The Booklet

Projects with Micro: bit

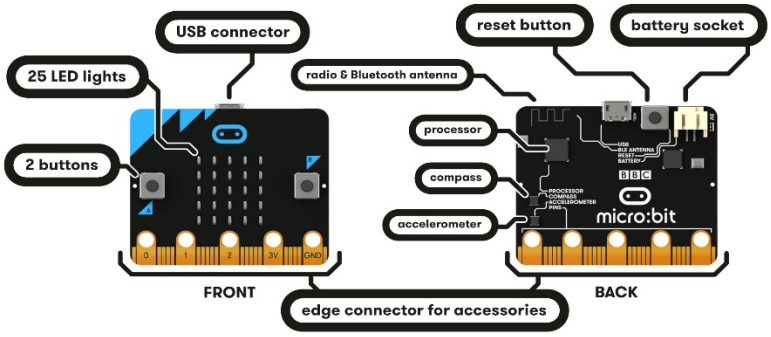
- **Micro:bit 2**
- **Hot Potato Game.....4**
- **Environment Exploratio System.....12**
- **Turtle-Friendly LightingSystem.....18**
- **Rock-Paper-Scissors Game.....23**
- **Digital Compass with Micro:bit.....28**
- **Steps Tracker.....37**

Micro: bit

1 - What is Micro: bit? :

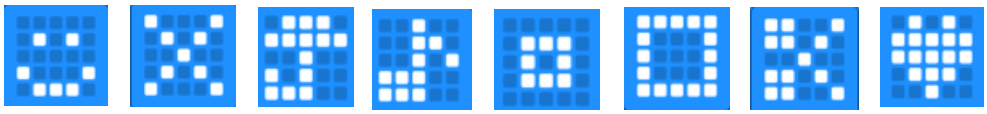
is a small, programmable computer designed to help students learn about technology, coding, and electronics.

2 - Micro: Bit Components:

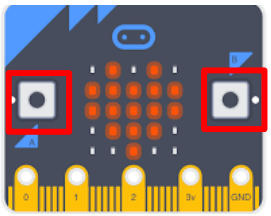


Front Side:

- LED Display (5x5 Grid):
  A grid of 25 small LED lights that can show numbers, letters, shapes, and even simple animations. You can program it to display images, text as shown below:



- Buttons (A and B):
  Two programmable buttons (A and B) that can be used for interacting with the micro: bit, such as controlling games or triggering actions in your program.



- USB Connector:
  A micro-USB port that is used to power the micro: bit and to transfer programs (code) from your computer to the micro: bit.

- Edge Connector for Accessories:
  The golden strip at the bottom has 25 pins. These can be connected to external components like sensors, motors, or LEDs, making the micro: bit highly customizable for projects.

Back Side:

- Radio & Bluetooth Antenna:
  Allows the micro: bit to wirelessly communicate with other devices, including other micro: bits or smartphones, using radio signals or Bluetooth.

- Reset Button:
  A small button to restart the micro: bit. This is useful if you want to run your program from the beginning or troubleshoot.

- Battery Socket:
  A slot for attaching an external battery pack to power the micro: bit when not connected to a computer.

- Processor:
  The (brain) of the micro: bit that runs the code you upload. It processes all inputs and controls the outputs.

- Compass:
  A sensor that detects the Earth's magnetic field, allowing the micro: bit to function as a digital compass or detect nearby magnets.

- Accelerometer:
  A sensor that detects motion and orientation. It can tell if the micro: bit is being shaken, tilted, or moved, which is great for motion-sensitive projects.

Hot Potato Game

Overview

The **Hot Potato Game** transforms the Micro:bit into a fun and interactive tool that combines logic, randomness, and real-time reactions. This project supports Logical Systems and introduces learners to the basics of Machine Learning and Decision-Making Algorithms. It's a playful way to develop critical thinking, creativity, and collaborative skills.

Goals Addressed

- **Goal 1. Logical Systems**
  - Use variables, loops, and decision-making algorithms to create the game logic.
- **Goal 14. Sensor Integration**
  - Incorporate button input as the trigger for gameplay.
- **Goal 15. Product Prototyping**
  - Build and test a fully functional game prototype.
- **Goal 16. Pitching Skills**
  - Present and explain the game mechanics to others.

Why It's Fun

- **Interactive Gameplay**: Engage players in a fast-paced, thrilling group activity.
- **Randomized Outcomes**: The timer adds suspense, as players never know when it will stop.
- **Creative Coding**: Customize the game with different sounds, images, or timer ranges for added excitement

Hot Potato Game

This Guide on How to Create the Hot Potato Game with Micro:bit

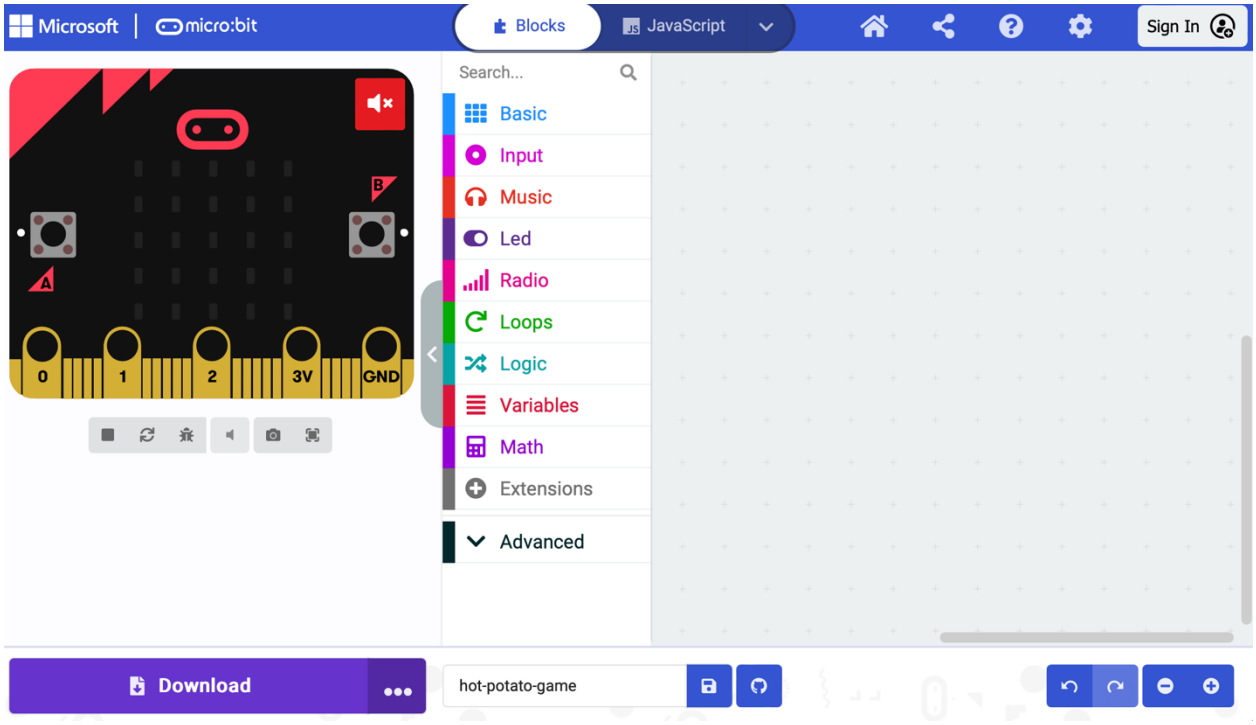
The Project Idea

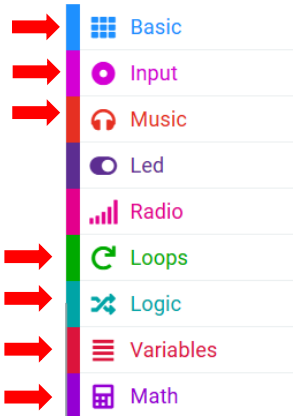
This project explores Logical Systems by turning the Micro:bit into a thrilling group game where players test their reaction and timing skills. The game adds suspense and excitement by using a random timer, making each round unpredictable and fun.

• Button A starts the game and sets the timer to a random value.
• The timer counts down, displaying a dynamic LED pattern.
• When the timer hits zero, the Micro:bit plays a sad sound and displays a cross to signal that the round is over.

To implement this project, we will use the **MakeCode Editor by Microsoft**.

- **Step 1:**
  Search for MakeCode Editor in google:
  https://makecode.microbit.org/#editor
- **Step 2:**
  Click on new project and name it: Hot Potato Game
  Then this screen will appear





- **Step 3:**

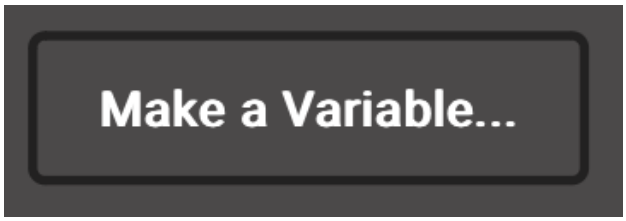
We just will use this command Groups:

-Start the Game with Button A

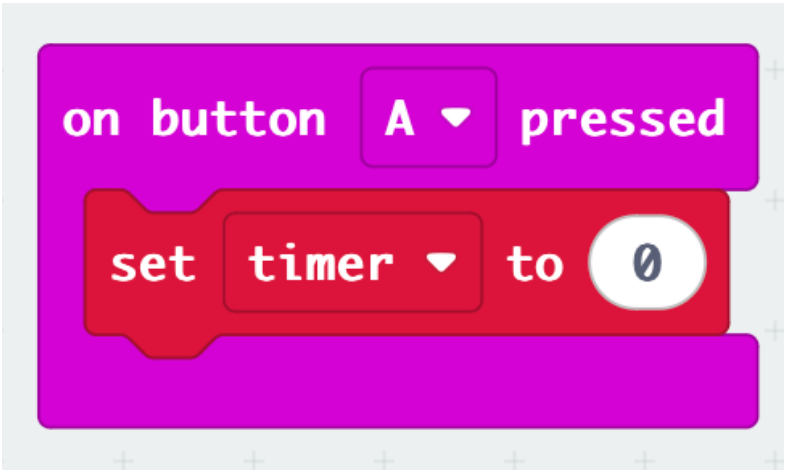
From the **Input** category, drag the **"on button A pressed"** block

-Create a Timer Variable

Go to the **Variables** category and create a new variable named **"timer"**.

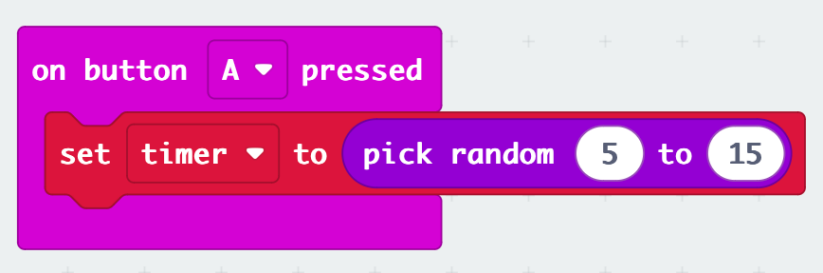


Note:
It's always a good practice to give your variables meaningful names to make your code easy to understand.



-Set a Random Timer

Since we want the timer to start with a random value, go to the **Math** category and select the **"pick random"** block.

