Have a go activities

Welcome from the Arm School Program! This document contains tasks that relate to our video playlist here: https://www.youtube.com/playlist?list=PLXwAdcOl0lCpKRep_Gb_YVDfYjcaqZSvA

Please note that not all modules include tasks.

Module 3 - Images

Have a Go

Program your micro:bit to store and display:

- Your initials one letter at a time
- Your year of birth one digit at a time

Change the program so there is a delay between each image.

Program your micro:bit to count from 0 to 9. Store the code for each of the numbers first.

Program the micro:bit to turn on 1 LED in a corner, then gradually increase until all the LEDs are lit, and then turn off each LED 1 at a time until they are all off. Use a sensible delay between each image.

Module 5 - Animation

Have a Go

Program your micro:bit to:

- Make the LED move along the first row, then the second, then the third, then the fourth, then the fifth before disappearing
- Change this program so that after it disappears, it comes back on and moves back up each row of LEDs
- Make the initial of your first name move from the right-hand side of the LEDs to the left and then disappear
- Repeat with two other letters of the alphabet (choose ones you can make a word from – you might want to do more than two). You should now have three different scrolling letters. Make them scroll one after the other to display a word
- Create a mini-movie where one fully bright LED is a pixel trying to escape, and there are a series of half-bright LEDs chasing it around the screen trying to catch it
- Create a firework show! Set each firework in its own list and then display different fireworks in different orders to create a show

Module 6 - Buttons and Event Handling

Have a Go

Program your micro:bit to:

- Turn all the LEDs on when button **a** is pressed
- Display three different images one after the other when button **b** is pressed
- Display the number 0 when button **a** is pressed and the number 1 when button **b** is pressed
- Display a happy face with button **a** is pressed and a sad face when button **b** is pressed
- Display 1 if button **a** is pressed, 2 if button **b** is pressed, and 0 otherwise
- Turn one vertical column of LEDs on if button **a** is pressed, one horizontal row on if button **b** is pressed, and all the LEDs otherwise
- Turn all LEDs off if button **a** is pressed, and turn them all on otherwise

Module 8 - Lists

Have a Go

Program your micro:bit to:

- Store the numbers 1 to 10 in a list. Output each item in the list in ascending order
- Change the program to output the list in descending order
- Store the names of you and your friends in a list. Output the names of some of the people in your list
- Store the numbers 0 to 9 in a list. Output the numbers. Add code to change these numbers to be 1 to 10
- Output the new numbers
- Store five items of furniture in a list. Append three new items of furniture in the list. Output all the elements in the list
- Store two pre-defined MicroPython images in a list. Create two new images and store them in variables. Append these images to your list. Display all the images in the list

Module 9 - Mathematical Operators

Have a Go

Program your micro:bit to:

- Store the number 66 and 9 in separate variables. Work out the result of 66 MOD 9 and 66 DIV 9 and output the result for both
- Store the number 173 and 55 in separate variables
- Work out the result of 173 MOD 55 and store it in a new variable, repeat with DIV
- Work out the result of 55 MOD 173 and store it in a new variable, repeat with DIV
- Output all four values after you have calculated them all

Module 10 - Constants

Have a Go

Program your micro:bit to:

- Store two numbers in constants
- Multiply these two constants together and output the result
- Store a third and fourth number in constants
- Add these two constants together and output the result

Module 12 - Making Music

Have a Go

Program your micro:bit to:

- Play the WEDDING tune
- Play the PYTHON tune
- Play the BIRTHDAY tune and display the image of a candle or a cake (you will have to create this image)
- Play the CHASE tune and make the LEDs chase around the screen i.e. move rapidly
- Display an image then play a tune. Change the image and play another tune. Repeat with different images and tunes
- Store and then play the following tunes. You will have to search for them if you don't know them! You can extend the tunes further than the notes given if you want to:
    - Doe, a deer, a female deer (C D E C E C E)
    - Twinkle twinkle little star, how I wonder what you are (C C G G A A G F F E E D D C)
    - Jingle bells, jingle bells, jingle all the way (E E E E E E E G C D E)
- Find the notes to your favourite song and program the micro:bit to play it
- Revisit the tunes you wrote last time, and this time put in the pauses where you would naturally pause in the music
- Find a version of a famous piece of music using single notes e.g. The 1812 overture and create a program to store and play this music. Make sure you add rests at the appropriate places
- Create yourself a theme tune. You might need an instrument e.g. a keyboard, to play around with first. Develop your tune, write down the notes and then write a program for your micro:bit to store and play your theme tune

Module 13 - Speech

Have a Go

Program your micro:bit to:

- Say your name
- Say one message when button **a** is pressed and another when button **b** is pressed
- Recite a poem

You have won an Oscar! Congratulations. You decide to send your micro:bit to the ceremony because you're too amazing to attend yourself.

Write an Oscar speech and program your micro:bit to read it out. Vary the speed, pitch etc. of your voice to make the speech interesting and varied. Don't forget to thank your friends for their support.

