Grade 5/6

Micro:Bits

Any questions?
Reach out
svcamp@engr.uvic.ca

Micro:Bit Arrays

Learn about arrays in coding by programming a Micro:bit. Use arrays to implement guessing games and musical note selection. This lesson introduces the fundamental concept of storing and retrieving data in an ordered fashion using Arrays. We'll look at the structure of a Melody as a list of notes.

Key Words

Element: An element of the array is an item of the collection

Length: the total number of items in the collection

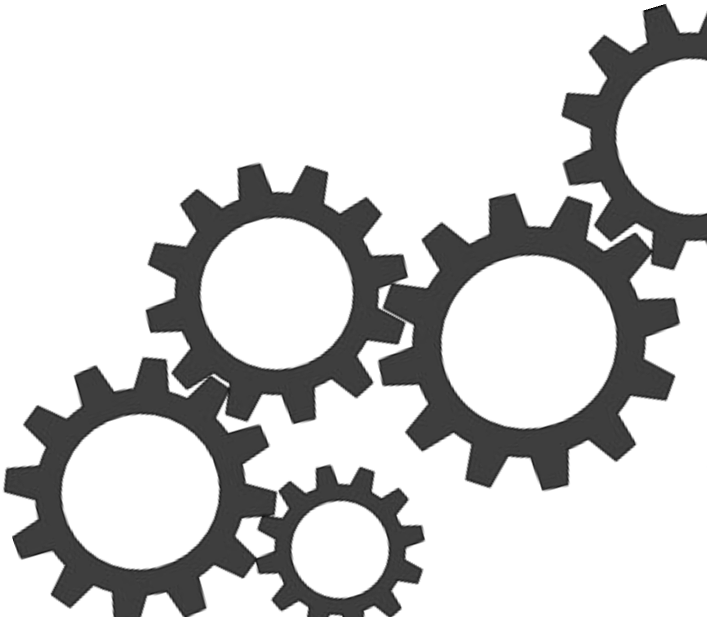
Index: A unique address or location in the collection. The index of the last element in an array is always one less than its length (because the array numbering starts at zero.)

Type: The type of item being stored in the collection

Sort: Items in the collection are ordered by a particular attribute (e.g., date, price, name)

Materials

- Micro:bit kit
    - 1 Micro:bit
    - 1 Battery pack
    - 1 Micro USB cable
- Optional: 1 Dongle adapter if using Mac



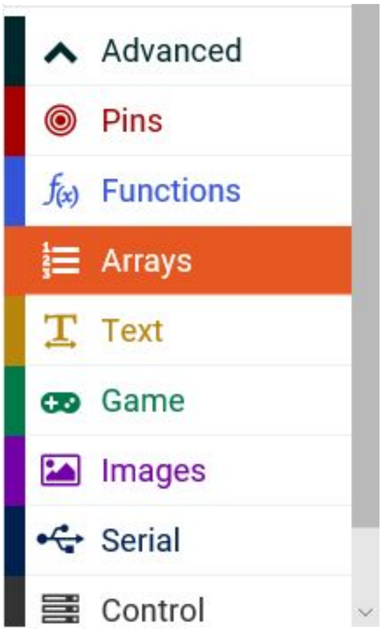
Micro:Bit Arrays

An array as a collection; each item needs its own storage space, and a unique address so you can find it later. Arrays can store numbers, strings (words), or sprites. They can also store musical notes.

- Do any of you collect anything? What is it? Comic books, cards, coins, stamps, etc.
- How big is the collection?
- How is it organized?
- Are the items sorted in any way?
- How would you go about finding a particular item in the collection?

As your MakeCode programs get more and more complicated, and require more variables to keep track of things, you will want to find a way to store and organize all of your data. MakeCode provides a special category for just this purpose, called an Array. Every spot in an array can be identified by its index, which is a number that corresponds to its location in the array. The first slot in an array is index 0.

In MakeCode, you can create an array by assigning it to a variable. The Array blocks can be found under the Advanced Toolbox menu.



Micro:Bit Arrays

Activity 1

Musical Arrays

In this project you will create a musical instrument that uses arrays to store sequences of notes. The array of notes can be played when an input occurs, such as one of the buttons being pressed.