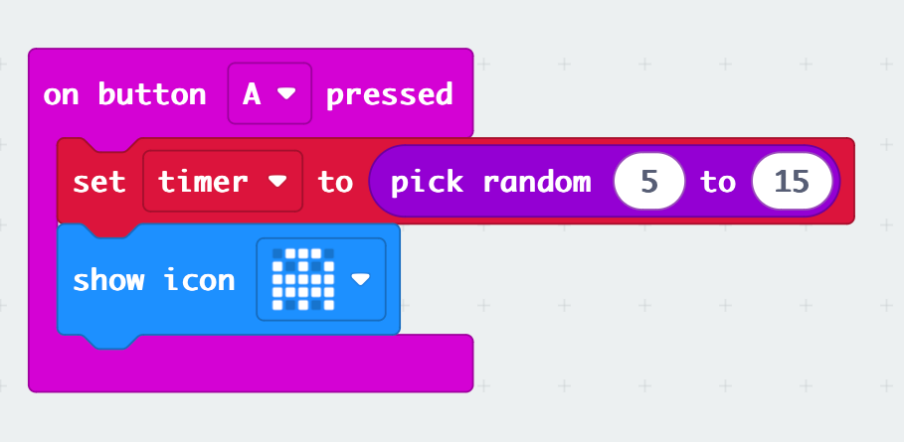


Set the range to any values you like (e.g., **5 to 15**).

Assign this random value to the **"timer"** variable by dragging the **"set timer to"** block from the **Variables** category.

-Show a Start Icon

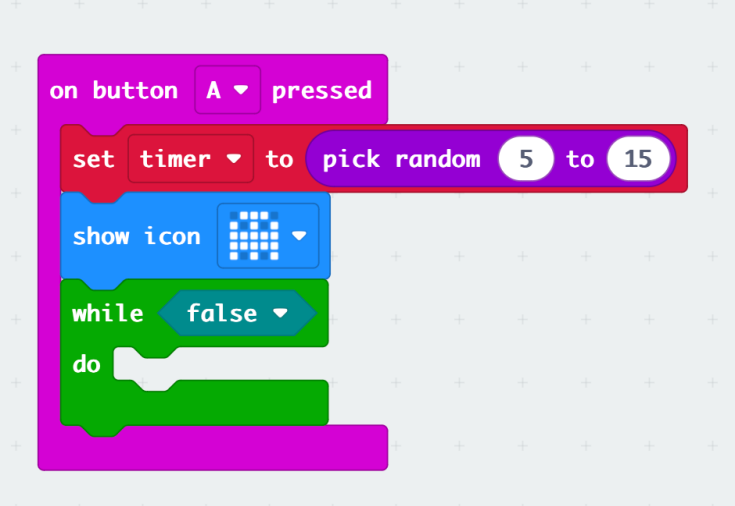
To indicate that the game has begun, go to the **Basic** category and drag the **"show icon"** block.



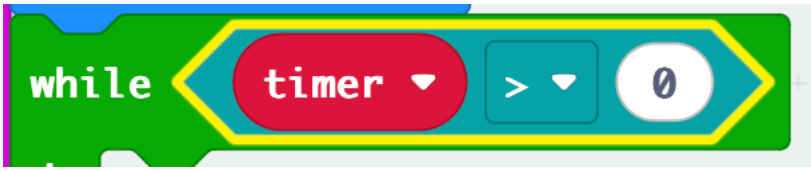
Choose any icon you like—be creative!

-Add a While Loop for Countdown

Use the **"while do"** loop from the **Logic** category to control when the countdown starts and stops

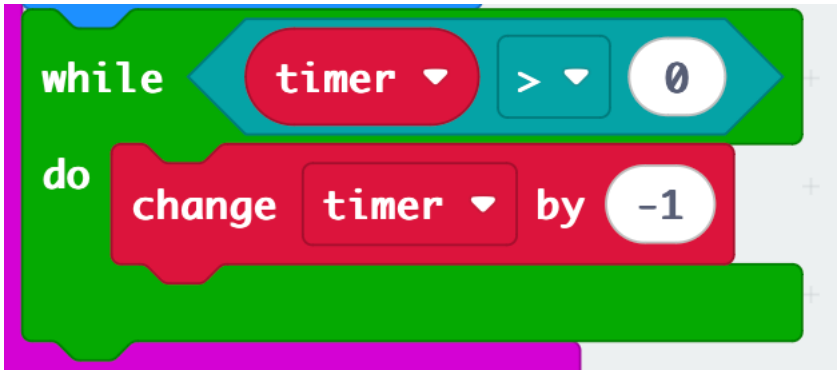


For the condition, use a **Logic Comparison** block to check if the **"timer"** variable is greater than zero (**timer > 0**).



-Countdown Logic

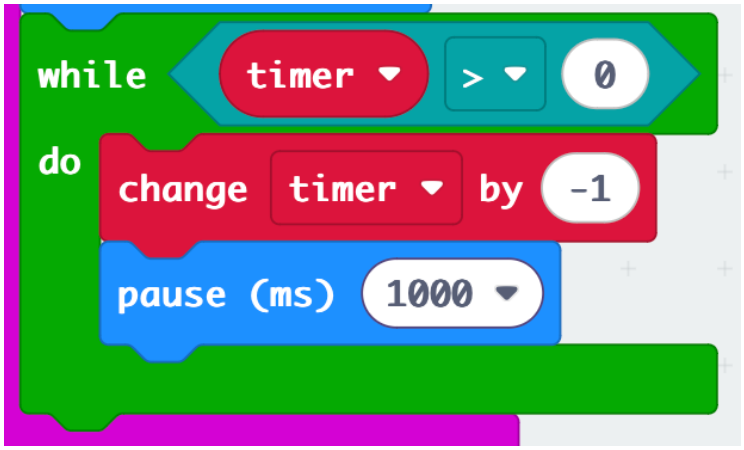
Inside the loop, drag the **"change timer by"** block from the **Variables** category.



Set it to **-1** to count down by one each time the loop repeats.

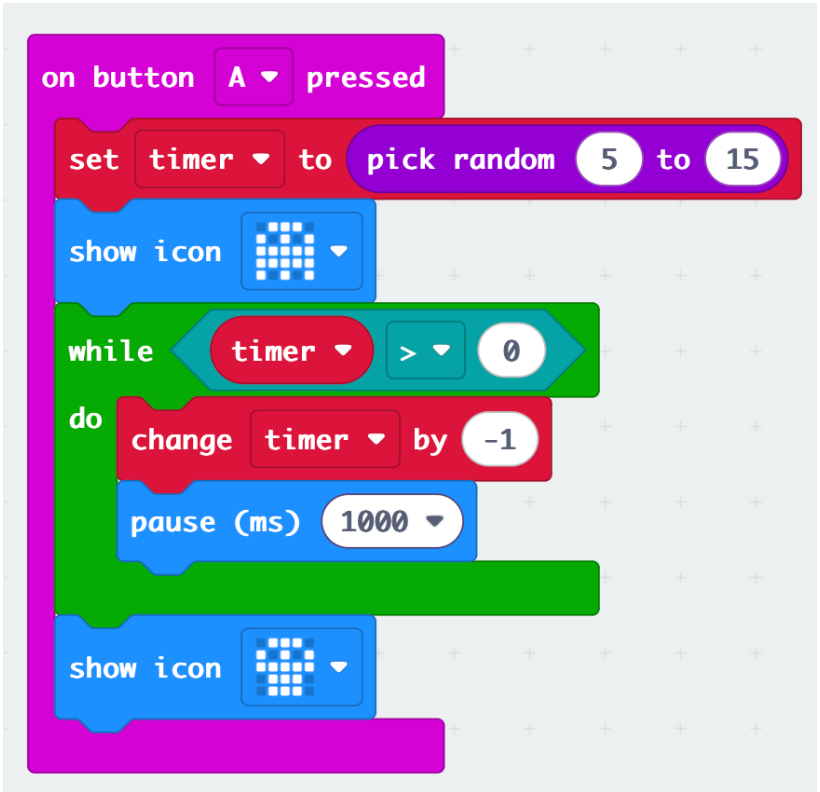
-Add a Pause

To create a delay between each countdown step, go to the **Basic** category and drag the **"pause"** block inside the loop. Set the pause to **1000ms** (1 second).



-End the Game

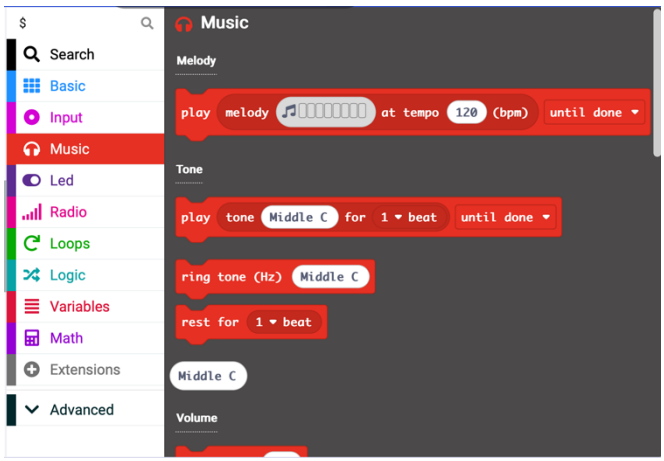
Once the timer reaches zero, the loop will stop. To indicate that the game is over, drag another **"show icon** "block from the **Basic** category and place it below the loop.



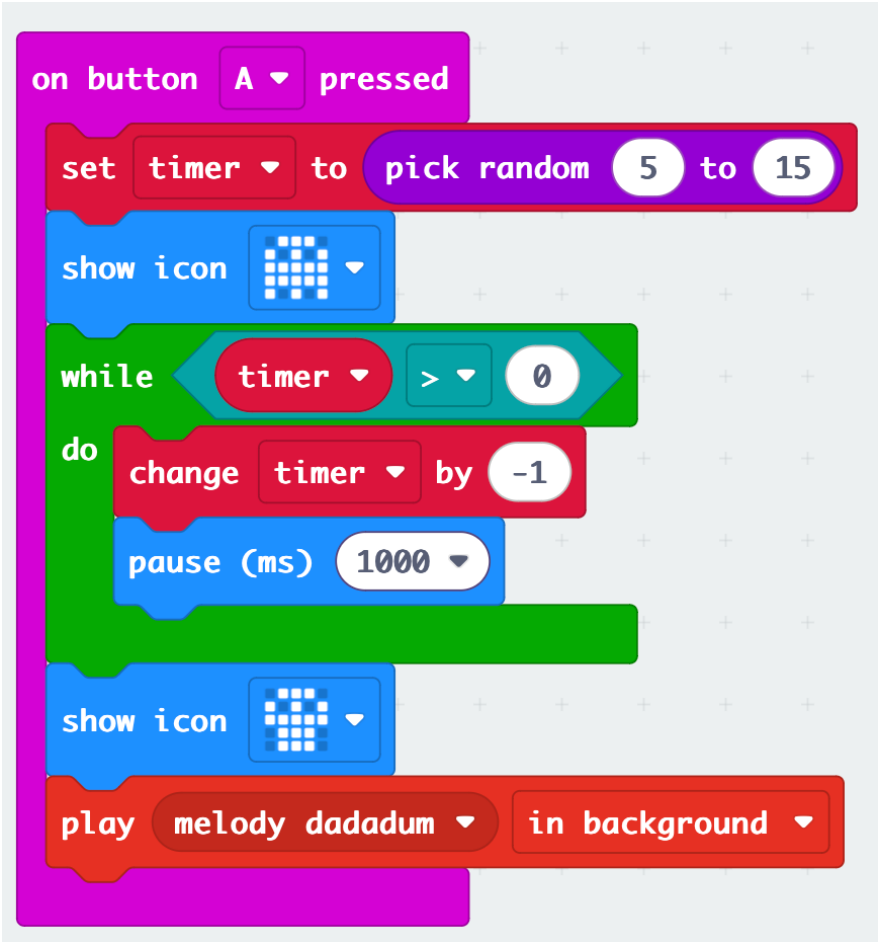
Choose an icon to signal the end of the game (e.g., a sad face or an "X")

