



# Climate Action Kit

## Using Algorithms & Flowcharts

### **YOUR TASK**

Create an algorithm and flowchart to help plan and debug your code using a project from the Climate Action Kit.

# 1. ALGORITHMS



# Understanding Algorithms

An **algorithm** is a step-by-step process or rules to solve a problem or complete a task. Algorithms are like recipes! They use data (like ingredients) as input and specific steps to transform the input into our desired *output (our final meal)*.

Algorithms don't need to be run by a computer. It helps us create a blueprint to describe what a program must do before we translate it into a coding language for the computer to read.

Algorithms are the first step to coding!

## Real World Application

The more specific the instructions in the algorithm are, the better.



For example, option B is a better example of an algorithm if we are making a cake.

- A. Crack some eggs into a bowl.
- B. Crack three eggs into a bowl.

# Algorithms: Lists and Instructions

Using the [coastal flood alarm lesson](#), let's create an algorithm to help understand how we would plan our code during the create phase.

This project uses the following materials:

micro:bit V2	<a href="#">LED Ring</a>	Watertight container
<a href="#">Breakout Board</a>	<a href="#">Moisture Sensor</a>	Water

## Example: Flood Detection with Coastal Flood Alarms

The coastal flood alarm needs to detect how high the water is, and change the colour of the LED ring depending on if the water is at high or low tide.

Using this project objective, we've created the following algorithm in the space below:

1. Turn on micro:bit
2. Sense the amount of water in a container
  - If it is low tide (the container is  $\frac{1}{3}$  full)
    - Turn the LED ring to green
  - If it is high tide (the container is  $\frac{1}{3}$  -  $\frac{2}{3}$  full)
    - Turn the LED ring to yellow
  - If there is a flood (the container is more than  $\frac{2}{3}$  full)
    - Turn the LED ring to red
    - Play an alarm
      - Play a sound for one second
      - Pause for two seconds

# Algorithms: **Input & Output**

Now that we've created an algorithm, we can use it to understand how the computer will sense and *display* data when we translate it into code.

Inputs are information that the computer receives as part of their program. This includes lots of things like sensor data from the Climate Action Kit and pressing a button on the micro:bit.

Outputs are actions that the computer performs to give feedback to the user. This could include making sounds or changing the colour of an LED on a micro:bit.

## Example: Identifying Inputs & Outputs

Using the algorithm we've created in the previous step, we've highlighted all of the inputs (data) and the *outputs (how the micro:bit responds)*.

1. Turn on micro:bit
2. Sense the amount of water in a container
  - If it is low tide (the container is  $\frac{1}{3}$  full)
    - Turn the *LED ring to green*
  - If it is high tide (the container is  $\frac{1}{3} - \frac{2}{3}$  full)
    - Turn the *LED ring to yellow*
  - If there is a flood (the container is more than  $\frac{2}{3}$  full)
    - Turn the *LED ring to red*
    - Play an alarm
      - *Play a sound* for one second
      - Pause for two seconds

