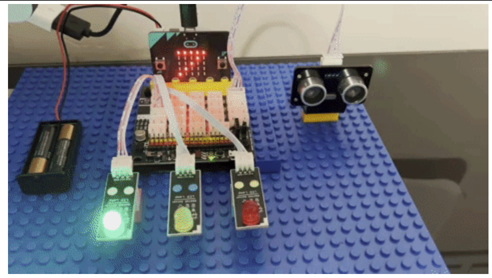


How to add Sonar Extension



Example of extended ultrasonic code



	
Wrap Up and Sign Out Surveys	

Day 3: Design Challenge –

Overview

Activity	Time, minutes
Welcome and birthday icebreaker	30
Design Challenge (Intro to Design Challenge, Mentor Presentations, Create teams, Assign roles, Start Kanban Boards, Interview, Define, Ideate, Feedback, Prototype 1, Feedback, Refine, Prepare to Present, Update Kanban boards)	330
Total (including +45 min for lunch)	405
Extra Time	15

Materials Needed for Day 3:

Technology	Craft Supplies	Activity Supplies
• Computers • Micro:bits - 1 per student • Servo motors • Dupont Wires (male –female, male – male) • Alligator Clips • AAA Batteries • Micro:bit Stem Kits – sensors, LED lights, motors	• Paper • Tape • Markers • Pencils • Scissors • Construction paper • Paper towel/toilet paper rolls • Cardboard/Cardstock • Other craft materials for design projects	• Sticky Notes • Sticky Easel Pad (optional) • Rulers • Student Rewards

Activity 1: Welcome and Icebreaker

Estimated Time*: Welcome – 15 minutes, Birthday Icebreaker – 15 minutes* ***Total = 30 minutes***

- Sort by birthday dates (day of the month) without talking
    - Introduce yourself to your neighbors

Activity 2: Design Challenge – Introduction, Mentor Presentations, Design Teams, Roles, Empathize, Define, Ideate, Prototype 1, Feedback

Estimated Time*: Introduction to Design Challenge – 10 minutes, Mentor Presentations – 30 minutes, Create design teams - 10 minutes, Assign roles and norms – 10 minutes, Kanbam Boards – 10 minutes, Interview (Empathize) – 30 minutes, Define design criteria and needs – 15 minutes, Ideate – 35 minutes, Prepare to present ideate- 20 minutes, Ideate feedback – 15 minutes, Prototype 1 – 40 minutes, Prototype 1 Feedback – 20 minutes, Refine – 10 minutes, Prototype 2 – 40 minutes, Prototype 2 feedback – 15 minutes, Progress Check and gallery walk – 20 minutes* ***Total time = 330 minutes***

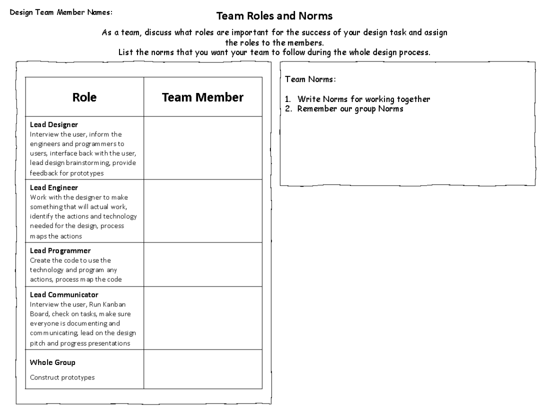
Activity Goals:

- Student will learn about and experience a design challenge and how the engineering design process is an essential component in completing design challenges.
- Students will start the Empathize, Define and Ideate stages of the engineering design process while learning how to use interview to gather information from users (as part of empathize), teamwork, project management (Kanban board).
- Student will learn about and experience a design challenge and how the engineering design process is an essential component in completing design challenges.
- Students will go through the cycle of engineering design process with the guidance of a mentor.
- Students will learn from mentors experiences and stories.
- Students will experience a rapid prototyping and testing challenge

Activity Procedure:

1. **Design Challenge Introduction** (10 minutes) You will be in teams of 2-3 and your task is to design a product for specific users. This is a multiday challenge where your team will go through the Engineering Design process stages. There will be daily whole-group check-in presentations and a final presentation at the end of the week. a. Introduce the Design Theme(s) for the camp b. Industry Mentors will provide design challenges for students
2. Challenge Presentations (30 minutes) Each mentor or teacher will talk about their experiences with computer science projects, engineering design and project management. The goal is to inspire and motivate the students. This is when the technical design challenges will be described to the students.
3. **Student Team Creation** (10 minutes) First students need to form design teams of four a. Teams could be formed randomly by asking students to count and group them according to their number. b. Student groups choose which design challenge they want to work on

4. **Team roles and norms** (10 minutes)
Once the teams are formed, students will introduce themselves to their team. They will then identify essential roles for successful teamwork and come up with an agreed list of norms. **Note students will have more than 1 role since there are groups of 2 –3.** Suggested roles are:
    a. Lead Designer
        i. Interview and interface with the user, inform the others of users input, lead design ideation, provide prototype feedback
    b. Lead Engineer
        i. Work with the designer to make something that will actually work, identify and process map the actions and technology needed for the design
    c. Lead Programmer
        i. Create the code to program the actions of the technology, process map the code, work with Engineer to make something that actually works
    d. Lead Communicator
        i. Run Kanban Board, check on tasks, make sure the team is documenting and communicating, lead progress presentations and design pitch


Design Team Member Names:

Team Roles and Norms

As a team, discuss what roles are important for the success of your design task and assign the roles to the members.
List the norms that you want your team to follow during the whole design process.