-Add Background Music

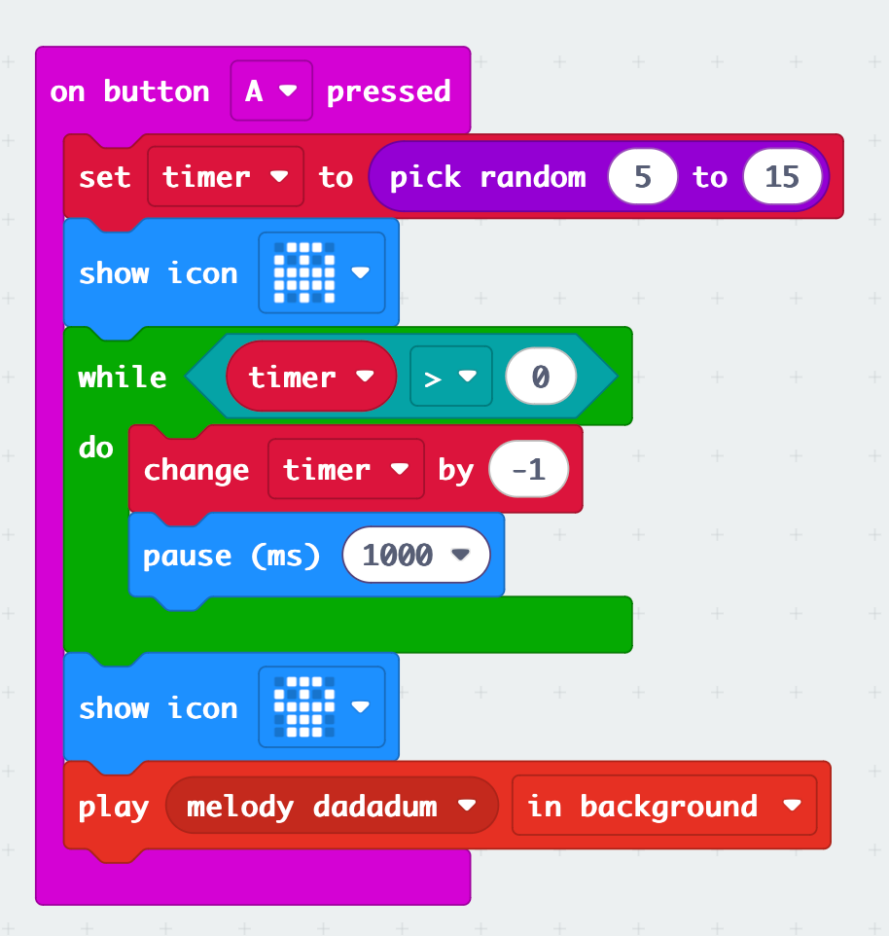
To make the game more exciting, go to the **Music** category and choose **"play melody"** or **"play sound"**.



Place it after the **show icon** block and select a fun melody or sound effect to play when the game ends.



This is how it should look when completed:

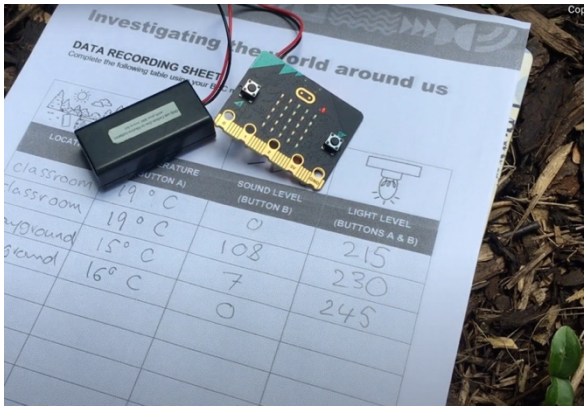


By diving into the Hot Potato Game project, you're turning simple coding into an exciting, interactive experience that brings **people together**. This activity combines creativity, logic, and teamwork, giving you the tools to create something fun and meaningful. Be proud—you're transforming programming into play and innovation into entertainment! 🎉

Environment Exploration System

Overview

The **Environment Exploration** activity transforms the Micro:bit into a versatile tool for measuring **temperature**, **sound levels**, and **light intensity**. This project supports **Sensor Integration** and **Logical Systems**, allowing learners to collect and analyze environmental data while fostering critical thinking and innovation.



Goals Addressed

- **Goal 7. Environmental Challenges**
  - Explore ways to monitor and understand environmental factors like temperature and noise levels.
- **Goal 14. Sensor Integration**
  - Combine multiple sensors to gather data about the surroundings.
- **Goal 1. Logical Systems**
  - Use decision-making algorithms to interpret sensor data and provide meaningful outputs.
- **Goal 15. Product Prototyping**
  - Build and test a prototype for environmental monitoring systems.

Why It's Fun

- **Hands-On Discovery:** Turn science investigations into a treasure hunt for the quietest, warmest, or brightest spot.
- **Instant Feedback:** See real-time results on the Micro:bit display, making data collection engaging and fun.
- **Personalization:** Add emojis or creative messages to represent your data visually (e.g., 🌡️ for heat, 🔊 for sound)

Environment Exploration System

This Guide on How to Implement the Environment Monitoring Project with Micro: bit

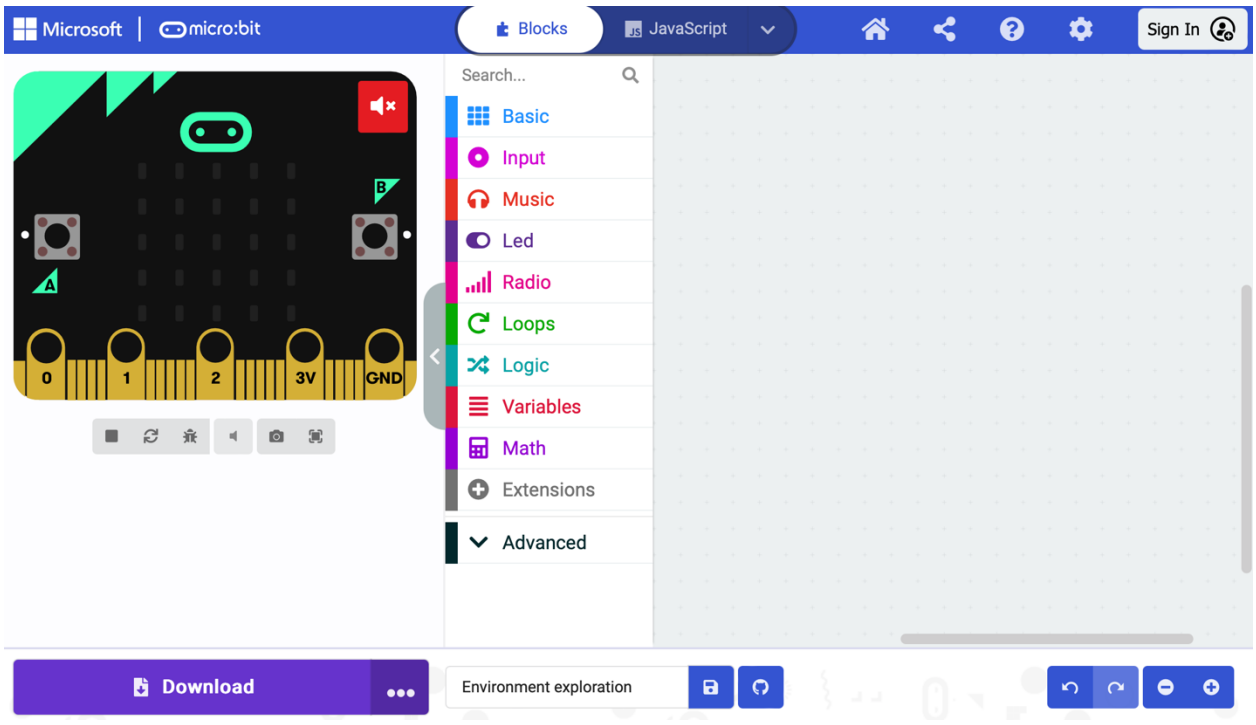
The Project Idea

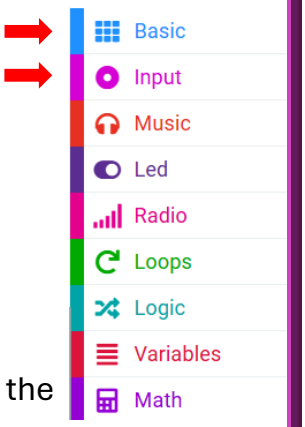
This project addresses **Environmental Challenges** by transforming the Micro: bit into a powerful tool for measuring temperature, sound levels, and light intensity. It empowers learners to collect and analyze environmental data while encouraging critical thinking and innovation.

- A temperature sensor measures environmental heat and displays it on the Micro: bit.
- A sound sensor detects noise levels and provides real-time feedback.
- A light sensor measures brightness, helping monitor light pollution or dimness.

To implement this project, we will use the **MakeCode Editor by Microsoft**.

- **Step 1:**
  Search for MakeCode Editor in google:
  https://makecode.microbit.org/#editor
- **Step 2:**
  Click on new project and name it: Environment Exploration
  Then this screen will appear



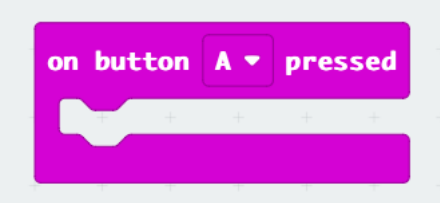


- **Step 3:**

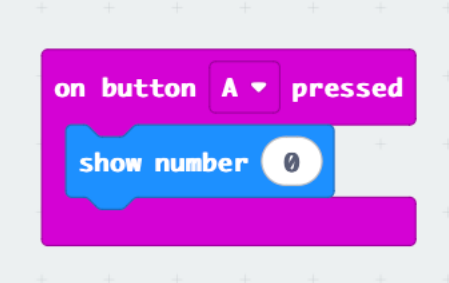
We just will use this command Groups:

-Temperature Sensor: Start with the temperature sensor setup

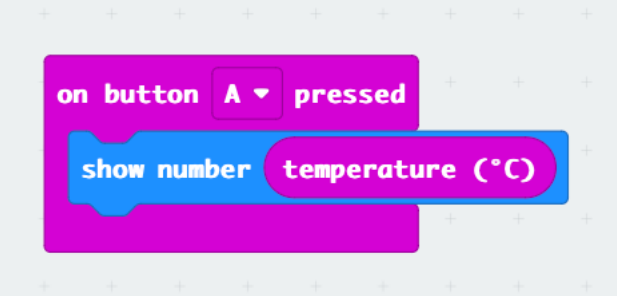
- From the **Input** category, drag the **"on button pressed"** block into the workspace and set it to **button A**.



- Next, go to **Basic**, find the **"show number"** block, and drag it into the workspace.



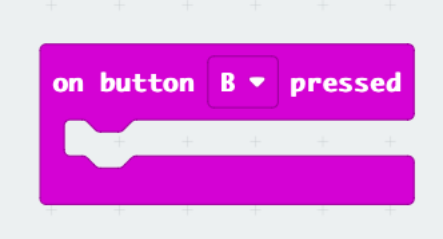
- Then, go back to **Input**, scroll down, and select the **"temperature"** block place it inside the **"show number"** block.