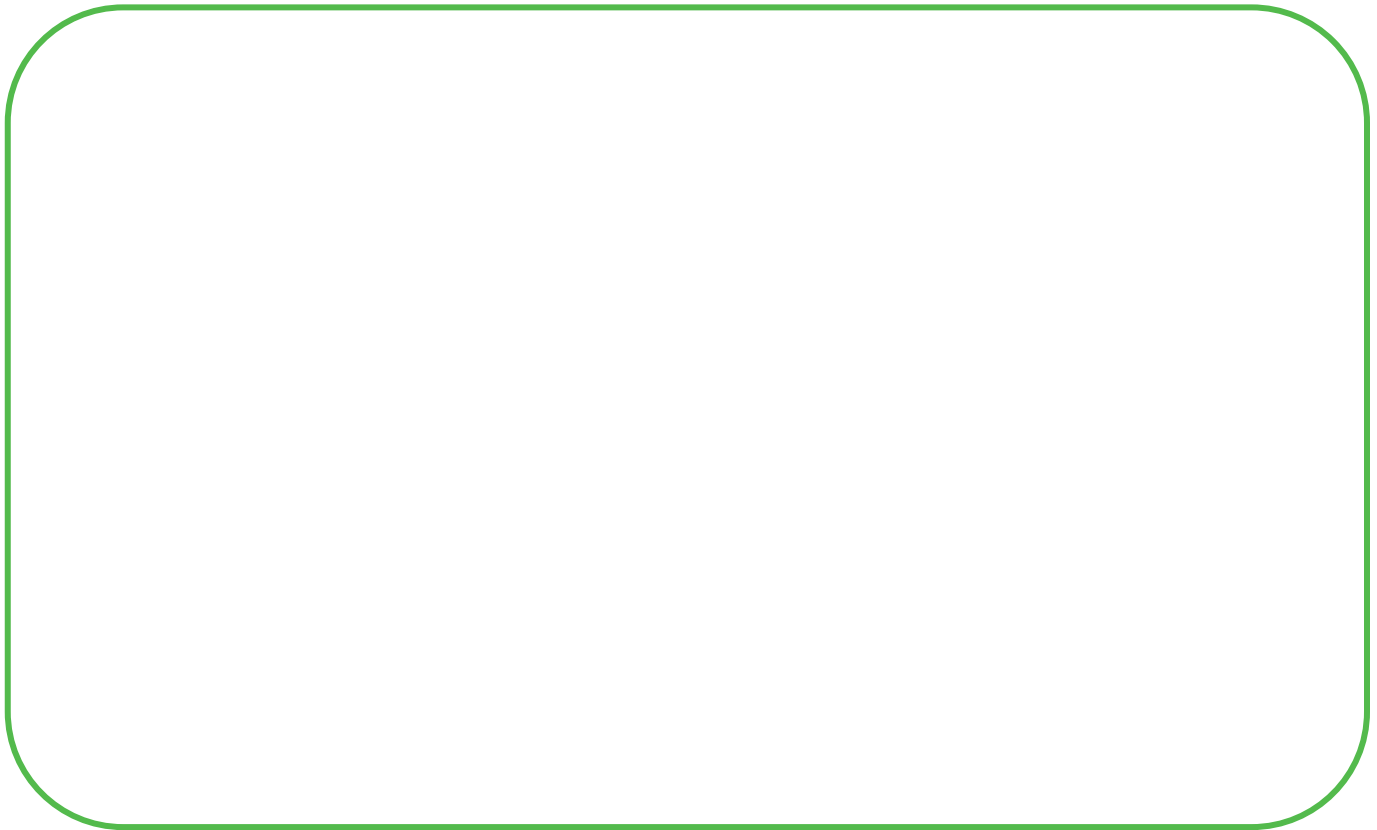
# Make your Algorithm

## ACTIVITY

Now it's your turn! In the space below, try making your algorithm for another project from the [project library](#), or practice using one of the following ideas:

Your morning routine	How many steps to exit the room you are in
Drawing a rectangle on a piece of paper	How to make lemonade

When you're done creating your algorithm, highlight all inputs (data) in one colour and the *outputs (how the micro:bit responds)* in a second colour.



### TIPS

1. Try reading your instructions to a partner or follow the instructions yourself to see if you need to clarify or add any steps.
2. Write down each step on sticky notes, then you can rearrange them as you edit your algorithm.

## 2. FLOWCHARTS









# Understanding Flowcharts

**A flowchart** is a type of diagram that shows the step-by-step process of an algorithm. The symbols used in flowcharts have specific meanings, like a process, a decision, input, and output.

Flowcharts don't need to be run by a computer. It helps us describe the information the program will use before we translate it into a coding language for the computer to read.