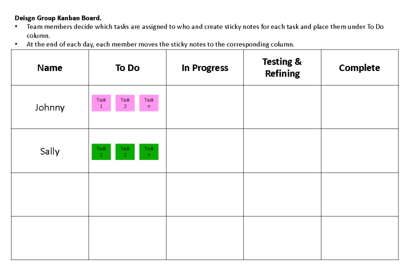
Role	Team Member
Lead Designer Interview the user, inform the engineers and programmers to users, interface back with the user, lead design brainstorming, provide feedback for prototypes	
Lead Engineer Work with the designer to make something that will actual work, identify the actions and technology needed for the design, process maps the actions	
Lead Programmer Create the code to use the technology and program any actions, process map the code	
Lead Communicator Interview the user, Run Kanban Board, check on tasks, make sure everyone is documenting and communicating, lead on the design pitch and progress presentations	
Whole Group Construct prototypes	

Team Norms:

1. Write Norms for working together
2. Remember our group Norms

Whole Group: Construct Prototypes

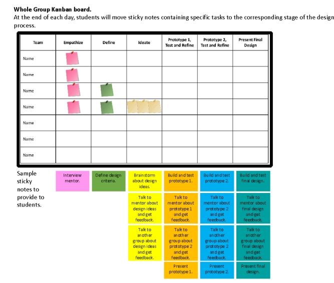
5. **Kanban boards** (10 minutes)
Now that the students have presented their four design ideas and received feedback from the mentor and other team, they can start planning for the next steps.
    a. The teams will identify tasks for each team member, write these tasks on sticky notes and post them under the To-do column on their team's Kanban board, which will be provided by the facilitators.
    b. The facilitators will also present the whole group's Kanban board and remind


Deisgn Group Kanban Board.
- Team members decide which tasks are assigned to who and create sticky notes for each task and place them under To Do column.
- At the end of each day, each member moves the sticky notes to the corresponding column.

Name	To Do	In Progress	Testing & Refining	Complete
Johnny	Task 1, Task 2, Task n			
Sally	Task 1, Task 2, Task n			

each team to update both boards at the end of each day.

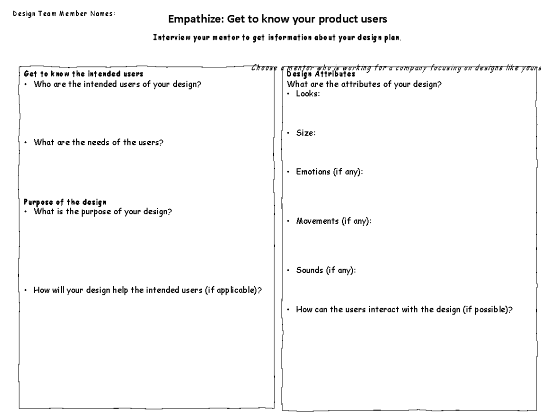
c.



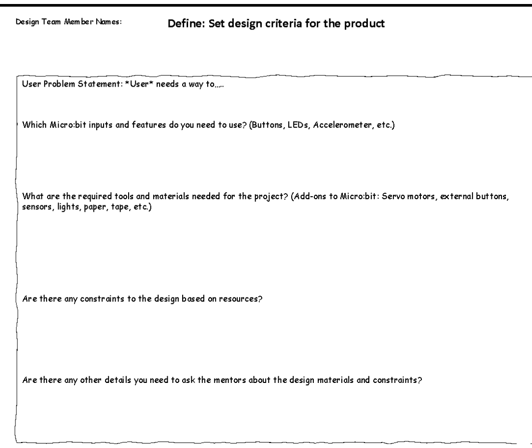
6. **Interview mentor (Empathize)** (30 minutes) Students will then proceed to interview a mentor to get to know more information about the users of the design and about the design needs.
    a. Who are the intended users of your design?
    b. What are the needs of the users?
    c. What is the purpose of the design?
    d. How will your design help the intended users (if applicable)?
    e. What are the attributes of your design?
        i. Looks
        ii. Size
        iii. Emotions (if any)
        iv. Movements (if any)
        v. Sounds (if any)

How can the users interact with their design? (if possible)

** Note if mentor is not available, students will use the information from the design challenge slides to get this information. Internet research can also be used at this stage.



7. **Define design criteria and needs** (15 minutes) After the interview, the design team will define the design criteria: the wants and needs of the users.
    f. User Problem Statement: *User* needs a way to…..
    g. Which Micro:bit inputs and features do you need to use? (Buttons, LEDs, Accelerometer, etc.)
    h. What are the required tools and materials needed for the project?



i. (Add-ons to Micro:bit: buttons, lights, paper, tape, drawing tools, etc.)

i. Are there any constraints to the design based on resources?

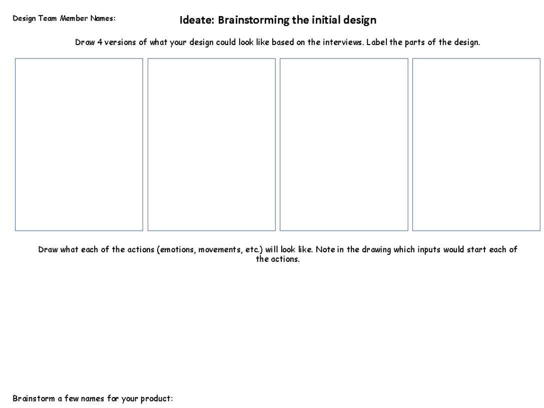
Are there any other details you need to ask the mentor about the design?

8. **Ideate** (35 minutes)
   Student teams then move into the Ideation/Brainstorming stage where they will draw 4 versions of the design and label the parts of the each design. They will also map out the actions the design will do (if applicable) – icons, sounds, movements, sensors, etc. They will also brainstorm few names for their design.

Lunch and Recess

9. Prepare to Present Ideate (20 minutes)

Student teams will prepare to present their Ideate stage to their mentor and another group for feedback.


Design Team Member Names:

Ideate: Brainstorming the initial design

Draw 4 versions of what your design could look like based on the interviews. Label the parts of the design.