Result: When **button A** is pressed, the Micro: bit measures and displays the temperature in degrees Celsius.

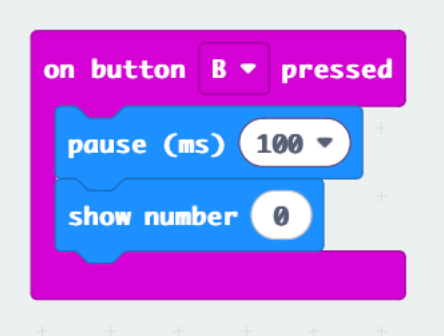
-Sound Meter:

- From the **Input** category, drag the **"on button pressed"** block and set it to **button B**.

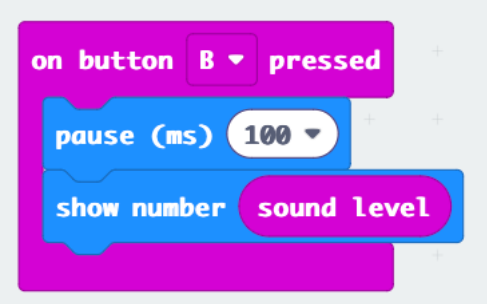


- Next, go to **Basic**, find the **"pause"** block, and set it to any number (e.g., 100).

- Then, go back to **Basic** and drag another **"show number"** block into the workspace.



- Finally, return to **Input** and select the **"sound level"** block, placing it inside the **"show number"** block.

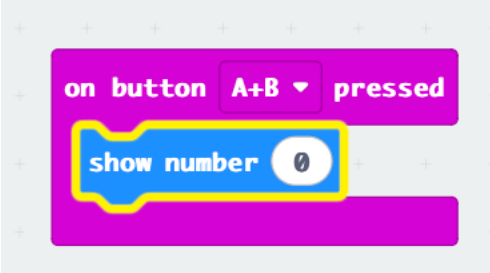


Result: When **button B** is pressed, it displays the sound level on a scale of **0** (quietest) to **255** (loudest).

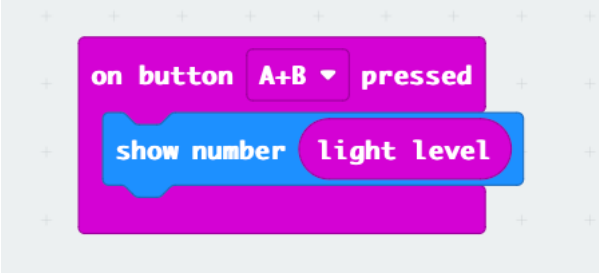
-**Light Intensity Meter:**

- From the **Input** category, drag the **"on button pressed"** block and set it to **button A + B**.

- Go to **Basic** and drag the **"show number"** block into the workspace.

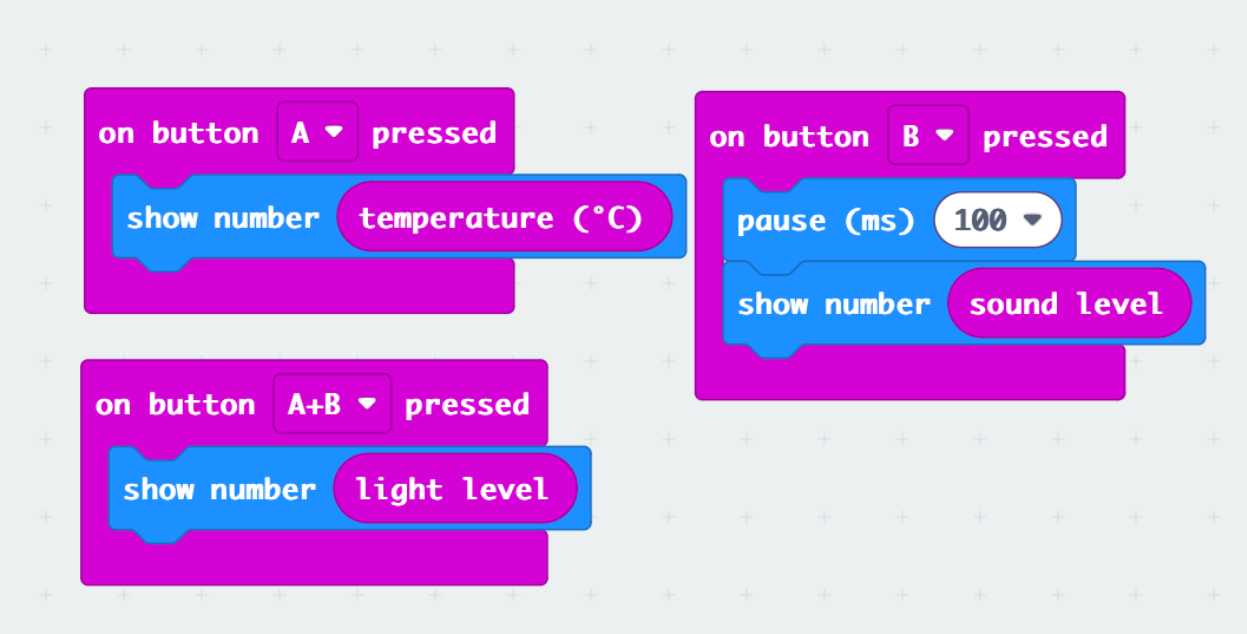


- Then, go back to **Input** and select the **"light level"** block. Place it inside the **"show number"** block.



Result: When **buttons A + B** are pressed together, the Micro: bit displays the light intensity level on a scale of **0** (darkest) to **255**(brightest).

This is how it should look when completed:



By engaging in the Environment Exploration project, you're taking science out of textbooks and into the real world. This activity turns everyday surroundings into a **living laboratory,** where curiosity meets technology. You'll uncover insights about temperature, light, and sound while developing the skills to think critically and solve problems creatively. Be proud—you're turning data into discovery and exploration into empowerment! 🌱🔍

Turtle-Friendly Lighting System

Overview

The Saving Sea Turtles activity uses the Micro:bit to tackle an **Environmental Challenge** by creating a turtle-friendly lighting system. Newly-hatched sea turtles rely on moonlight to navigate toward the ocean, but artificial lights often disrupt their path. This project leverages **Logical Systems**, **Sensor Integration**, and **Product Prototyping** to guide humans along beach paths at night while ensuring sea turtles are not confused by artificial lights.

Goals Addressed

- **Goal 7. Environmental Challenges**
  - Develop a solution that reduces the impact of artificial light pollution on marine life.
- **Goal 1. Logical Systems**
  - Program the Micro:bit to detect light levels and adjust the LED display accordingly.
- **Goal 14. Sensor Integration**
  - Use the Micro:bit's built-in light sensor to create a turtle-friendly system.
- **Goal 15. Product Prototyping**
  - Build a functional prototype for sustainable beach lighting.

Why It's Fun

- **Real-World Impact:** Build a solution that directly supports wildlife conservation.
- **Interactive Coding:** See the Micro:bit react to changes in light levels in real-time.
- **Creative Freedom:** Customize the LED display with unique patterns or animations to enhance user experience

Turtle-Friendly Lighting System

This Guide on How to Implement the Saving Sea Turtles Project with Micro: bit:

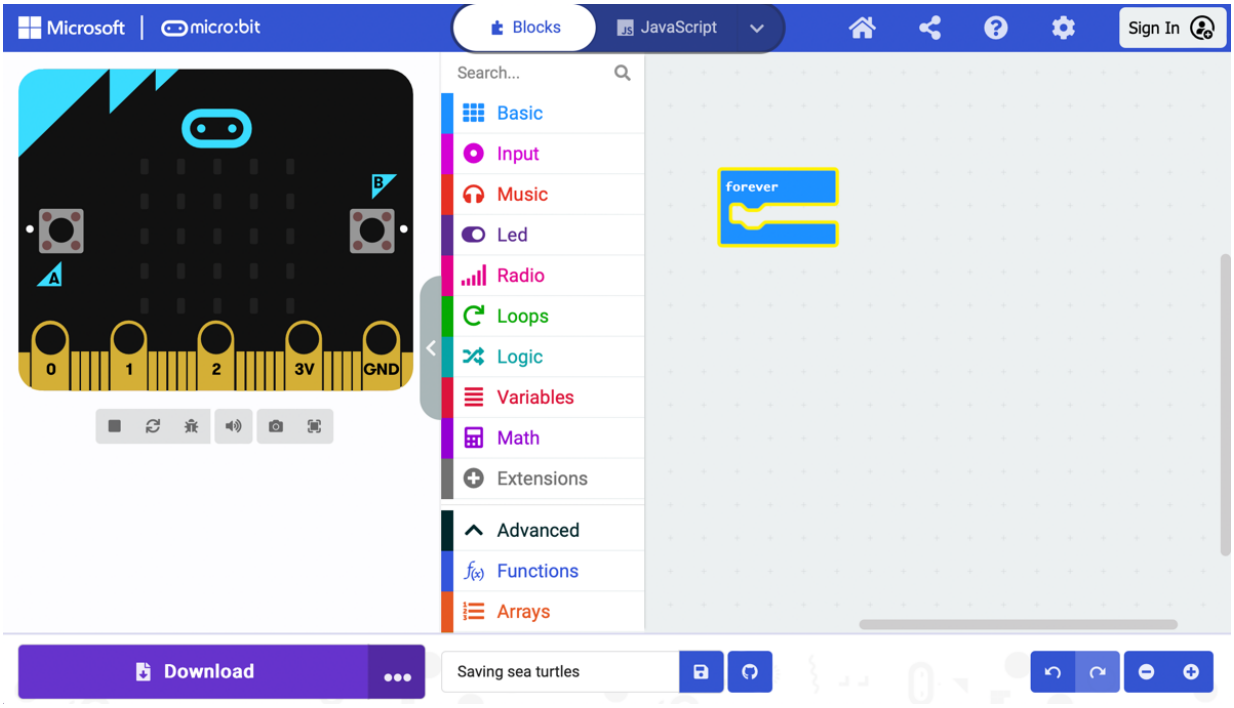
The Project Idea

This project tackles an **Environmental Challenge** by creating a turtle-friendly lighting system using the Micro:bit. Newly-hatched sea turtles rely on moonlight to navigate toward the ocean, but artificial lights often disrupt their path. This project ensures human safety on beach paths at night while preventing artificial lights from confusing sea turtles.

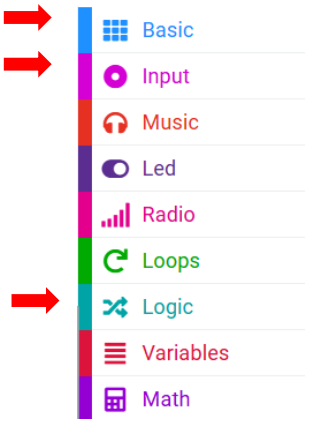
- A light sensor detects the brightness of the environment.
- If it's dark, the Micro:bit displays a turtle-friendly LED design.
- If it's bright, the LEDs turn off to minimize distraction.

To implement this project, we will use the **MakeCode Editor by Microsoft**.

