# Debugging

Ages: 7-11
Length: 1 hour
Equipment: PDF print out

## Introduction

This lesson introduces the relevant words; ***debugging*** through activities that link to student's lives to help them relate to the concepts introduced. This lesson will build upon prior learning relating to building algorithms.

| | |
|---|---|
| **Curriculum Alignment** | • UK National Curriculum Computing Key Stage 2 |
| **Learning Objective** | • To describe how to debug a program<br>• To demonstrate the skill debugging within a program |
| **Keywords** | • Debugging |
| **Resources** | • Debugging cards worksheet and solution<br>• Anagrams worksheet and solution |
| **Lesson Sections** | • Setting the Scene<br>• Activity 1 – What is debugging and how is it used<br>• Activity 2 – MiRo simulator<br>• Summary |

## Setting the Scene

How do we fix something that is not working?

We look through each part and locate what is not working and fix it.

| Concept | How it is used |
|---|---|
| Debugging is the ability to look through a problem to locate what is not working as expected and fix it.<br><br>*Can you think of a math lesson where you did not get the correct answer and you had to work out what went wrong and how to fix it?* | When you write a program mistakes can occur. Mistakes can help you learn as you are investigating different sections of the code.<br><br>If the code is not working you need to look through all the lines or blocks of code to find the error and fix it. |

***Debugging*** is a skill that can be utilised in many aspects of life, you will be faced with different challenges and not always be successful the first time; being able to work out how to get it right is debugging.

First let us look at what debugging looks like.

## Activity 1

Activity 1 is focused on getting students to think about what ***debugging*** is and relate this to their everyday tasks.

| Scenario 1 | Scenario 2 |
|---|---|
| You have received some money for your birthday, and you are going shopping. You had £10 and you should have £2.50 left, but you have £1 left. Where has the money gone?<br><br>• How much have you spent?<br>• How much should you have left?<br>• Have you missed something you bought, or have you put money in a different pocket? | You are playing a new game on your console?<br><br>• How do you work out how to play the game?<br>• How do you know what the buttons do?<br>• How can you improve and find hidden elements of the game? |

Where else do you debug a problem?

## Whole Class Activity

Explain to the group that they have several words in front of them that are all jumbled up. The task is to 'debug' the words and sort the letters into the correct order to reveal the hidden word.

This activity shows the use of a debugging and how you use the information in front of you to find the solution.

**Differentiation**

- If pupils need extra support to understand what debugging is, place one set of cards in the right order to help visualise the end solution.

## Small Group Activity

Split the students into small groups and give them the debugging cards.

*N.B. worksheet works better if cut up to move the cards into place but can be left as a worksheet to look at and debug.*

In your group, I want you to look at the cards given you.

Without any information can you work out what order or position they should go in?

Encourage discussion in the groups.
**Ask them why they are in that order?**

After a minute give them the outcome expected; the answer for all is 10, using all cards.
**Ask them to debug and fix.**

You use ***debugging*** without thinking daily when you are faced with a new challenge or a problem that you have not got quite right on the first attempt.

**Differentiation**

- If pupils need extra support to understand what debugging is, set out one set of cards in the right order to help visualise the end solution.
- For higher ability pupils ask them to support other pupils as 'debugging' assistants by giving hints not full solutions.

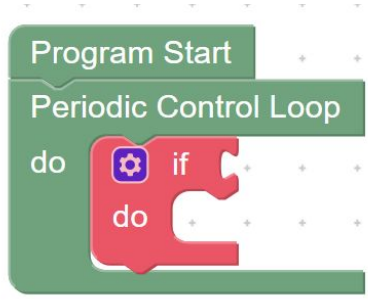
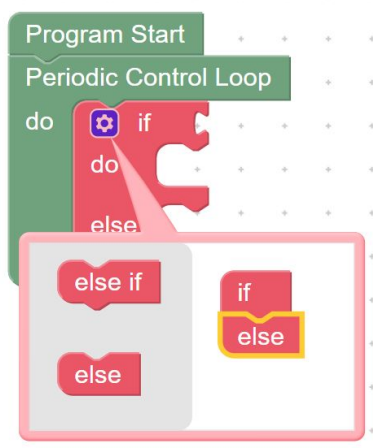
## Activity 2 Part 1

All the blocks could be used when debugging as it will depend on the error that has occurred and how it can be fixed.

It is good practice when creating a program, to develop and test in stages as this will help find an error for debugging.

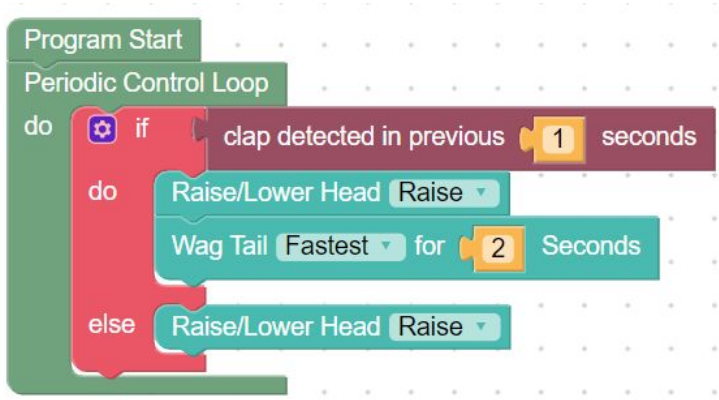
**This code is not working, can you help and debug it?**
We are going to recreate the code and debug and fix it.