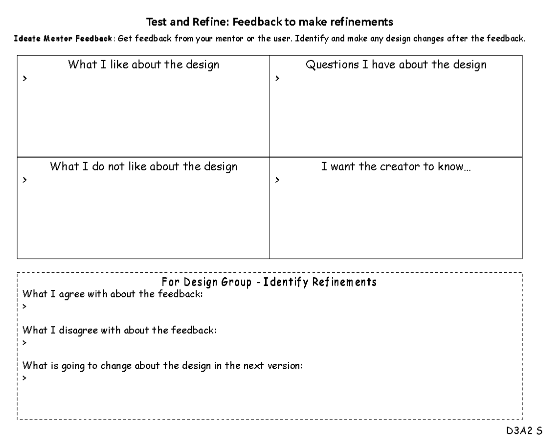
Draw what each of the actions (emotions, movements, etc.) will look like. Note in the drawing which inputs would start each of the actions.

Brainstorm a few names for your product:

10. **Ideate Feedback** (15 minutes)
    Students will approach their mentor for feedback on their design ideas.
    The mentor will use the four-square framework as a guide:
    a. What I like about the design?
    b. What I do not like about the design?
    c. Questions I have about the design.
    d. I want the creator to know.

After talking with the mentor, the students will brainstorm on how to refine their design.
** Note if a mentor is not available, the students will get feedback from the teachers or other adults supporting the camp.

11. Students will another design team to ask for feedback about their four design ideas. They will use the four-square framework as a guide:
    a. What we like about the design?
    b. What do we not like about the design?


Test and Refine: Feedback to make refinements

Ideate Mentor Feedback: Get feedback from your mentor or the user. Identify and make any design changes after the feedback.

What I like about the design	Questions I have about the design
>	>
What I do not like about the design	**I want the creator to know…**
>	>

For Design Group - Identify Refinements
What I agree with about the feedback:
>
What I disagree with about the feedback:
>
What is going to change about the design in the next version:
>

D3A2 S

c. Questions we have about the design. d. We want the creator to know.	**Test and Refine: Feedback to make refinements** **Ideate Peer Feedback**: Get feedback from another design group. Identify and make any design changes after the feedback. What I like about the design > Questions I have about the design > What I do not like about the design > I want the creator to know… > For Design Group - Identify Refinements What I agree with about the feedback: > What I disagree with about the feedback: > What is going to change about the design in the next version: > D3A2 S
12. **Build Prototype 1** (40 minutes) The students will continue to brainstorm following the feedback they received from their mentor and another design team during the previous day. The students will use the feedback received to start building the first prototype for their design. a. The students will start by making changes to the process map of the actions for their design. b. The students will identify the changes needed for their codes and sequence of codes. c. Once they have identified the changes to the codes and sequence, they will use MakeCode editor to program their Micro:bit.	Design Team Member Names: **Prototype: Build and test different designs** **PROTOTYPE 1** Instructions: 1. Grab a blank sheet of paper and label it: **PROTOTYPE 1** with your name at the top 2. Create 2 process maps for the (1) ACTIONS of the Micro:bit and the (2) CODE you are going to use to create the design. Label the process maps. 3. Begin building a prototype of your design based on the ideas you came up with in the ideate stage. 4. Add any additional resources you may have used to the DEFINE page of the design plan. 5. Answer the questions on the **REFINEMENT** page to track the changes in your design Example Paper Names Prototype 1 Actions Process Map Code Process Map D3A2 S
13. **Prototype 1 Feedback** (20 minutes) Students will approach their mentor and another design team for feedback on their first prototype. The mentor will use the four square framework as a guide: a. What I like about the design? b. What I do not like about the design? c. Questions I have about the design. d. I want the creator to know. ** Note if a mentor is not available, the students will get feedback from the teachers or other adults supporting the camp.	**Test and Refine: Feedback to make refinements** **Prototype 1 Mentor Feedback**: Get feedback from your mentor or the user. Identify and make any design changes after the feedback. What I like about the design > Questions I have about the design > What I do not like about the design > I want the creator to know… > For Design Group - Identify Refinements What I agree with about the feedback: > What I disagree with about the feedback: > What is going to change about the design in the next version: > D3A2 S

14. Students will approach another design team to ask for feedback about their first prototype using the four square framework as a guide:
    a. What we like about the design?
    b. What we do not like about the design?
    c. Questions we have about the design.
    d. We want the creator to know.

Test and Refine: Feedback to make refinements

Prototype 1 Peer Feedback: Get feedback from another design group. Identify and make any design changes after the feedback.

What I like about the design	Questions I have about the design
>	>
What I do not like about the design	**I want the creator to know…**
>	>

For Design Group - Identify Refinements

What I agree with about the feedback:
>

What I disagree with about the feedback:
>

What is going to change about the design in the next version:
>

D3A2 S

15. **Refine** (10 minutes)
    The student teams will continue to discuss the feedback from their mentor and another design team. They will brainstorm on how to incorporate the feedback and proceed to changing the necessary actions in their action process maps and then changing the codes.
    a. Then the students will approach their mentors again to show their actions process map and codes. The students will not incorporate the changes yet, but only show the changes to the actions process map. This is to prepare them for the following day's activities.

Refinement: Making and tracking changes to the design

Stage	What stayed the same from your previous design? How does it support the problem statement?	What changed from your previous design? How does it support the problem statement?
Ideate to Prototype 1		
Prototype 1 to Prototype 2		
Prototype 2 to Final Design		

D3A2 S

16. **Build prototype 2** (40 minutes)
    a. The students will continue to brainstorm following the feedback they received from their mentor, another design team and from the Gallery walk. The students use the feedback to start building the second prototype for their design.
    b. The students will start by making changes to the process map of the actions for their design.
    c. The students will identify the changes needed for their codes and sequence of codes.
    Once they have identified the changes to the codes and sequence, they will use MakeCode editor to make changes to their program.