Attach 2 alligator clips to your micro:bit pin inputs to attach headphones to hear your sounds.

- Take a look at this video (starting at 3:43) to see how to attach your alligator clips to headphones.

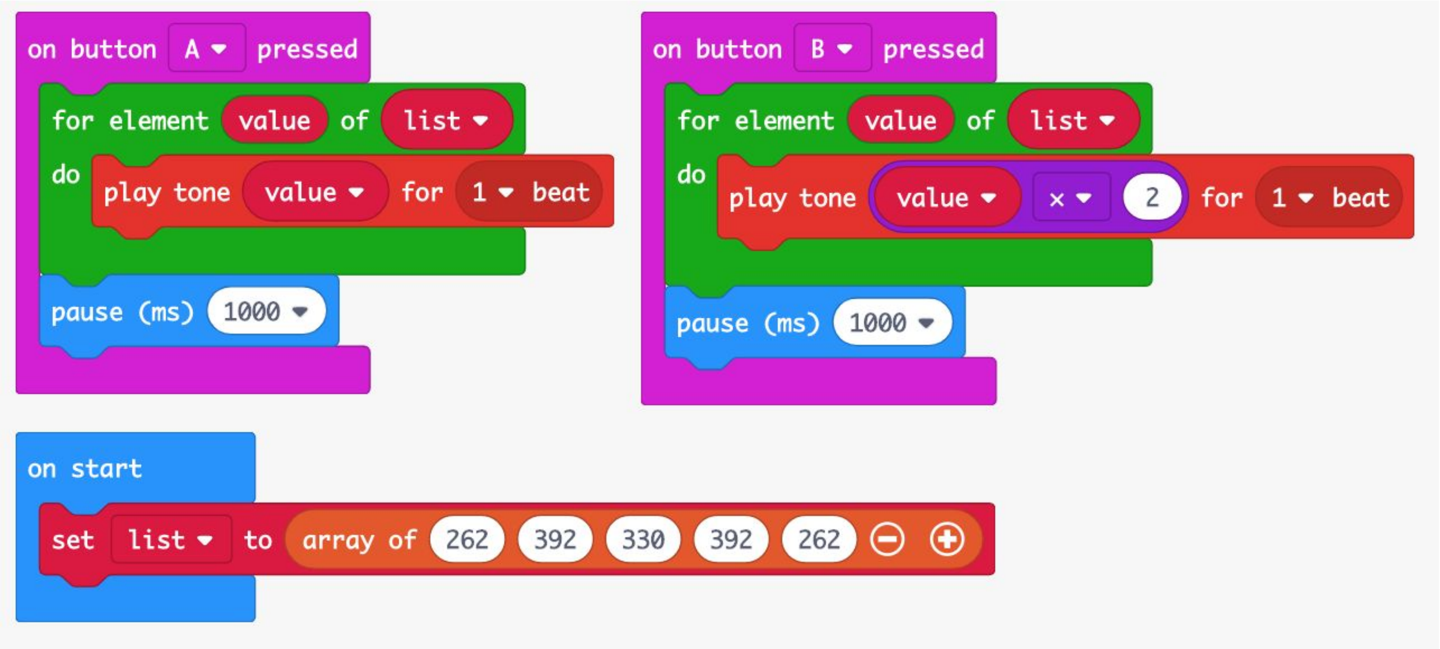
Have campers test a simple tone block first to be sure they have wired their headphones correctly to the Micro:bit pins. Experiment with the musical melodies and default sounds available. Campers can add their favorite tones to an array for use in the next part.

Using arrays with musical notes

You can create an array of notes by attaching Music blocks to an array. Musical notes are described in words (e.g., Middle C, High C) but they are actually numbers of different frequencies. You can do Math operations on those numbers to change the pitch of your song.

Micro:Bit Arrays

Here is an example of how to create an array with musical notes. Button A plays every note in the array. Button B plays the notes at twice the frequency (but doesn't alter the original notes.)