Program your micro:bit to phonetically say the words:

- House
- Love
- Lost
- Road
- Begin
- Night time
- Tomorrow

Module 14 - Selection

Have a Go

Program your micro:bit to:

- Output "yes" if 23 is greater than 10
- Store your name in a variable. Check if this variable is equal to "Sarah". Output "Sarah" if it is not equal to it
- Store two numbers in different variables. Compare the two variables and output the largest
- Store two different words in two variables. Compare the variable contents. If they are not the same, output a tune, output a different tune if they are the same
- Store a number in a variable. If the content of the variable is 10, display an image that lights up 10 LEDs on the micro:bit. If it isn't, then display all the LEDS on the micro:bit
- Display an image if button **a** is pressed. Display a different image if button **b** is pressed. Turn on all the LEDs if neither is pressed. (You will also need a while True loop to run this)
- Store a number in a variable. If the number is 1, output a tune, if it is 2 output a different tune; repeat up to and including the number 9. If the number is not any of these, play one different tune
- Store four numbers
- Find and add together the two largest numbers, and the two smallest numbers. Output these two totals

Module 15 - Pins

Have a Go

Program your micro:bit to:

- Output a different image if pin 0, 1 or 2 are pressed
- Output an animation if pin 0 is pressed
- Output the number of each pin you have pressed
- Output to a buzzer when you click on button **a**
- Output a different tune (by using different sleep values) when you click on button **a** and button **b**
- Investigate another output device, connect it to your micro:bit, find the code to output (use the Internet) and see what you can make it do

Module 16 - Gestures

Have a Go

Program your micro: bit to:

- Display an animation if it is being shaken, for example random LEDs come on and off
- Display a single LED in the centre and change the LED that is lit depending on whether the micro:bit it tilted up, down, left or right. For example, if you tilt it to the left then an LED to the left of the centre lights up
- Light two LEDs and when the user tilts the micro:bit one of the LEDs moves in that direction by one space. The user has to get their LED to touch the other LED
- Light one LED and when the user tilts the micro:bit more LEDs in that direction light up. The user has to get every LED lit, and when they do a happy face appears
- Light up all the LEDs and each time the user shakes the micro:bit one LED is removed
- Check the gestures performed since the last call, and then perform an action depending on each one. For example, play a tune if it was shaken, display an image if it was tilted up etc.
- Use a random number generator to display a random LED every few seconds. The user has to tilt the micro:bit to move an LED (or series of LEDs) to touch the LED before it disappears. This gives them a point. This continues until the user shakes the micro:bit, and then their score is displayed
- Challenge: change the previous program to simulate the traditional snake game. Each time an LED is touched the user's snake increases in length by 1

Module 17 - Directions

Have a Go

- Write a program so that different images are displayed depending on which is the nearest compass point to the micro:bit is pointing. For example, if it is pointing more north than east then the north image is displayed, if more east than north or south the east image is displayed

Module 18 – Storage on micro:bit

Have a Go

Program your micro:bit to:

- Store your name in a file
- Create an image and store it in a file
- Create a list of images, or sounds, and store them in a file
- Store a series of ten images in a text file separated by a character e.g. " "
- Read in the images into a list and output a random image each time the user clicks button **a**
- When the user clicks button **b**, output all the images as an animation

Module 19 - Subroutines

Have a Go

Program your micro:bit to:

- Check if button **a** or **b** has been clicked, run one procedure if button a is clicked, and **a** different procedure if button **b** is clicked. These could output an animation or tune
- Change the program so if button **a** is clicked both procedures are run, but if button **b** is clicked the procedures are run in the opposite order
- Create two procedures. One performs an animation. One plays a tune. In the main program, loop from 1 to 20. Output alternately the animation, then the tune

Module 20 - Built in Functions (BIFs)

Have a Go

Program your micro:bit to:

- Call a function to check which button has been pressed
- Run a different procedure depending on which button has been pressed, and if neither button has been pressed
- These procedures could output an image, or song etc.
- Call a function to work out which gesture has been performed
- Write three procedures that perform different animations and run them in different orders depending on which gesture has been performed
- Send two numbers to a function. The function returns the largest of the two numbers. The main program then outputs this larger number
- Repeat the task above with three numbers being sent to the function
- Use a function to work out which button was pressed most since it was last checked and return this value. Send the letter of the button returned to a procedure to display a different image depending on whether it receives 'a' or 'b' as a parameter
- Send two numbers to a function. The function returns the largest of the two numbers. The main program then outputs this larger number. Use global variables instead of parameters
- Repeat with three numbers being sent to the function (using global variables again)
- Use a function to work out which button was pressed most since it was last checked and return this value. Use a procedure to display a different image depending on whether button **a** or button **b** was clicked the most times

Module 21 - Networks

Have a Go

Program your micro:bit to:

- Display an image when a message is received from one of the buttons
- Display the letter of the button that was pressed
- Display a different image depending on which button was pressed
- Display an animation depending on which button was pressed
- Output a different message depending on which button was pressed
- Display four different images depending on which button combination was pressed
- Display 6 different images depending on which button combination was pressed
- Display a letter of the alphabet depending on which button combination was pressed
- Micro:bits can send and receive, change the programs so that both micro:bits can send and receive

Module 22 - Radio

Have a Go

Program your micro:bits so that:

- One micro:bit sends an image and the second outputs the image. The second micro:bit then sends a different image back to the first, which then displays the image
- One micro:bit sends a message, the second then sends a confirmation to say it has been received. When the first micro:bit receives the confirmation, it stops sending the message
- The two micro:bits count alternatively, so the first outputs ‘1’, then the second outputs ‘2’, then the first outputs ‘3’ etc. up to ‘10’
- One micro:bit sends a different message depending on which button the user presses. The other micro:bit displays a different image depending on which button on the first micro:bit was pressed

Module 24 - Defensive Design

Have a Go

Program your micro:bit to:

- Ask the user to enter their username and password. The username must be one already stored in the system, and the password must have been at least eight characters when they created it
- When logged in, the program loads the user’s game, saved from the previous session (this could be any of the games you have made previously)
- The player then controls their character using the buttons. They must click on items to select them in the game

Module 27 - Classes and Modules

Have a Go

- Create a new instance of the class Image
- Find the different methods that can be performed on an image and use each one to find out what they do