Design Team Member Names:

Prototype: Build and test different designs
PROTOTYPE 2

Instructions:
1. Grab a blank sheet of paper and label it: **PROTOTYPE 2** with your name at the top
2. Create 2 process maps for the (1) ACTIONS of the Micro:bit and the (2) CODE you are going to use to create the design. Label the process maps.
3. Begin building a prototype of your design based on the ideas you came up with in the ideate stage.
4. Add any additional resources you may have used to the DEFINE page of the design plan.
5. Answer the questions on the **REFINEMENT** page to track the changes in your design

Example Paper

Names — Prototype 2

Actions Process Map

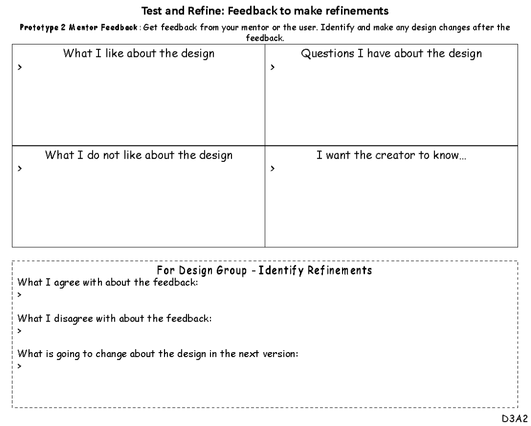
Code Process Map

D3A2 S

17. **Prototype 2 feedback** (15 minutes)
    Students will approach their mentor and a peer design team for feedback (30 minutes)
    The mentor will use the four square framework as a guide:
    d. What I like about the design?
    e. What I do not like about the design?
    f. Questions I have about the design.
    g. I want the creator to know.

** Note if a mentor is not available, the students will get feedback from the teachers or other adults supporting the camp.

Students will approach another design team to ask for feedback about their second prototype using the four square framework:
    h. What we like about the design?
    i. What we do not like about the design?
    j. Questions we have about the design.
We want the creator to know.

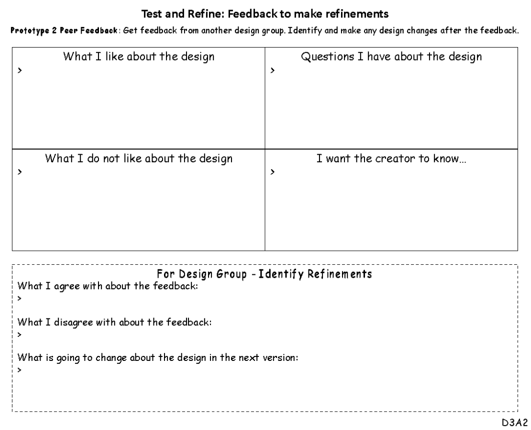

Test and Refine: Feedback to make refinements

Prototype 2 Mentor Feedback: Get feedback from your mentor or the user. Identify and make any design changes after the feedback.

What I like about the design	Questions I have about the design
>	>
What I do not like about the design	**I want the creator to know…**
>	>

For Design Group - Identify Refinements
What I agree with about the feedback:
>
What I disagree with about the feedback:
>
What is going to change about the design in the next version:
>

D3A2 S


Test and Refine: Feedback to make refinements

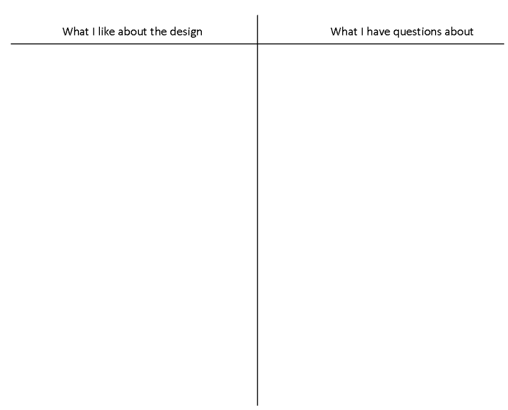
Prototype 2 Peer Feedback: Get feedback from another design group. Identify and make any design changes after the feedback.

What I like about the design	Questions I have about the design
>	>
What I do not like about the design	**I want the creator to know…**
>	>

For Design Group - Identify Refinements
What I agree with about the feedback:
>
What I disagree with about the feedback:
>
What is going to change about the design in the next version:
>

D3A2 S

18. **Progress check and gallery walk** (20 minutes) In pairs, members of each design teams will take turns in presenting their design and walking around to check the designs of other teams. Each pair must visit all the design groups and provide feedback.
    During the first 30 minutes, two members of each design team will stay to present, while the other two members will walk around to give feedback on the other designs. The pairs switch after 30 minutes. (60 minutes)
19. When students approach a Design team, the design team will talk the following:
    a. Who are the intended users.
    b. The purpose of the design.
    c. Features of your designs. (demonstrate what the prototype can do)
    d. The functions of your design.
    e. What are your next steps?


What I like about the design	What I have questions about

f. Students will leave sticky notes about the following: i. What I like about your design. ii. A question I have about the design ***Each design group has a T-chart with "what I like about the design" and "what I have questions about." Students will have sticky notes to give feedback on***	
Wrap Up and Sign Out Surveys	

Day 4: Design Challenge – Continued

Overview

Activity	Time, minutes
Welcome and Ice Breaker	20
Design Challenge (Final Design, Work on Pitch Presentation, Project Pitch and Final Feedback)	190
Extra Challenge (whatever time is left)	120
Clean Up/Wrap up	30
Total (including +45 min for lunch)	405
Extra Time	15

Materials Needed for Day 4:

Technology	Craft Supplies	Activity Supplies
● Computers ● Micro:bits - 1 per student ● Servo motors ● Dupont Wires (male –female, male – male) ● Alligator Clips ● AAA Batteries ● Micro:bit Stem Kits – sensors, LED lights, motors	● Paper ● Tape ● Markers ● Pencils ● Scissors ● Construction paper ● Paper towel/toilet paper rolls ● Cardboard/Cardstock ● Other craft materials for design projects	● Sticky Notes ● Sticky Easel Pad (optional) ● Rulers ● Student Rewards