- **Step 1:**
  Search for MakerCode Editor in google:
  https://makecode.microbit.org/#editor
- **Step 2:**
  Click on new project and name it: Saving sea turtles
  Then this screen will appear

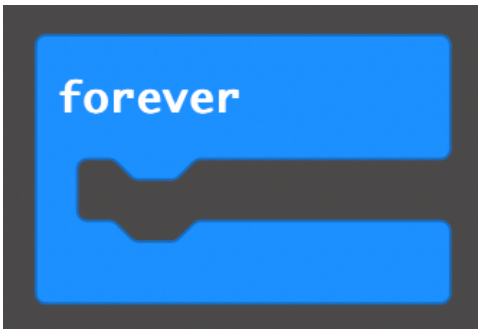


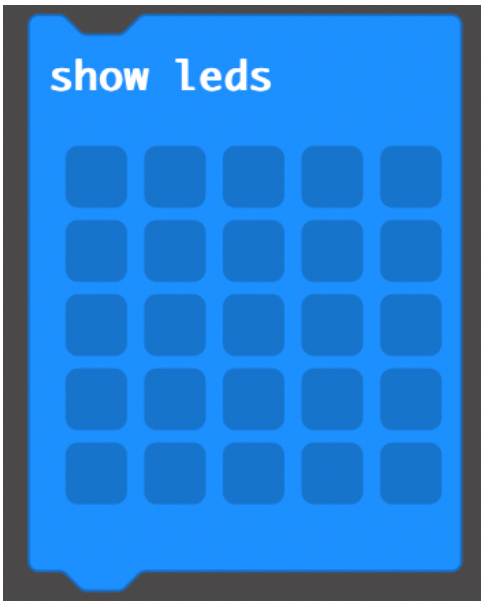
- **Step 3:**
  We just will use this command Groups:



  From Basic :

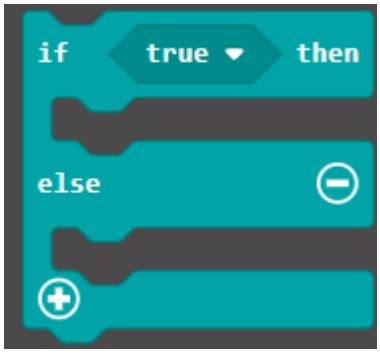
  We need





From Logic :

We need :



AND

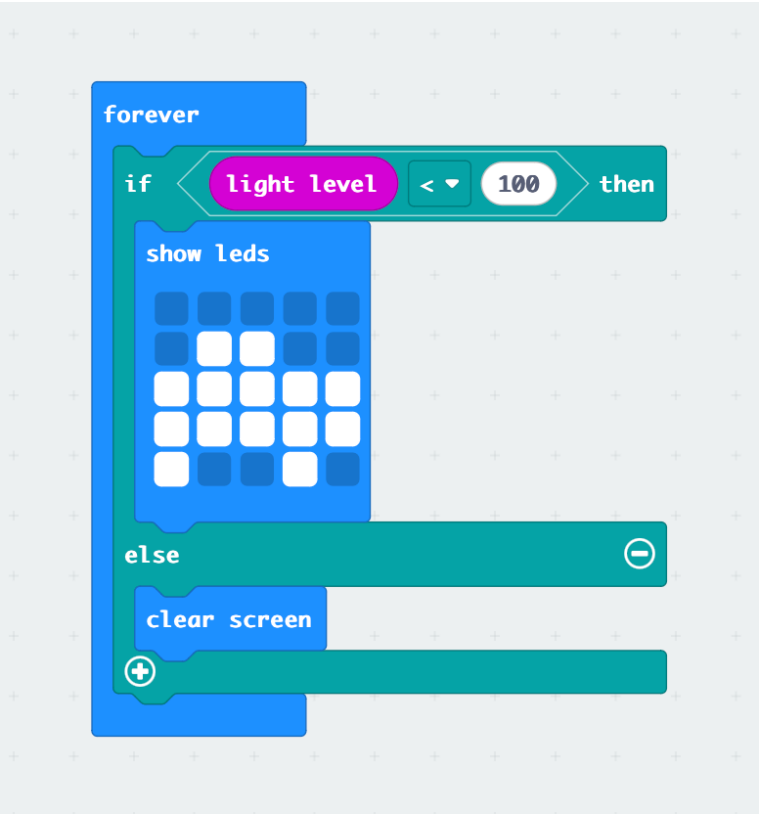
From Input :

Scroll down until you find (**light level**) Block

Next, drag it into the comparison logic block so that it looks like this

Feel free to experiment with the numbers and try different options; I've set it to 100 here as a starting point

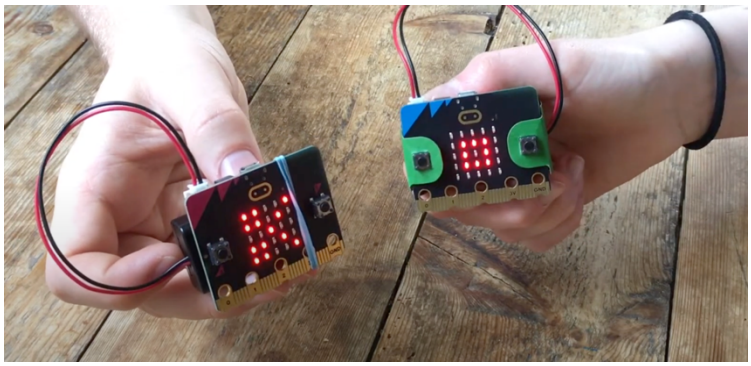
This is how it should look when completed:



By bringing this project to life, you'll not only gain hands-on experience with coding and problem-solving but also contribute to a **meaningful cause**. Together, we can make a positive impact on the environment and play a small yet vital role in protecting sea turtles and their natural journey. Be proud—you're helping save the sea turtles! 🌊🐢

Rock-Paper-Scissors Game

Overview



The **Rock-Paper-Scissors Game** transforms the Micro:bit into a fun, interactive platform that introduces programming and decision-making. This project addresses Logical Systems and fosters creativity while exploring randomness and probability. Learners gain hands-on experience applying STEM concepts to build a game that combines logic, interaction, and excitement.

Goals Addressed

- **Goal 1. Logical Systems**
  - Design decision-making algorithms to interpret random outcomes and display corresponding icons.
- **Goal 14. Sensor Integration**
  - Utilize button inputs as triggers for gameplay.
- **Goal 15. Product Prototyping**
  - Build and test a functional, user-friendly game prototype.
- **Goal 16. Pitching Skills**
  - Present and explain the project mechanics and logic to peers or an audience.

Why It's Fun

- **Interactive Gameplay**: Engage in a timeless game with a tech twist, combining luck and strategy.
- **Customizable Features**: Add unique icons, sounds, or messages for a personalized gaming experience.
- **Learning Through Play**: Develop essential coding and logical thinking skills while having fun.

Rock-paper-scissors Game

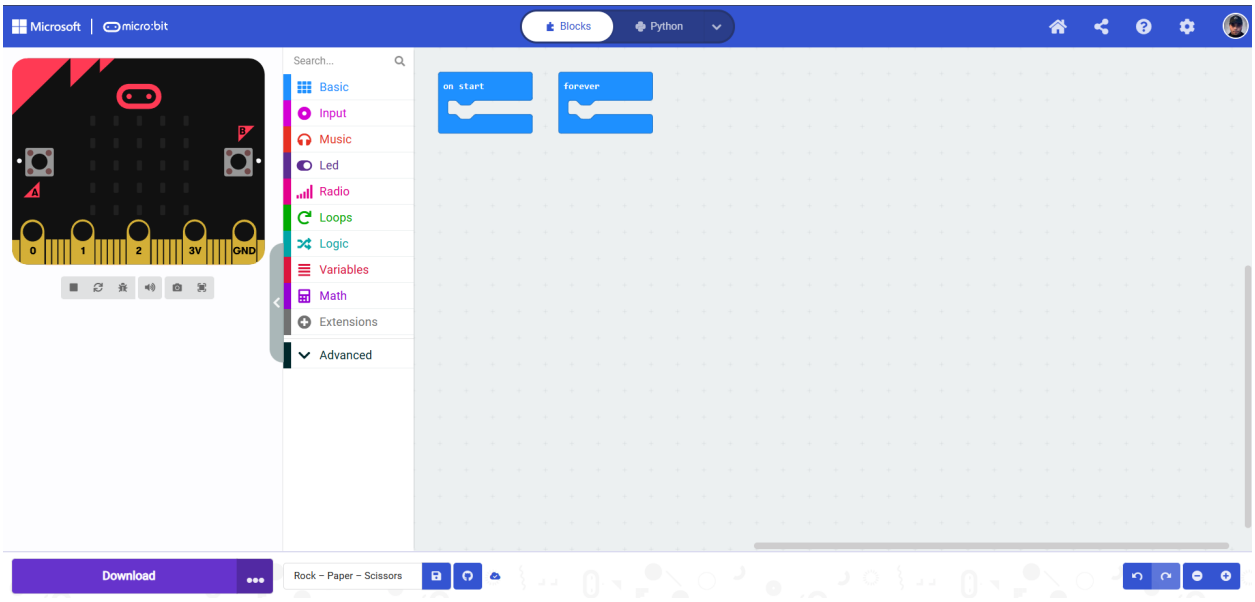
This Guide on How to Create the Rock-Paper-Scissors Game with Micro: bit
The Project Idea

This project introduces programming fundamentals through an exciting Rock-Paper-Scissors game. Using the Micro: bit, players can randomly generate outcomes and test their luck in a fun and interactive way. The game encourages creativity while building essential coding skills.

• Button A generates a random number representing Rock, Paper, or Scissors.
• The Micro: bit displays icons for Rock, Paper, or Scissors based on the random outcome.
• Players compare their choices to determine the winner.

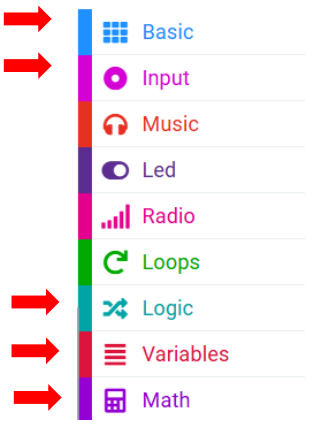
To implement this project, we will use the MakeCode Editor by Microsoft.