## Flowchart Symbols

Symbol	Definition	Example
	The start of the program, this is always the first symbol.	START
	The arrow points in the flow of the program, including loops in a program.	Connects each symbol
	Used when collecting input from the user or sensors.	Press the A button Sense moisture
	Used when displaying information to the user.	Set LED colour Play sound
	Instructions that are completed by the program that don't involve user input.	Create a variable Wait 1 second
	Used each time the program has to make a choice (if/else, true/false).  Multiple arrows branch off this symbol, with labels describing each decision criteria.	True, False  If less than 33%



# Algorithms to Flowcharts

## ACTIVITY

Let's make a flowchart using the algorithm we created for the [Flood Detection with Coastal Floods Alarm lesson](#)!

Flowcharts use a few more criteria to describe a program than algorithms.

### Instructions

When you created your algorithm you highlighted all the **inputs** and **outputs** in your program. Using the example provided, identify the **{start}**, **[processes]**, and **<decisions>** on the algorithm you created earlier.

1. **{Start}** Turn on micro:bit
2. **Sense** the amount of water in a container
  - **<Decision>** If it is low tide (the container is  $\frac{1}{3}$  full)
    - Turn the **LED ring to green**
  - **<Decision>** If it is high tide (the container is  $\frac{1}{3} - \frac{2}{3}$  full)
    - Turn the **LED ring to yellow**
  - **<Decision>** If there is a flood (the container is **more than  $\frac{2}{3}$  full**)
    - Turn the **LED ring to red**
    - Play an alarm
      - **Play a sound** for one second
      - **[Process]** Pause for two seconds