Activity 1: Welcome and Icebreaker

Estimated Time: *Welcome – 10 minutes, Icebreaker – 10 minutes.* ***Total Time = 20 minutes***

Welcome

Icebreaker Activity: Students choose a superpower but cannot use words to say what it is

- Students are in random groups of 5
- Each one has to share their superpower

Activity goals:

- Students have to use system thinking to portray an idea without using words

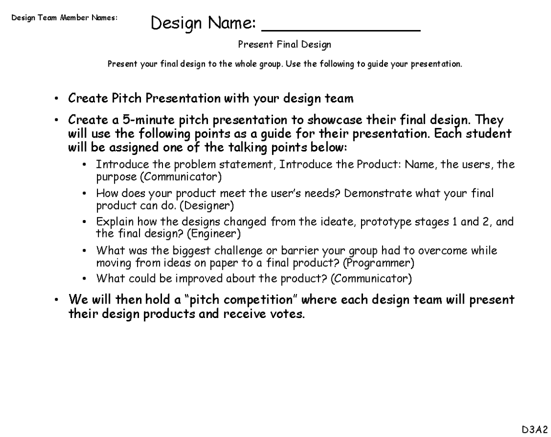
Activity 2: Design Challenge – Prototype 2, Feedback, Final Design

Estimated Time: *Final Design – 60 minutes, Work on Pitch Presentation – 60 minutes, Project Pitch and Final Feedback - 70 minutes, Extra Challenge – 90 minutes, Wrap-up/Clean Up – 30 minutes*. ***Total time = 340 minutes***

Activity Goals:

- Student will learn about and experience a design challenge and how the engineering design process is an essential component in completing design challenges.
- Students will go through the cycle of engineering design process with the guidance of a mentor.
- Student will learn public speaking skills.

Activity Procedure:

<table>
<tr>
<td>1. Final Design Preparation (60 minutes)
Design Teams will complete their final design. They will have to make sure that all the tasks in the Kanban boards are completed.</td>
<td></td>
</tr>
<tr>
<td>2. Create Pitch Presentation (60 minutes)
Students will create a 5-minute pitch presentation to showcase their final design. They will use the following points as a guide for their presentation. Each student will be assigned one of the talking points below:
a. Introduce the problem statement, Introduce the Product: Name, the users, the purpose (Communicator)
b. How does your product meet the user’s needs? Demonstrate what your final product can do. (Designer)
c. Explain how the designs changed from the ideate, prototype stages 1 and 2, and the final design? (Engineer)
d. What was the biggest challenge or barrier your group had to overcome while moving from ideas on paper to a final product? (Programmer)
e. What could be improved about the product? (Communicator)</td>
<td>
</td>
</tr>
<tr>
<td>3. Project Pitch and Feedback (70 Minutes) – 5 mins, 3 mins, 2 mins set up (10 mins per group X 6 groups)
Students will create a 5-minute pitch presentation to showcase their final design. They will use the following points as a guide for their</td>
<td></td>
</tr>
</table>

presentation. Each student will be assigned one of the talking points below:

a. Introduce the problem statement, Introduce the Product: Name, the users, the purpose (Communicator)
b. How does your product meet the user's needs? Demonstrate what your final product can do. (Designer)
c. Explain how the designs changed from the ideate, prototype stages 1 and 2, and the final design? (Engineer)
d. What was the biggest challenge or barrier your group had to overcome while moving from ideas on paper to a final product? (Programmer)
e. What could be improved about the product? (Communicator)

Lunch and Recess

Final Design Presentation Rubric Chart

Product Name:	
Category	**Rating 1-5 (1 lowest, 5 highest)**
The final product has a clear purpose.	
The final product meets the users needs.	
There was a clear progression in product design from each of the design stages. (Ideate, Prototype 1, Prototype 2, Final Design)	
The design team was able to identify and explain how they overcame a barrier in the design process.	
The design team had a firm understanding of where future improvements could be made.	
The pitch presentation was organized and easy to understand.	
Total Points	______/30 points

Activity 3: Extra Design Challenge

Estimated Time*: 0 - 120 minutes (Can be adjusted due to remaining time).*

Activity Goals:

- Students apply what they have learned this week to create a product of their own design using the microbit and extensions of choice.

Activity Procedure:

1. **Project Explanation and Creation** (100 minutes)
   a. Present Challenge to students. Students are to create their own product of choice using the microbit and extensions. Some possible choices are games and toys. Students may work independently or in groups of two.
      a. Encourage students to create a process map of their product first.
      b. Give students remaining time to create their product.
      c. Encourage students to give and receive feedback to peers during this time.

2. **Share Products** (20 minutes) a. Students will informally share their final products with the class. b. Students can also vote on the best product.	
3. **Wrap Up** 30 minutes a. Clean up all Materials b. Complete Final Surveys	
Wrap Up ?Clean Up 30 minutes	

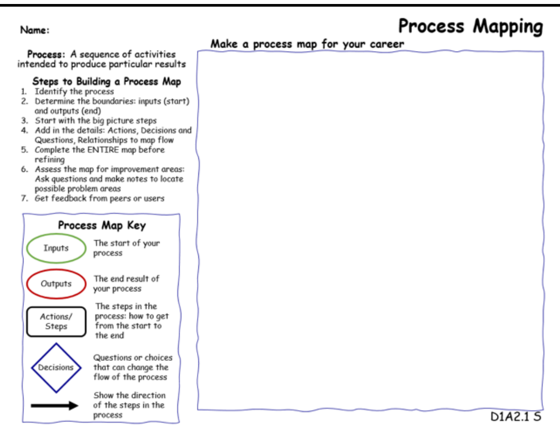
Extra Camp Activities

This section is composed of extra activities that you can do on certain activity days or on any day during the camp if there is extra time during the day.

Any Time

Activity: Career Talks

***Estimated Time**: 30 minutes*

1. Students will create a process map for their future including career goals, higher education goals, and the path they wish to take to reach their goals.	