- **Step 1:**
  Search for MakeCode Editor in google:
  https://makecode.microbit.org/#editor
- **Step 2:**
  Click on new project and name it: Rock – Paper – Scissors
  Then this screen will appear



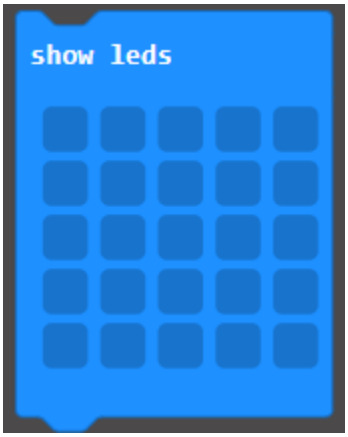
- **Step 3:**

  We just will use this command Groups:



  From Basic :

  We need 3 of

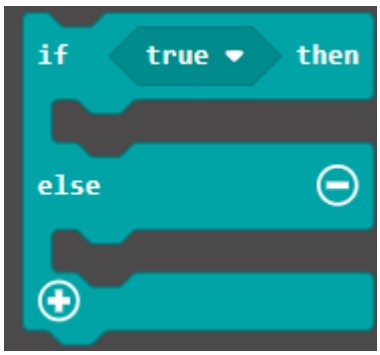


  From Input :

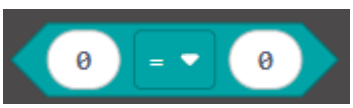
  We need 3 of

  From Logic :

  We need :



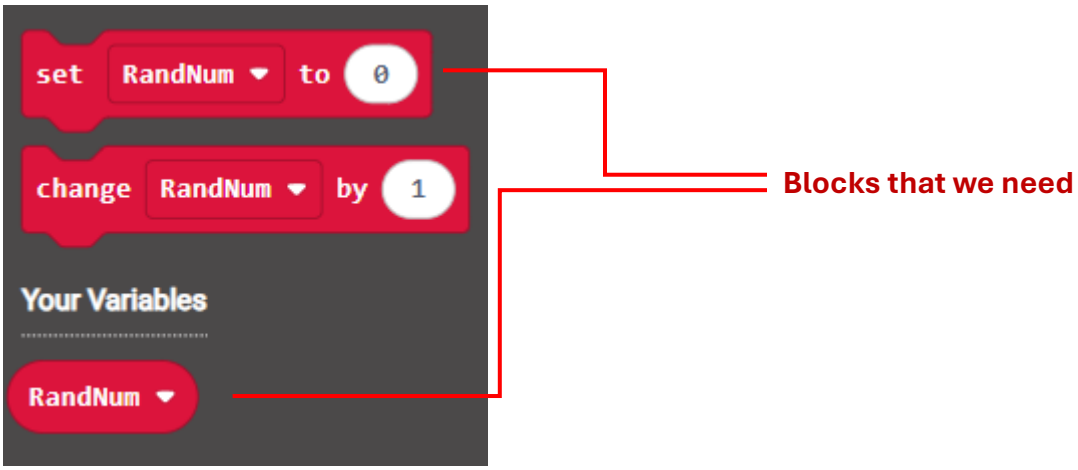
  AND



From Variables :

Go to click on Make a Variable... button and name a variable “Random_Number“ or “RandNum” For Short then click **OK**

These Blocks will appear:

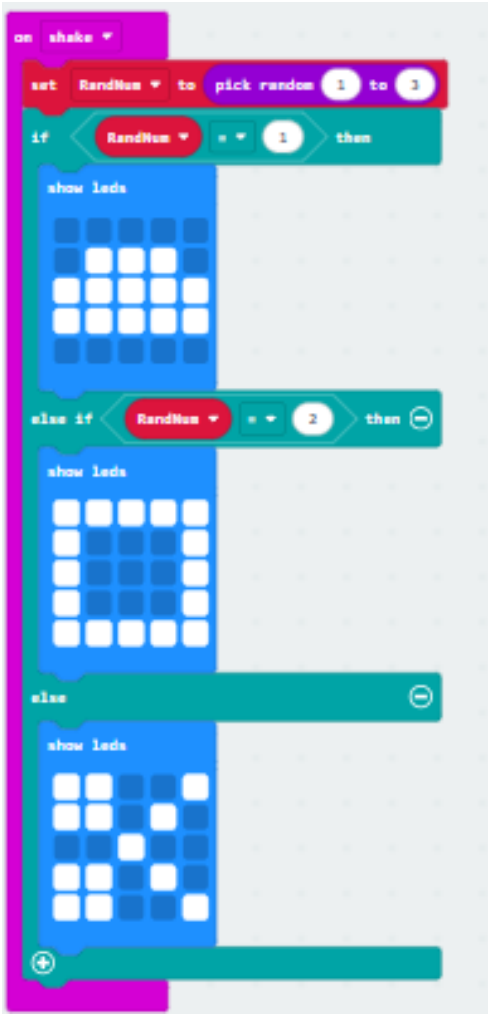


From Math :

Scroll down until you find **(Pick Random)** Block

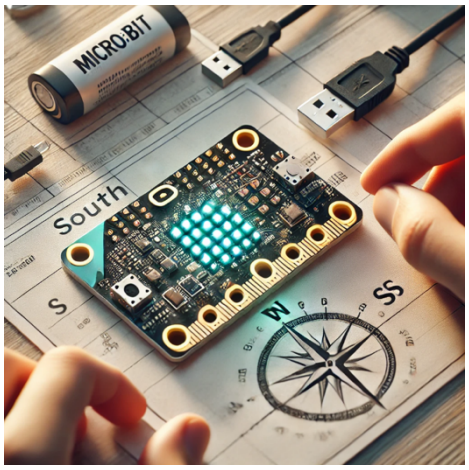
Set the range of numbers From 1 to 3

This is how it should look when completed:



By diving into the Rock-Paper-Scissors project, you're transforming basic programming into a fun and engaging challenge that **sparks competition** and **creativity**. This activity blends logic, randomness, and excitement, helping you develop essential coding skills while enjoying a classic game. Be proud—you're turning coding into a playful journey of innovation and interaction! ✊✋✌️🎉

Digital Compass



Overview

The **Digital Compass project** transforms the Micro:bit into a practical, pocket-sized navigation tool. By leveraging the built-in magnetometer, the compass displays cardinal directions (North, East, South, West) on the LED matrix. This project introduces learners to Logical Systems, Sensor Integration, and Prototyping, offering real-world applications in coding and hardware.

Goals Addressed

- **Goal 14. Sensor Integration**
  - Utilize the Micro:bit's built-in magnetometer to detect Earth's magnetic field and provide real-time feedback.
- **Goal 15. Product Prototyping**
  - Build and test a fully functional digital compass prototype for practical navigation.
- **Goal 9. Education Challenges**
  - Use the project as an educational tool to teach magnetism, coding, and sensor applications in real-world scenarios.
- **Goal 17. Business Strategy**
  - Explore possibilities for scaling this project into affordable, portable navigation tools for outdoor enthusiasts.

Why It's Fun

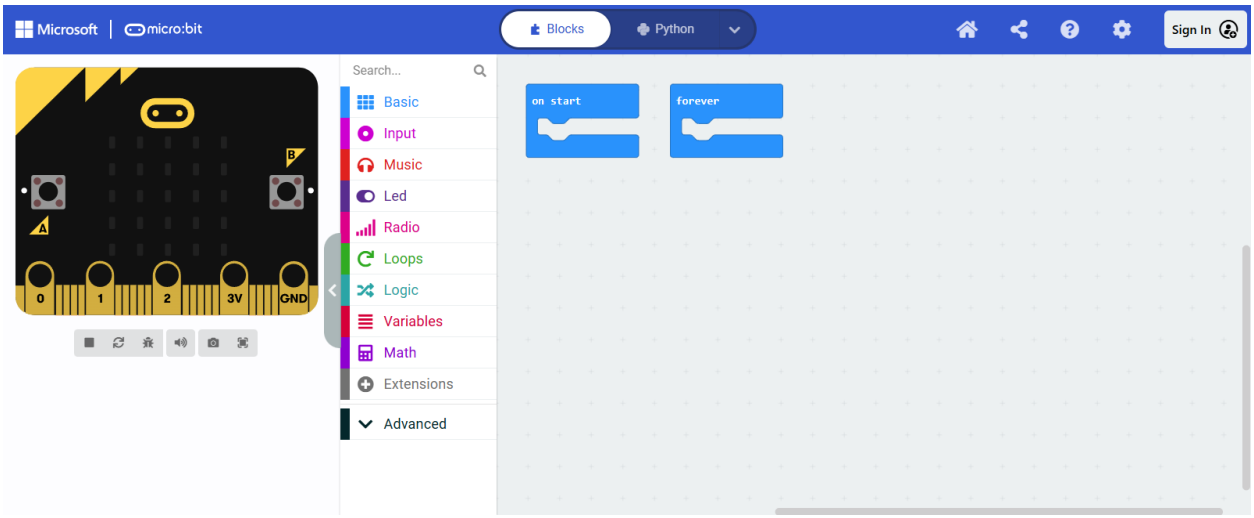
• **Hands-On Exploration**: Gain practical experience in coding, sensor integration, and environmental navigation.
• **Real-World Applications**: Build a device that can be used in outdoor adventures and educational activities.
• **Customizable Features**: Add creative animations, sounds, or extra functionality to enhance the compass's usability.

Steps to create a digital compass

- **Step 1**

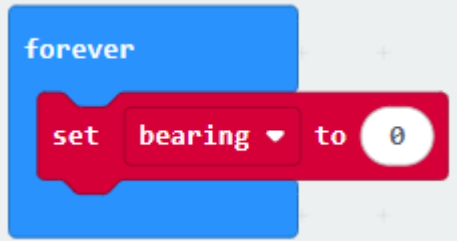
  Go to MakeCode website start a new project and Name it

  "Digital Compass" Click OK then this window will appear:



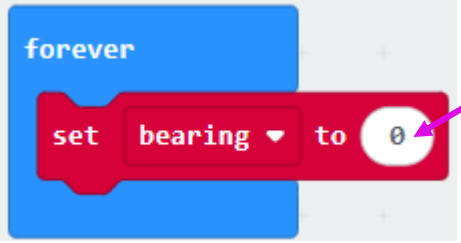
- **Step 2**

  Remove On start block right click on this block and choose "Delete Block", And from Variables click on Make a Variable... and name the variable "bearing" click OK and choose set bearing to 0 drag it inside [forever] block like this:

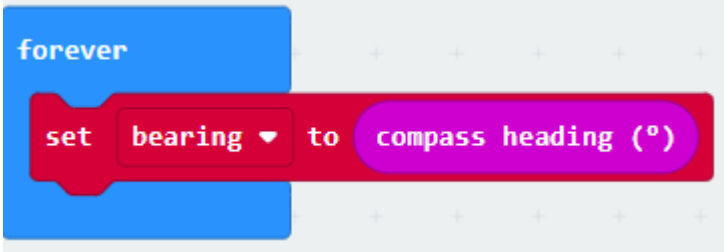


- **Step 3**

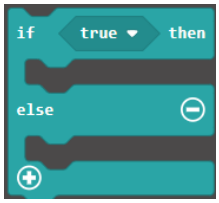
  from Input choose compass heading (°) drag it inside here:



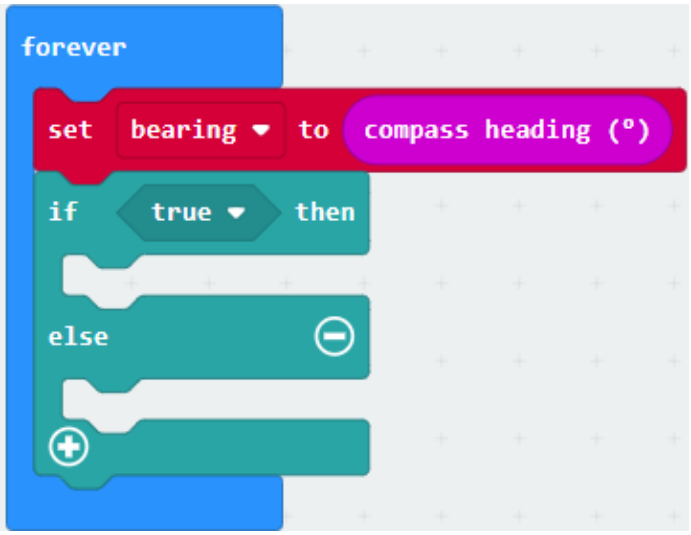
To be like this :