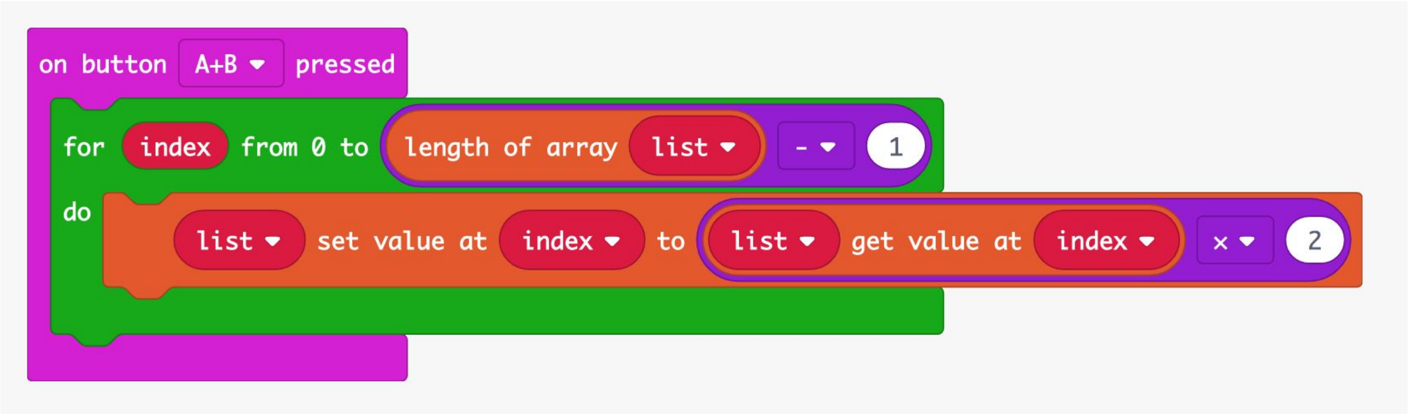


```
on button A ▾ pressed
    for element value of list ▾
    do
        play tone value ▾ for 1 ▾ beat
    pause (ms) 1000 ▾

on button B ▾ pressed
    for element value of list ▾
    do
        play tone value ▾ × ▾ 2 for 1 ▾ beat
    pause (ms) 1000 ▾

on start
    set list ▾ to array of 262 392 330 392 262 ⊖ ⊕
```

Remember that a 'for element value of list' loop makes a temporary copy of the value, so even if you change a value, it will not change the original element in the array. If students want to permanently change the values in their array (transpose music to increasingly higher keys, for example) they can use a for loop like this:



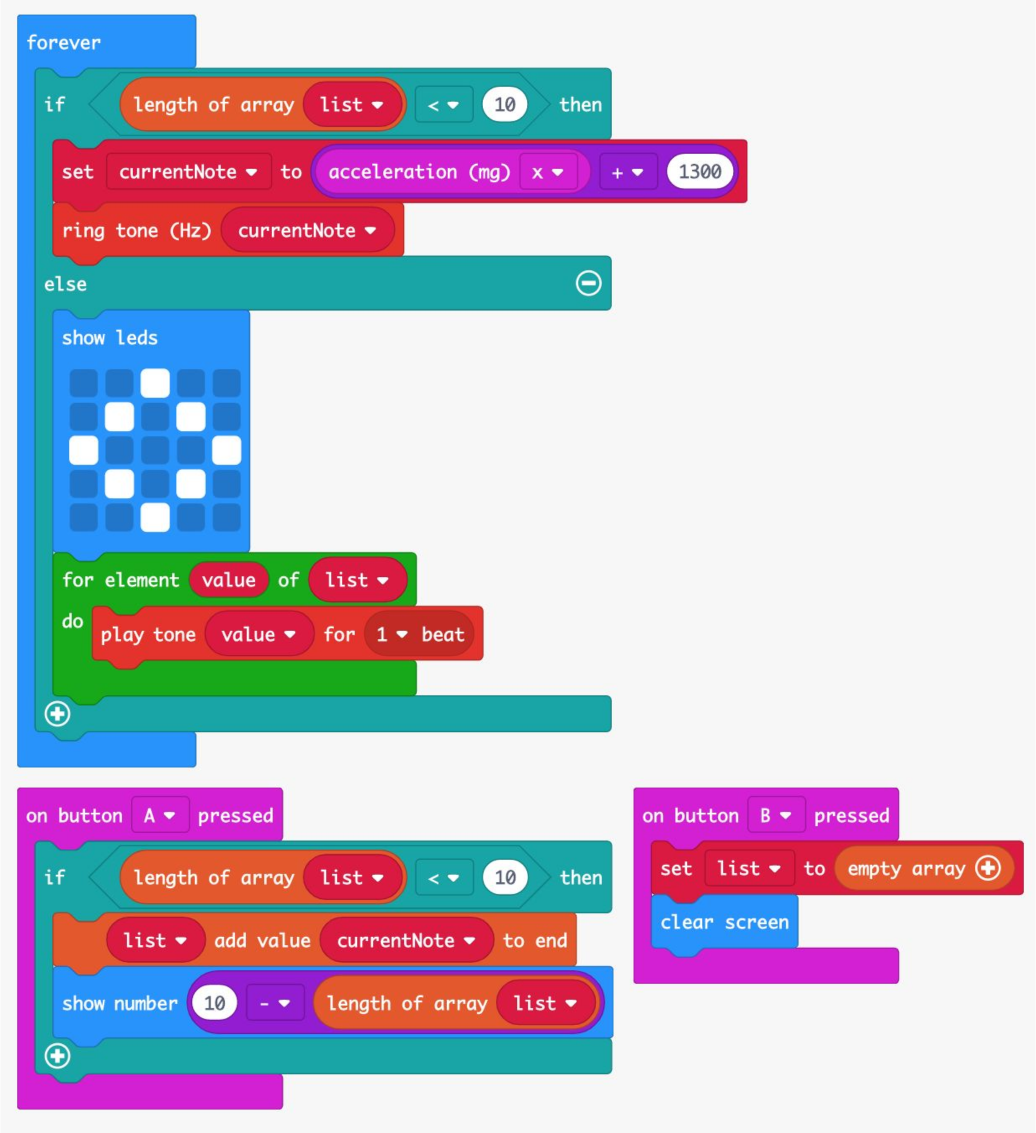
```
on button A+B ▾ pressed
    for index from 0 to length of array list ▾ - ▾ 1
    do
        list ▾ set value at index ▾ to list ▾ get value at index ▾ × ▾ 2
```

Micro:Bit Arrays

Activity 2

Song-maker

This project uses the micro:bit accelerometer to play different tones when the guitar is held and tilted while playing. Pressing the A button will save the current tone to an array. After ten tones, a repeating melody will be performed. Press the B button to clear the array and start over.



```
forever
    if length of array list < 10 then
        set currentNote to acceleration (mg) x + 1300
        ring tone (Hz) currentNote
    else
        show leds
            . . # . .
            . # . # .
            # . . . #
            . # . # .
            . . # . .
        for element value of list
        do
            play tone value for 1 beat

on button A pressed
    if length of array list < 10 then
        list add value currentNote to end
        show number 10 - length of array list

on button B pressed
    set list to empty array
    clear screen
```

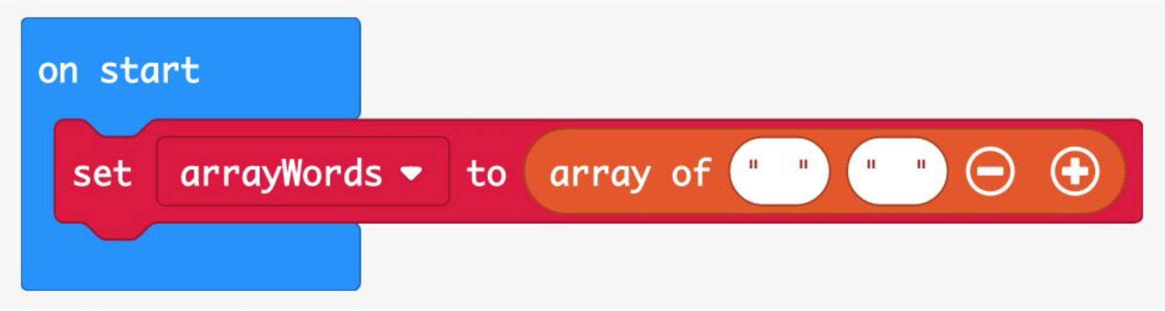
Maker Extension: Ideally, the micro:bit should be mounted in some kind of housing, perhaps a guitar shape or a music box. Start by looking at different kinds of musical instruments to get a sense of what kind of shape you might want to build around your micro:bit.