After Programming Basics

The minimum requirement is to complete the programming basics to be able to complete the following activities. These activities can be added in extra time throughout the camp.

Activity: Hot Potato Game

***Estimated Time:** 60 minutes*

Activity Goals:

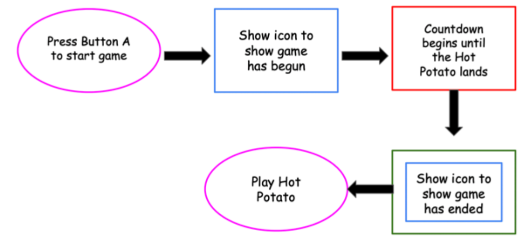
- Describe how the time taken grows with the size of input
- Explore the probability of finding a particular value in a random set.
- Identify how loops determine code

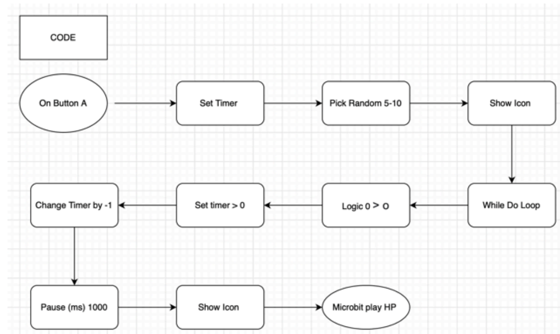
Activity Procedure:

- Have students play Hot Potato and familiarize themselves with the game
- Have students go through process mapping the Hot Potato Game
- Once students have created a map, allow them to code the game until they are satisfied and have hit all the requirements
- Have students play the game to recognize the different countdown possibilities

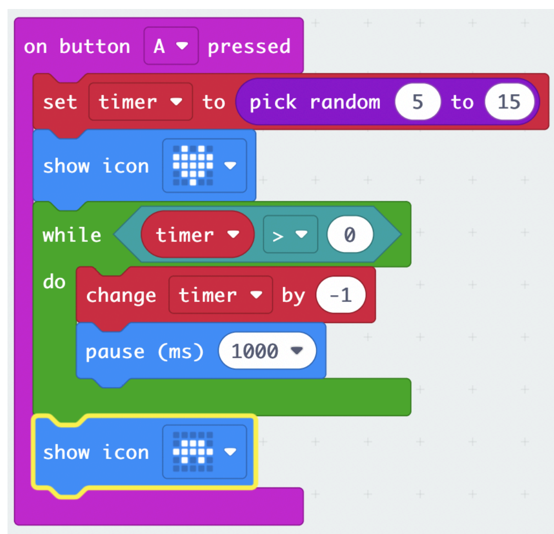
1. Have students play hot potato in groups of 4	
2. Introduce the activity: Your challenge is to design a Hot Potato Game using the Micro:bit. Provide the Program Requirements Below: a. Micro:bot needs a starting button to know when to start the game (the "potato becomes hot") b. Micro:bit needs to countdown to represent the number of seconds left before someone is caught holding the potato c. Micro:bit needs to display an icon to show the game has started and ended d. Micro:bit needs a loop and a timer variable **Facilitator Shares Slide and/or directs students to slide in online activity handout workbook**	Program Requirements 1. Microbit needs a starting button to start the game 2. Microbit needs a countdown to create the passing motion between students 3. Microbit needs to display an icon for the starting and ending of the game 4. Microbit needs a loop and a timer
2. Have the students identify the inputs and outputs for the game a. What will start the game? What is the input? b. What is the end result of the game or the output? 3. Have students think about the actions that have to happen to get from the start to the end result	Process Mapping 1. Review the process map for the **ACTIONS** needs for the Micro: bit to play Hot Potato ○ Recall the process map symbols ○ What is the input? What is the output? ○ Which actions do you think need to happen to get the end result? 2. Create a process map for the **CODE** you will need to make the actions happen • What kind of code blocks will you need? • What kind of input do you want to use to start the game • What is the sequence the code needs to run in? • How could you use **Logic** to make different actions happen based on different inputs? Example Process Map

<table>
<tr>
<td>
a. Which actions needs to happen for the game to run?
b. Actions Process Map is developed

Example Process Map Shown
</td>
<td>
Actions Processing Map


</td>
</tr>
<tr>
<td>
4. Guide students to take the Actions Process map they built and identify the code blocks they believe they will need. Students will create a CODE process map to match the actions they listed.
a. What kinds of code blocks will you need?
b. What kind of input do you want use to start the game?
c. What is the sequence the code needs to run in?
</td>
<td>
Example Code Process Map


</td>
</tr>
<tr>
<td>
5. Students move to MakeCode editor to build out their code the process mapped
a. Use the MakeCode editor to program the Micro:bit to play rock, paper, scissors
b. Test your code periodically using the simulator or the Mirco:bit to identify any bugs or test your sequence.

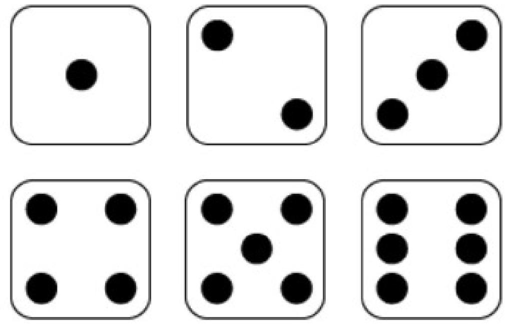
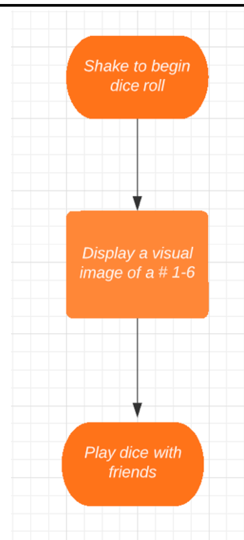
Example of Code Shown
</td>
<td>
Continuous Code