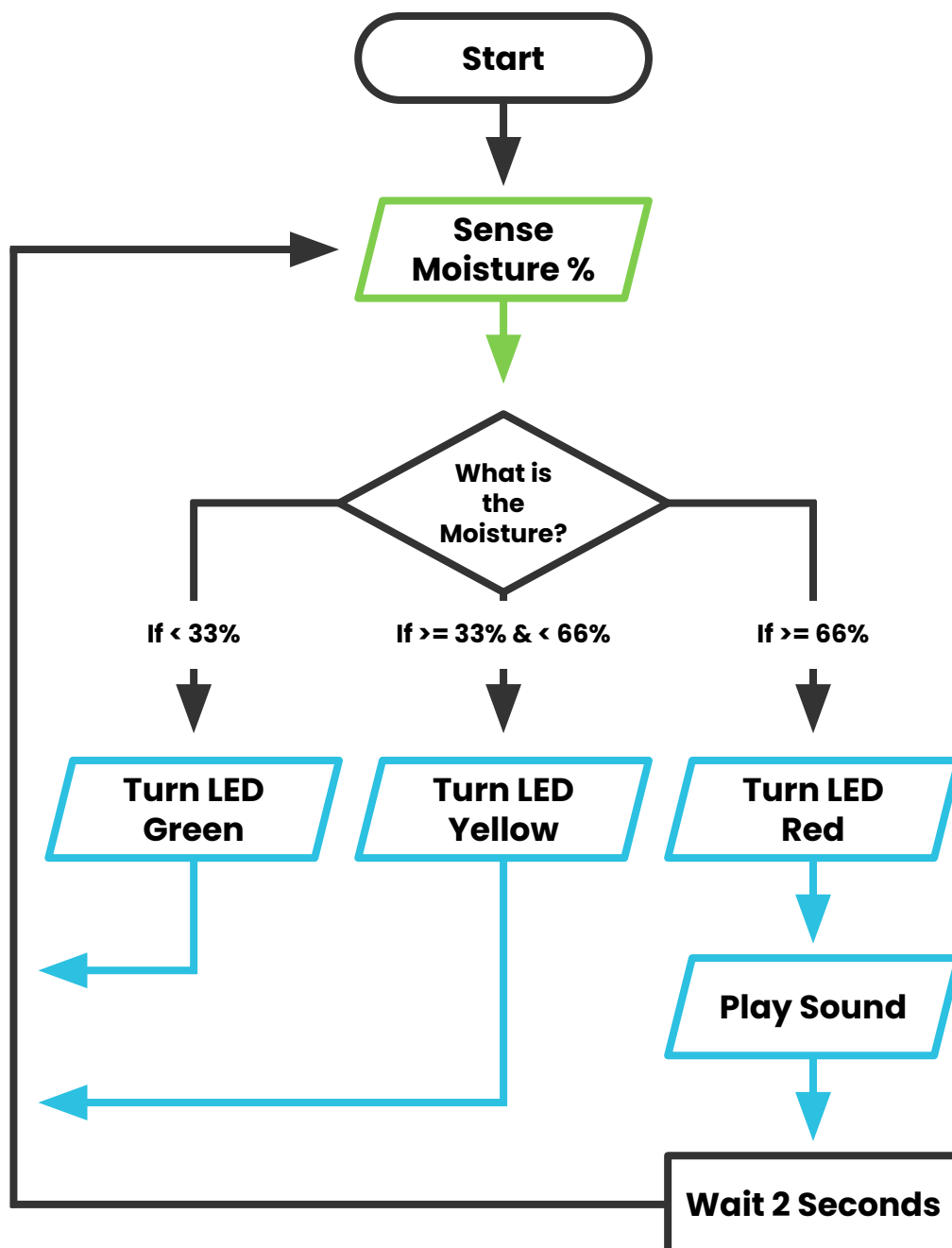
### TIP

Use the table on the previous page to remember the difference between a **[process]** and a **<decision>**.

# Flowchart Example

## ACTIVITY

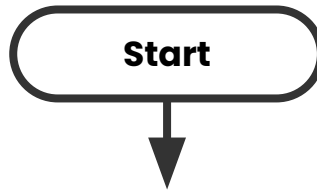
Now that we've identified all the data from our algorithm, we can create our flowchart!



# Make your own Flowchart

## ACTIVITY

1. Using the algorithm from your previous activity, identify the **{start}**, **[processes]**, and **<decisions>** in each step.
2. Use the space below to create your own flowchart! Remember to refer to your algorithm and the table of symbols used in flowcharts.



### TIPS

1. Visualize the steps of your flowchart first, then figure out which shape it should be.
2. Write down each step on sticky notes, then you can rearrange them as you edit your flowchart.

# 3. CODING

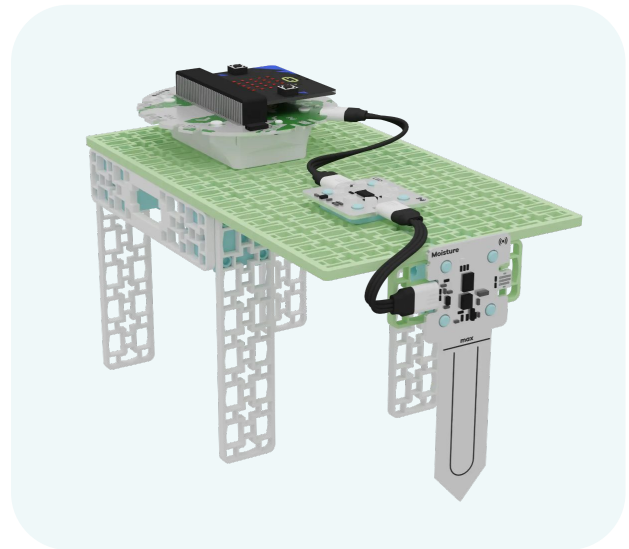


# Translating to Code

## ACTIVITY

Algorithms and flowcharts are two great tools to help you during the Modify and Create phases of a lesson. Depending on how you prefer to learn, you can make an algorithm, a flowchart, or both!

We've used the algorithm and flowchart activities to [create the code](#) for the Flood Detection with Coastal Floods Alarm lesson.



## Instructions

Using this sample [project template](#), transform your flowchart and algorithm into a working program! Once you're done, download your code to your micro:bit and try it out.

## TIPS

1. Look at each step in your flowchart or algorithm one by one
2. Download your code to your micro:bit each time you change your code or add a step
3. Work with a partner, one of you reads each step of the algorithm or flowchart, and the other can find the blocks in the code space.