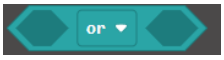
- **Step 4**

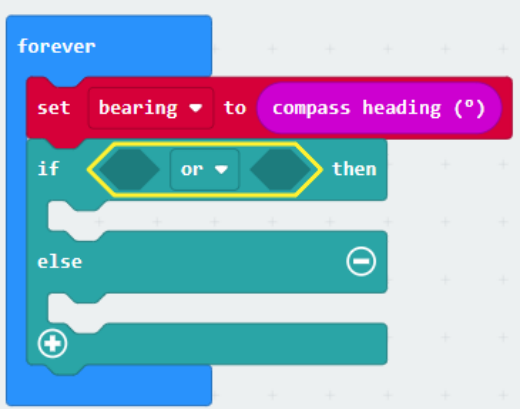
From Logic choose 

drag it below the [set] block like this:

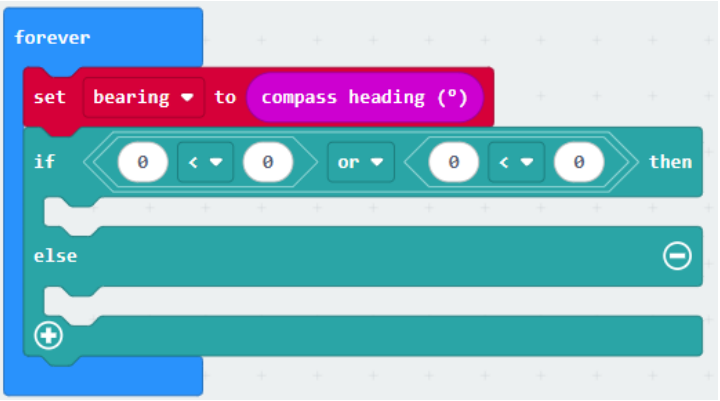


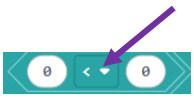
- **Step 5**

From Logic choose  drag it inside the “true” like this:

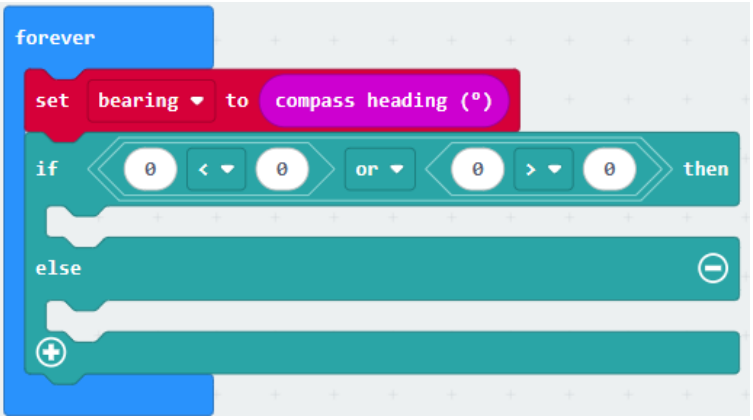


Also from Logic choose 2 of  but the first one in the right side and in [or] block and the other one in the left side like this:

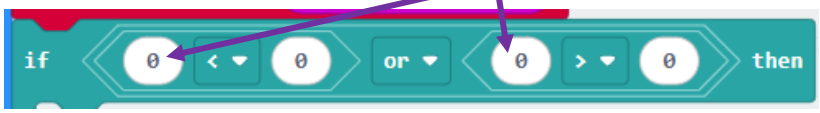


Chage the comparison sign of the left side to be larger than from here 

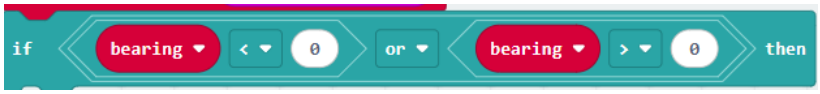
Choose > to be like this:



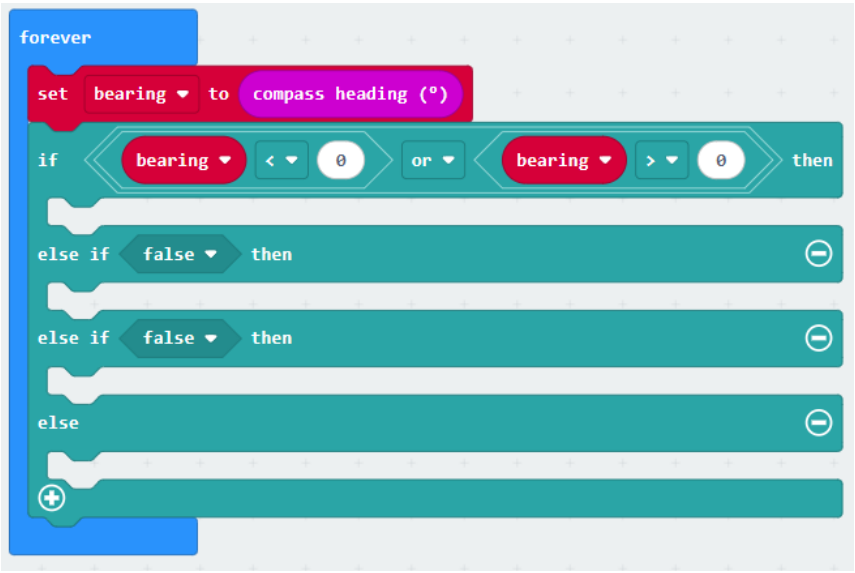
Now go to Variables choose 2 of bearing drag them here :



To be like this:

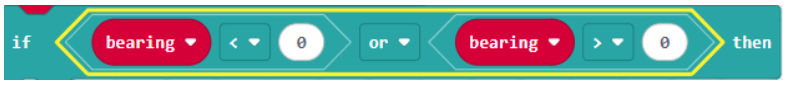


click the ⊕ sign in the [if] block 2 times, the else if will appear and the blocks will be like this :



- **Step 6**

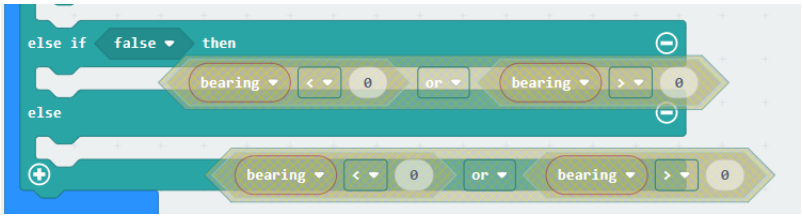
  Now select the [or] Block, a yellow boundary will appear like this :



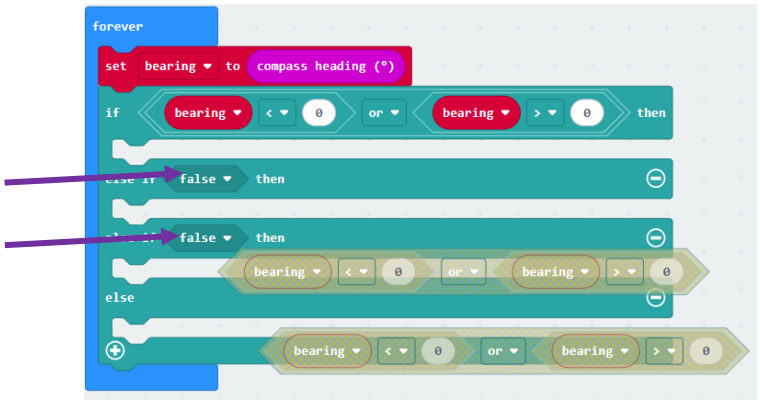
  Now click CTRL + C to Copy it

  Then paste it 2 times by clicking on CTRL + V

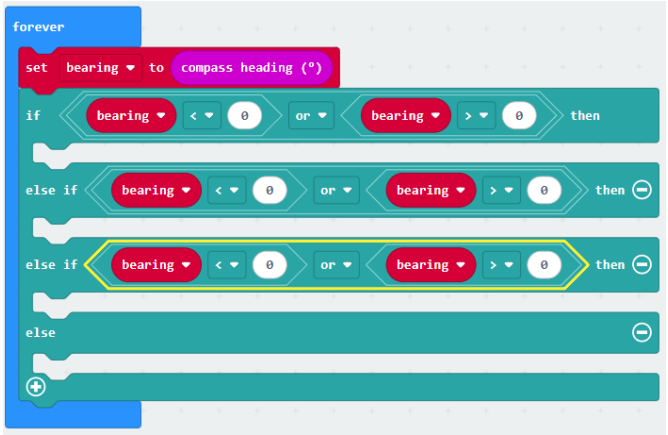
  These 2 copies will appear:



Drag them here:

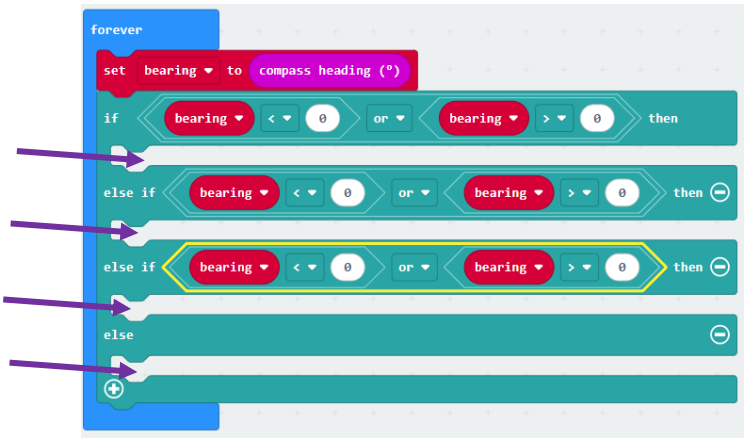


To be like this:

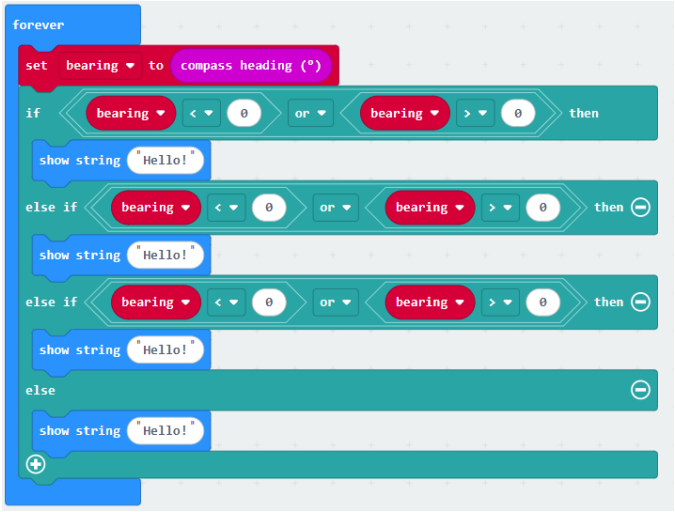


- **Step 7**

  From Basic choose 4 of show string "Hello!" and place them here



To be like this:



Now change the “Hello” by clicking on “Hello” then you can change it by the cardinal directions (North, East, South, West):

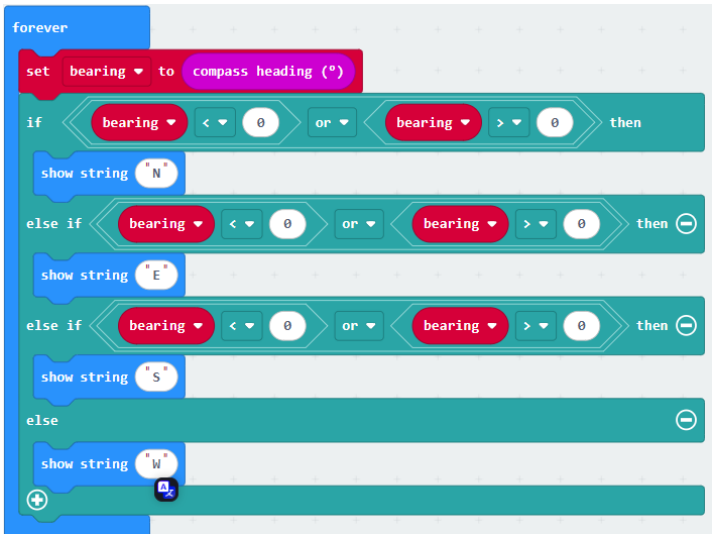
Change the first one by: N

Change the second one by: E

Change the third one by: S

Change the forth one by: W

Like this:



Now we need to set the bearing ranges of cardinal directions

45(North, East, South, West):

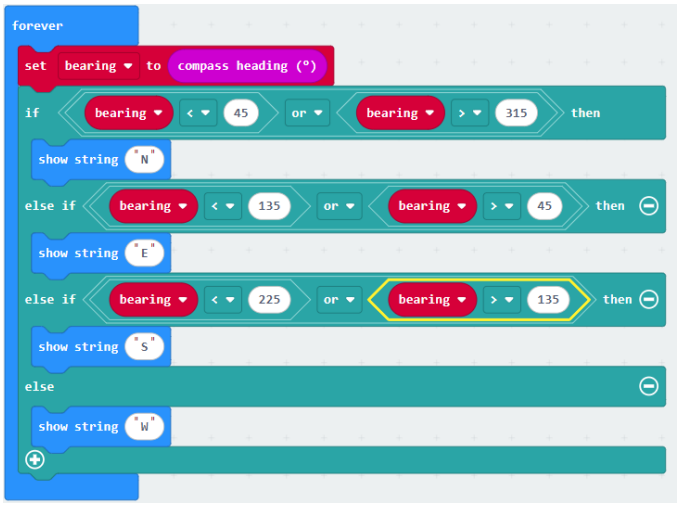
For the North range is less than 45 or more than 315

For the East range is less than 135 or more than 45

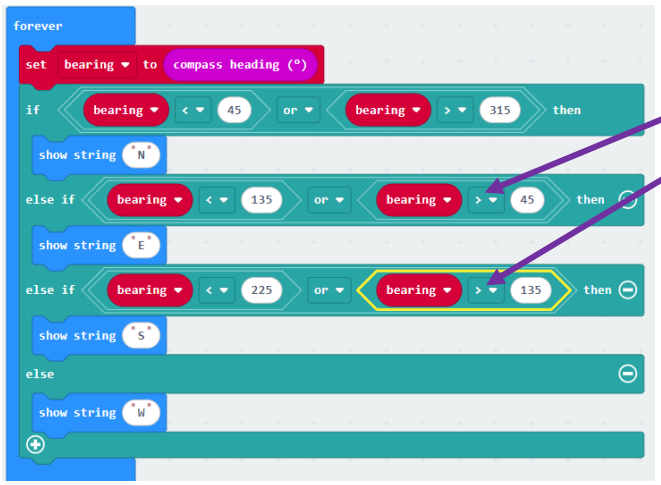
For the South range is less than 225 or more than 135

For the West the range is less than 315 or more than 225

It should be like this:



We have reached the final step on this project, Change those signs to “Equal to” =



This is how it should look when completed:

By creating the Digital Compass project, you're turning advanced concepts like magnetism and sensor integration into an accessible and exciting hands-on experience. This activity combines logical thinking, creativity, and real-world applications, helping you develop essential **coding** and **hardware skills.** Be proud—you're transforming coding into an innovative journey of exploration and navigation! 🧭✨

Steps Tracker

Overview

The Steps Tracker project transforms the Micro:bit into a fun and interactive step counter. By detecting shakes, it counts steps and displays the total on its LED screen. Users are rewarded with a cool flashing heart animation for every 20 steps. This project utilizes the Microsoft MakeCode Blocks editor, making it simple and accessible for all skill levels.

Goals Addressed

- **Goal 1: Logical Systems**
  - Program the Micro:bit using Blocks from the Microsoft MakeCode editor to track the number of steps.
- **Goal 15: Product Prototyping**
  - Build a fully functional prototype for step tracking.
- **Goal 9: Education Challenges**
  - Use the Steps Tracker as a teaching tool to encourage students to combine physical activity with coding and technology.
- **Goal 17: Business Strategy**
  - Explore potential for scaling this concept into a fitness tracker product, considering cost, functionality, and marketability.

Why It's Fun and Useful

• **Visual Feedback**: The Micro:bit's LED display makes it enjoyable to track progress in real-time with a clear, visual representation of steps.
• **Simple and Creative**: The MakeCode Blocks editor is beginner-friendly, enabling users to experiment and customize their projects without complexity.
• **Customizable**: Add personal touches, like changing animations, step thresholds, or incorporating sounds to make the tracker uniquely your

Steps Tracker

Steps to Create the Environment Exploration Tool

Step 1: Assemble Your Tools

- A Micro:bit.
- A battery pack for portability.
- Access to the or Python editor.

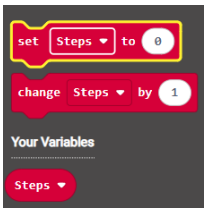
Step 2: Code Your Micro:bit

1. **We need to set our blocks:**
   from Basic :

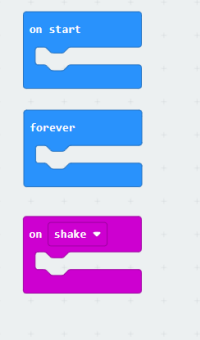
   from Input :

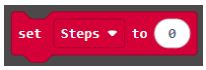
   from Variables we need to set a new variable by click on Make a Variable... called "Steps" then click Ok

   Now New Block appears:

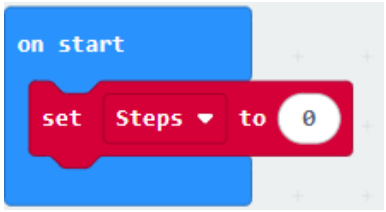


   Now The Blocks space should be like this:



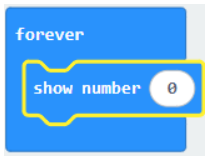
2. **Set the Steps by Zero**:
   From Variables Chose 

the drag it into [On Start] Block to be like this:



3. **Show the number in the LEDs:**
   From Basic Chose 

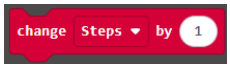
then drag it into [Forever] Block to be like this



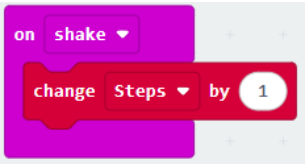
   and From Variables chose 

then drag it into the Zero inside [Show Number] Block to be like this:

4. **Setting up the shacking command:**

   From Variables chose 

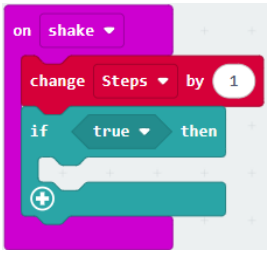
drag it into [Shack] Blockto be like this:

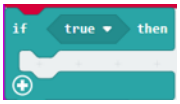


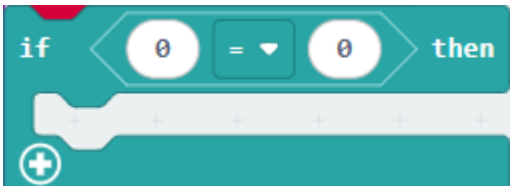
   From Logic chose 

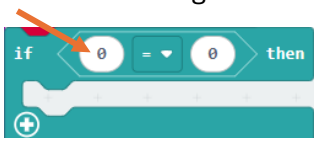
drag it into [Shake] Block, under 

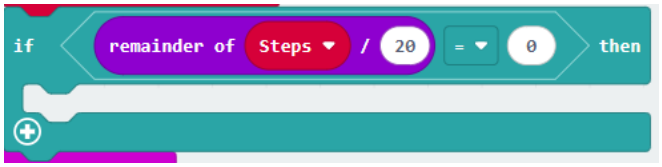
like this:

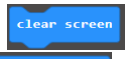
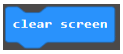
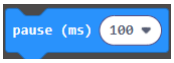
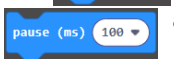
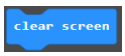


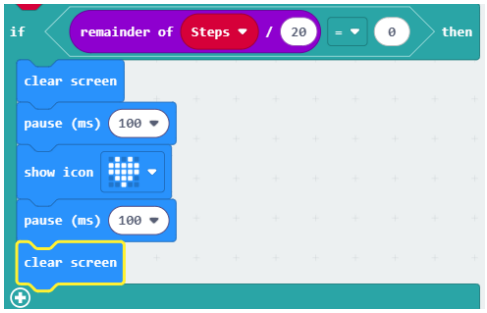
From Logic chose  drag it into “true”  to be like this:



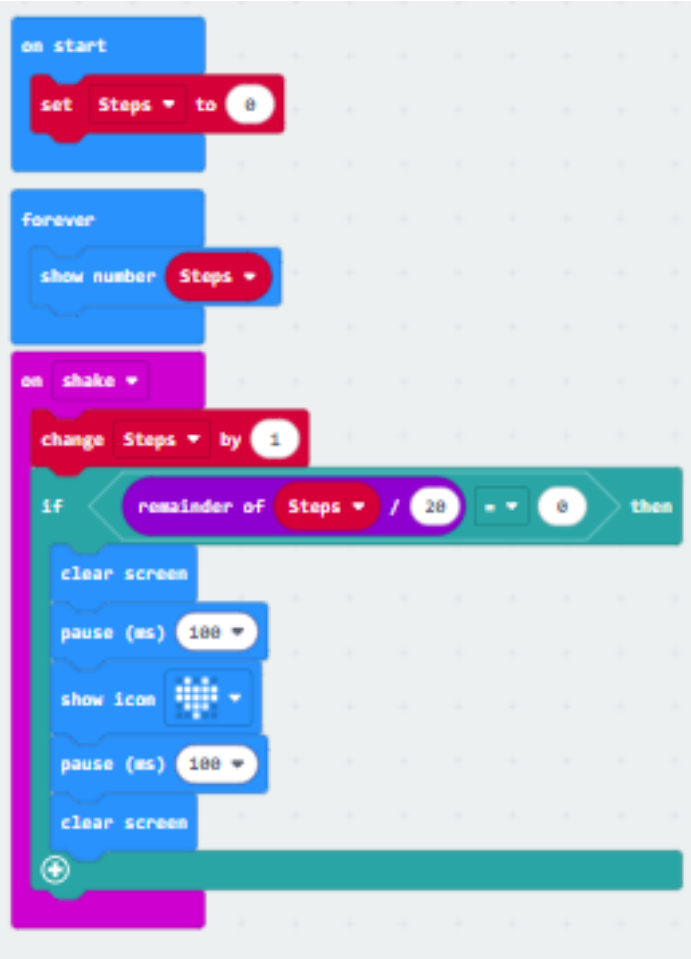
From Math chose  but inside the 0  variable and change number 1 to 20 to be like this  then drag it here  to be like this:



From Basic chose  drag inside [if] Block, under  drag  , then drag  and  and  to be like this :



This is how it should look when completed:



By diving into the Steps Tracker project, you're turning simple motion detection into an exciting and rewarding activity. This project blends creativity, logic, and real-time interaction, helping you develop essential coding skills while **promoting physical activity**. Be proud—you're transforming programming into a fun and engaging journey of innovation and movement! 🚶‍♂️🚶‍♀️✨