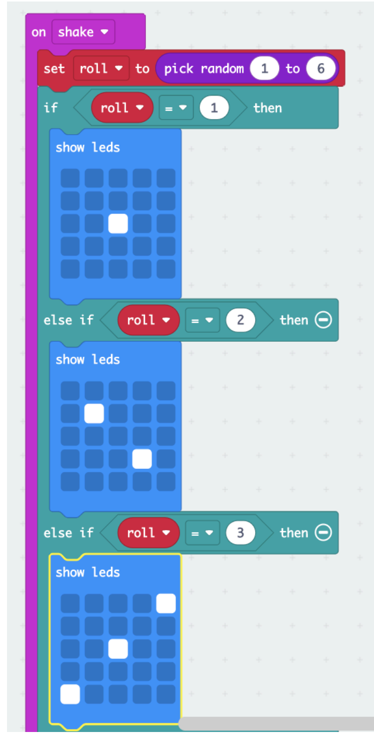
</td>
</tr>
<tr>
<td>
6. Check each group's programming when they are finished to make sure it follows their process maps and results in a complete game of Hot Potato
a. Students then load the program onto their Micro:bits
</td>
<td></td>
</tr>
</table>

7. Each students goes 2 rounds of Hot Potato against their Micro:bit	

Activity: Dice Roll

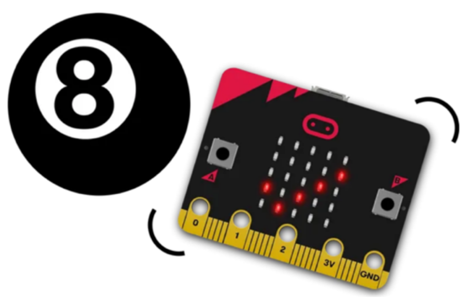
***Estimated Time**: 20 minutes*

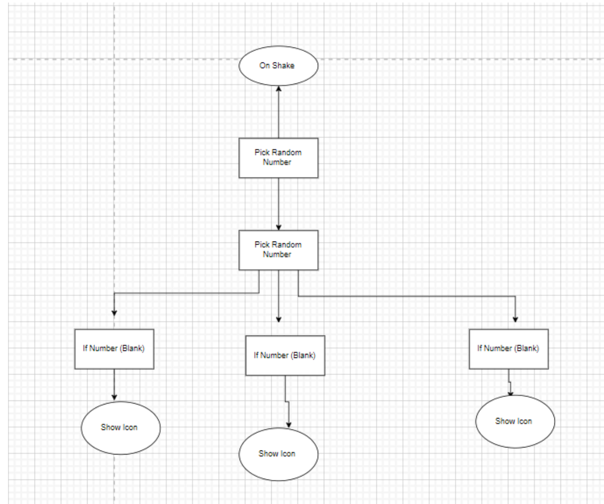
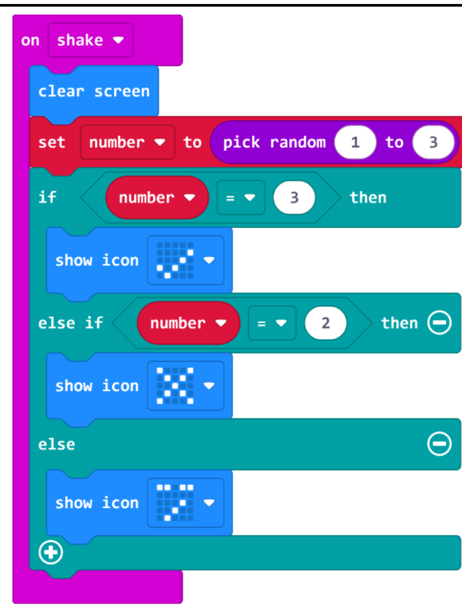
1. Students will code their Micro:bit to mimic that of a standard die with 6 numbers a. Students will use code to pick a random number and then assign led's to show the number	
2. Students will follow the process map given to code the Micro:bit. b. Students will utilize the actions process map to create a code process map	

3. When coded correctly, the students will shake the Micro:bit and and LEDs will appear that look like a face of a die c. Students will utilize the “on shake” function to initiate the start of the roll. d. Students will utilize variable tab, math tab, and logic tab in their code. e. Have students add sounds for each number or create new symbols to represent numbers 4. Have students play against each other and create games. Example games below. a. Highest roll wins b. First to get equal numbers c. Try to see how long it takes for all six numbers to appear on die (Micro:bit)	

Activity: Magic 8 Ball

***Estimated Time**: 20 minutes*

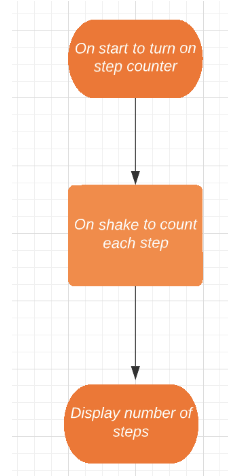
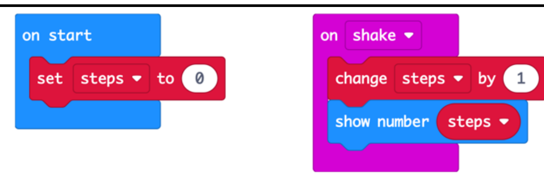
1. Students will code their Micro:bit to mimic that of a magic 8 ball	

2. Students will follow the process map given to code the Micro:bit. f. Students will utilize the actions process map to create a code process map	
3. Then coded correctly, every time the Micro:bit is shaken, the students will see an icon/image pop up on the screen g. Students will utilize the "on shake " function to initiate the the game h. Students will utilize a "pick random" to create game i. Students will use the logic blocks to create different scenarios j. Students will then display the variable 5. Utilize the battery pack to have students walk around and test out their Micro:bits	