Activity 3

Headband Charades

Create a program that allows you and a partner to play a game of charades using the Micro:Bit at the game piece!

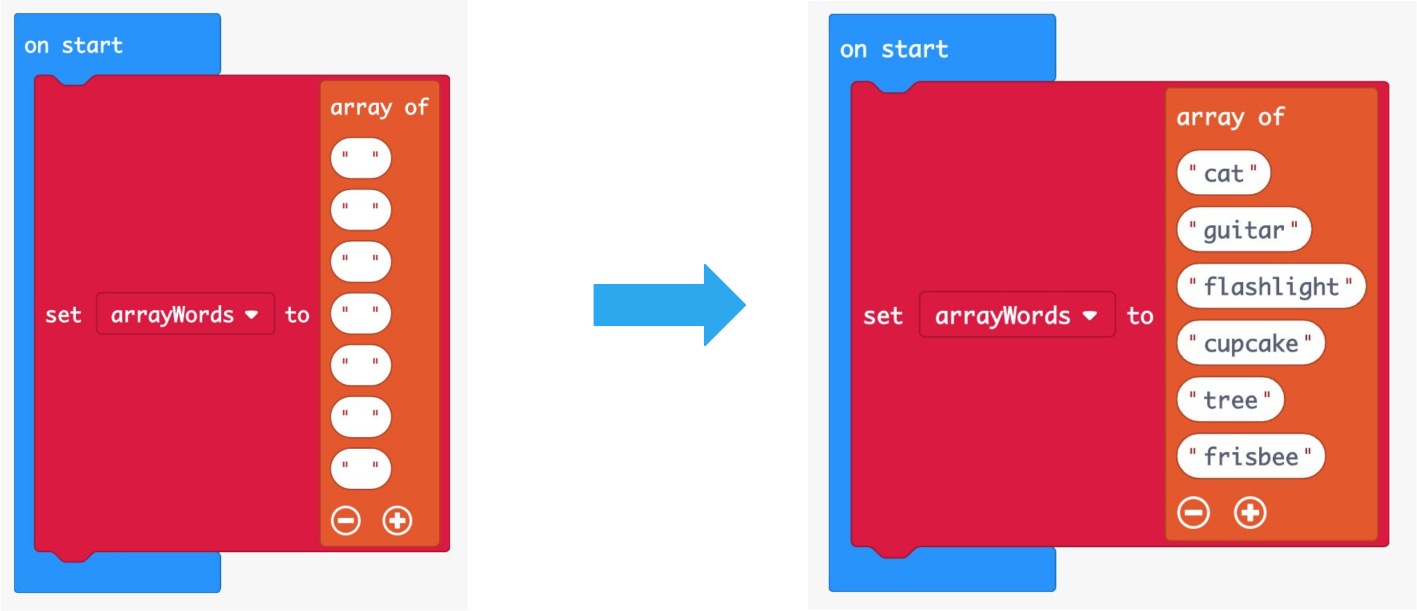
- Create a new variable and give it a name, such as “arrayWords”.
- Insert a ‘set variable’ block into the ‘on start’ block.
- Change the default variable name to this new variable name.
- From the Array Toolbox drawer, drag an ‘array of’ block to the coding workspace.
- Drag the orange ‘array of’ block into the ‘set variable’ block



```
on start
    set arrayWords ▾ to array of " " " " ⊖ ⊕
```

The array starts with 2 string elements in the array. Click on the (+) symbol at the end of the ‘array of’ block to add more elements to your array. For now, we’ll add 4 more values for a total of 6 values. Fill each string with one word. Choose words that will be fun for a game of charades.