Activity: Step Counter

***Estimated Time**: 20 minutes*

1. Students will code their Micro:bit to mimic that of a step counter k. Students will use the Micro:bit accelerometer to count how many times the Micro:bit has been shaken	

<table>
<tr>
<td>2. Students will follow the process map given to code the Micro:bit.
l. Students will utilize the actions process map to create a code process map</td>
<td>
</td>
</tr>
<tr>
<td>3. When coded correctly, every time the Micro:bit accelerometer input senses a shake, the program increases the variable by 1, and shows the new number on the LED display output.
m. Students will utilize the “on start” function to initiate the variable to 0
n. Students will utilize “on shake” function to count each step
o. Students will use the “change variable” function to increase the variable by 1 each time
p. Students will then display the variable
4. Utilize the battery pack to have students walk around and test out their Micro:bits
a. See if students can fashion the Micro:bit to their shoe detect the steps without having to manually shake it</td>
<td>

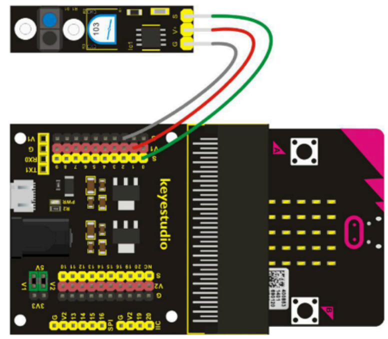
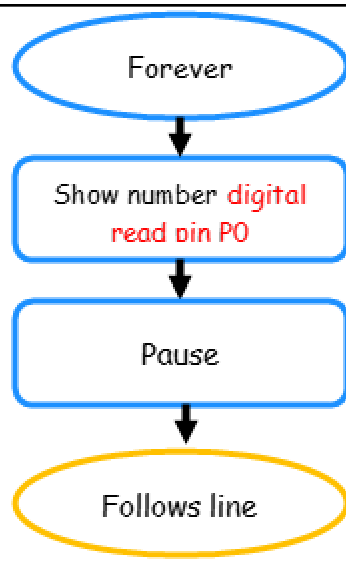
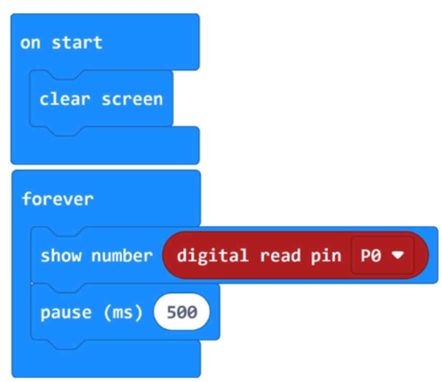
</td>
</tr>
</table>

After Micro:bit Pet Extensions

The completion of the Micro:bit pet extensions is required to complete this activity. Students need to have experience wiring external sensors to the Mibro:bit.

Activity: Follow Black Line

Estimated Time*: 20 minutes*

1. Students will wire the magnetic sensor to their Micro:bit. a. Wiring example shown. b. The sensor will identify if it there is a black line under it. This is done by detecting if there are infrared rays are emitted at a high or low level.	
2. Students will follow process map given to code the Micro:bit.	
3. When coded correctly, the sensor when detecting a black line will read the number 1, if reading any number besides 1, there is no line detected. a. This is due to when a line is detected, the infrared rays are not emitted or have a low intensity. The sensor's signal will output a high level showing the number 1. 4. Have students test the code using different thicknesses of black lines.	

After Technical Design Challenge

This activity is a great way to get students into programming python. Python is a language that could be considered the next step after block programming.

Activity: Turtle

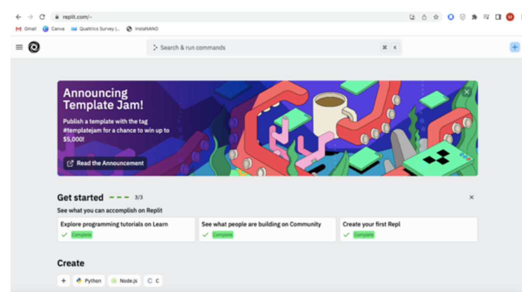
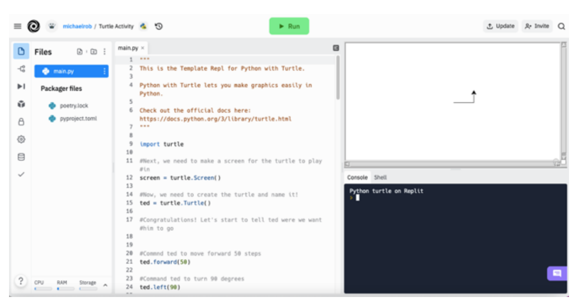
Estimated Time: 60 minutes

Activity Goals:

- Develop a basic understanding of how to use the Turtle package in Python
- Learn how to customize and create a simple software application like a software engineer

Activity Procedure:

- Have students work through the included activity posted on Repl.it
- Have students get creative with the code they learn in the activity
- Have students identify what the different functions presented in the activity do

1. Have students go to the the Repl.it site and find the activity named "Turtle Activity"	replit.com Show the first 42 seconds of this video if able https://www.youtube.com/watch?v=pGpDFmVa4Uo
2. Introduce the activity: You are to follow the activity and eventually create a drawing using the code learned Requirements Below: a. The drawing needs to have color b. The drawing should have at least two shapes included	
3. After letting the students work through the activity, have the students identify the functions in the application a. Describe different functions that make the turtle move? b. What is the end result of a certain command or the output? 4. Have students think about the actions that have to happen to get from the start to the end result	

The activity is entirely done in a script of code they can run on Repl.it, all the instructions are in there. Basically, I have created a template in which the students will copy and paste lines of code in the appropriate order. A big thing the students need to know how to do is to uncomment lines of code.	Students can either copy and paste whole commented code lines with the "#" and then delete the "#" when they want to use the command Students can also only copy lines of code without including the "#"