Micro:Bit Arrays



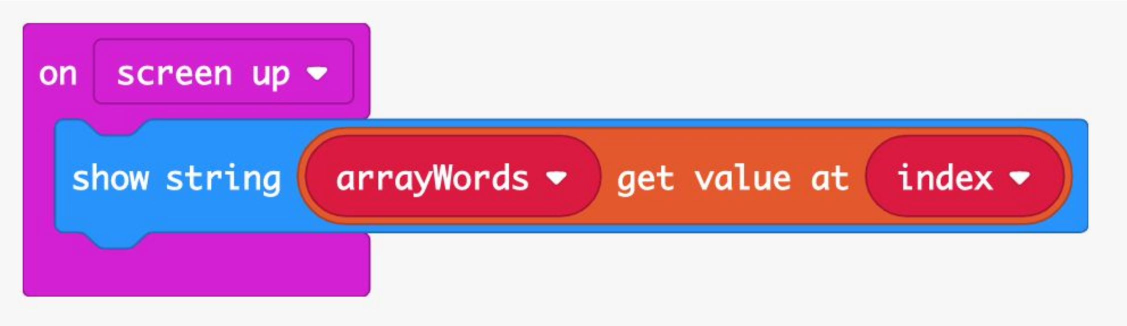
```
on start
  set arrayWords ▾ to array of
    " "
    " "
    " "
    " "
    " "
    " "
    " "
    ⊖ ⊕
```

```
on start
  set arrayWords ▾ to array of
    " cat "
    " guitar "
    " flashlight "
    " cupcake "
    " tree "
    " frisbee "
    ⊖ ⊕
```

Now we have set our array full of words. Next we need a way to display one word at a time from this array of words.

To program our Micro:bit to display a word when we tilt the micro:bit up, We can use the 'show string' block from the Basic Toolbox drawer, and the 'on screen up' event handler from the Input Toolbox drawer (this is a drop-down menu choice of the 'on shake' block) to tell the micro:bit.

For this version, we'll display the words one at a time in the order they were first placed into the array. We'll use the index of the array to keep track of what word to display at any given time, so you'll need to create a new variable which you can title "index".



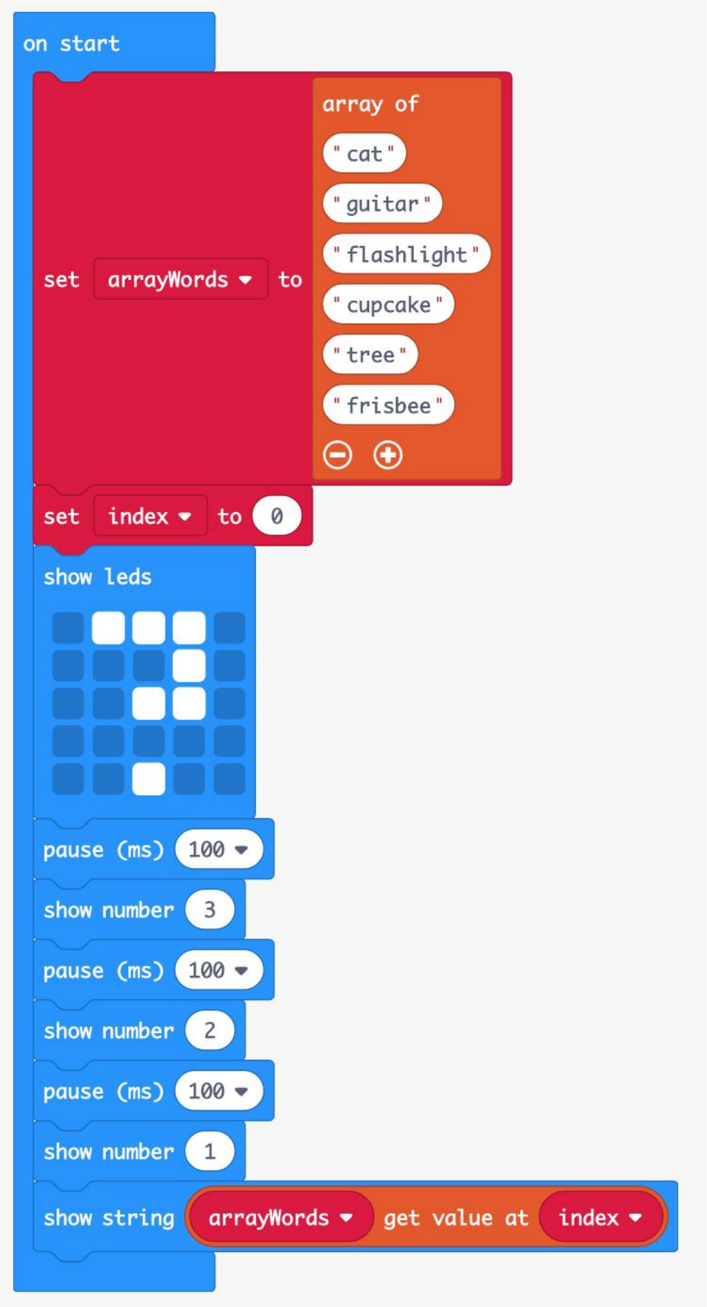
```
on screen up ▾
  show string  arrayWords ▾  get value at  index ▾
```

Micro:Bit Arrays

To start the game with the index at zero, add a ‘set variable’ block to the ‘on start’ block.

Next, add the following:

- Change the Micro:bit display when the program has started. Since charades is a guessing game, we displayed the question mark (?),
- Countdown to the first word using ‘show number’ blocks and ‘pause’ blocks
- And finally, display the first word in the array

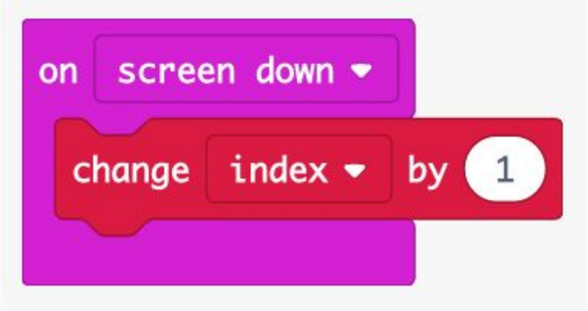


```
on start
    set arrayWords ▾ to array of "cat" "guitar" "flashlight" "cupcake" "tree" "frisbee" ⊖ ⊕
    set index ▾ to 0
    show leds
        . # # # .
        . . . # .
        . . # # .
        . . . . .
        . . # . .
    pause (ms) 100 ▾
    show number 3
    pause (ms) 100 ▾
    show number 2
    pause (ms) 100 ▾
    show number 1
    show string arrayWords ▾ get value at index ▾
```

Micro:Bit Arrays

So far we have a start to our game and a way to display the first word. Once that word has been guessed (or passed), we need a way to advance to the next word in the array.

We can do this by changing the index of the array with the ‘on screen down’ event handler from the Input Toolbox drawer (this is a drop-down menu choice of the ‘on shake’ block) to advance to the next word when we tilt the micro:bit down.



```
on screen down ▾
    change index ▾ by 1
```

We have a limited number of elements in our array, so to avoid an error, we need to check and make sure we are not already at the end of the array before we change the index.

Under the Arrays Toolbox drawer, drag out a ‘length of’ block. The ‘length of’ block returns the number of elements in an array. For our array, the length of block will return the value 6.

But because computer programmers start counting at zero, the index of the final (6th) element is 5.

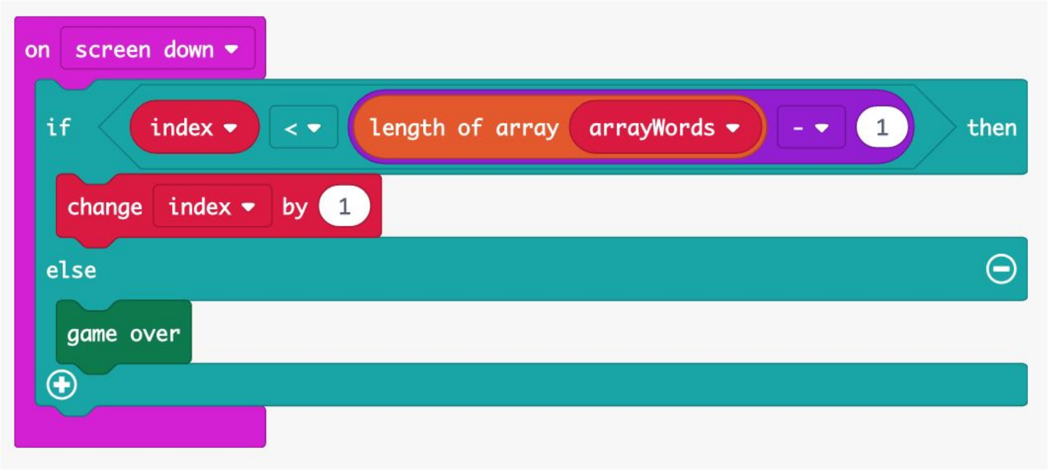
Some pseudocode for our algorithm logic:

- When the player places the micro:bit screen down, check the current value of the index.
- If: the current value of the index is less than the length of the array minus one
- Then: change the value of the index by one,
- Else: indicate that it is the end of the game.

Our array has a length 6, so this will mean that as long as the current value of the index is less than 5, we will change the array by one.

Using 'less than the length of the array minus one' instead of the actual numbers for our array makes this code more flexible and easier to maintain. We can easily add more elements to our array and not have to worry about changing numbers elsewhere in the code.

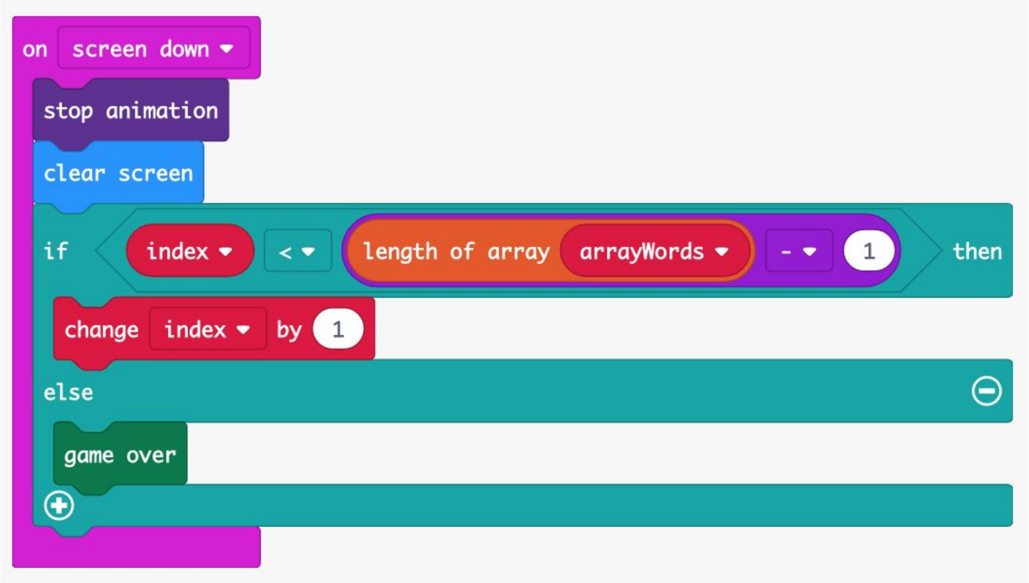
We can put this all together with an 'if...then...else' block and a 'less than' comparison block from the Logic Toolbox drawer, a subtraction block from the Math Toolbox drawer, and a 'game over' block from the Game Toolbox drawer (located under the Advanced menu).



```
on screen down
  if index < length of array arrayWords - 1 then
    change index by 1
  else
    game over
```

To make our game more polished, we'll add 2 more blocks for smoother game play.

In case a word is already scrolling on the screen when a player places the micro:bit screen down, we can stop this animation and clear the screen for the next word by using a 'stop animation' block from the Led More Toolbox drawer, and a 'clear screen' block from the Basic More Toolbox drawer.



```
on screen down
    stop animation
    clear screen
    if index < length of array arrayWords - 1 then
        change index by 1
    else
        game over
```

Game Play

There are different ways you can play charades with our program. Here is one way you can play with a group of friends.

With the micro:bit on and held so Player A cannot see the screen, another player starts the program to see the first word.

The other players act out this word charades-style for Player A to guess.

When Player A guesses correctly or decides to pass on this word, a player places the micro:bit screen down.

When ready for the next word, a player turns the micro:bit screen up. Play continues until all the words in the array have been used.