Simulate MiRo to output a sequence of sounds when the input 'clap' is heard.

| | |
|---|---|
| **Step 1-3**<br>➔ Add 'program start' from 'Setup'<br>➔ Add 1 x Periodic Control Loop block<br>➔ Add 1 x if do block |  |
| **Step 4**<br>➔ Add else to the if do block<br><br>Click the blue settings icon<br><br>Click and drag an 'else' block from the left and connect under the 'if' on the right<br><br>Click the blue settings icon to close the pop up |  |

**Step 5-10**
- ➔ Add 1 x clap detected in previous 2 seconds block
- ➔ Change the number of seconds to 1
- ➔ Add 1 x Raise/Lower Head Raise block
- ➔ Add 1 x wag tail slow for 3 seconds block
- ➔ Set the tail to wag 'fastest' and for 2 seconds
- ➔ Add 1 x Raise/Lower Head Raise block

**The error is within the 'else' section and students are encouraged to complete the test table to identify the error themselves and how to fix it.




**Step 11**
- ➔ Click Simulator Play OR
- ➔ Click Robot play

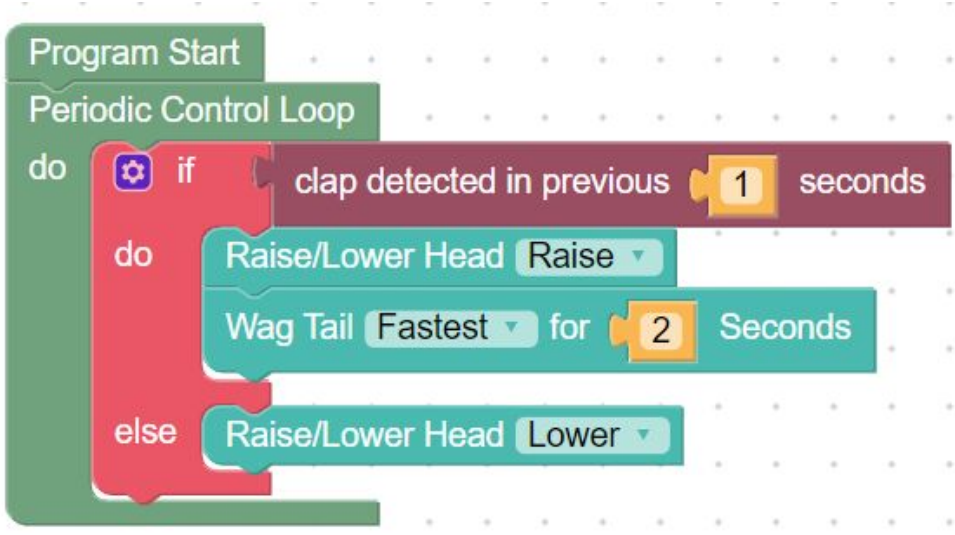
** If you are using the physical MiRO ensure the IP address is correctly added to the onscreen code to allow the code to be communicated to MiRO to action.

** If you are using the simulator it is recommended to zoom in to see miRo clearly but not essential

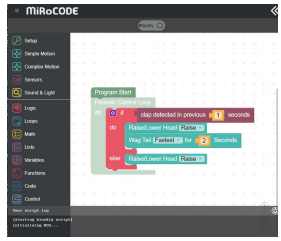
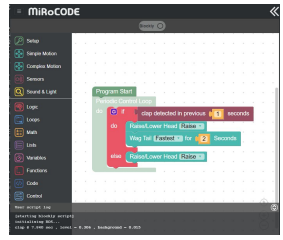
**Test table:**

| Test | Expected outcome | Actual outcome | Action |
|---|---|---|---|
| Within 1 second of clap. | Head raises and tail wags | | |
| After 1 second of clap | Head lowers and tail stops | **The error will be identified here as the head does not lower.** | |

**Solution**



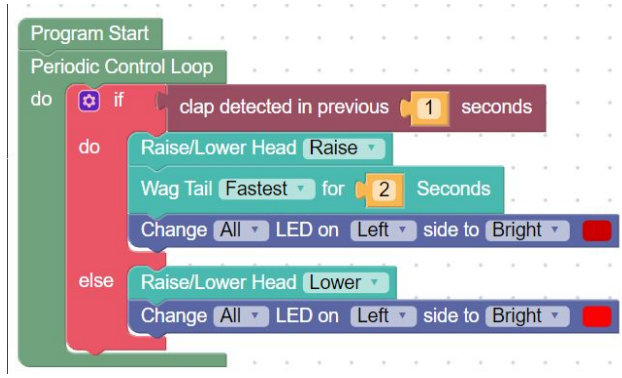
Following how the program executes will help the pupils identify when an error occurs.

| | | | |
|---|---|---|---|
| **Follow a program** |  **Script at bottom of screen shows code is ready and waiting for the 'clap'** |  **Simulate the clap by clicking this icon OR With physical MiRo - clap your hands** |  **Script at bottom shows program ran correctly.** |

## Activity 2 Part 2

Within this activity the use of colour is added to the program to simulate the mood when MiRo raises and lowers its head.

**This program builds on the previous activities program so pupils need to keep their program on their screen to build on.**

| | |
|---|---|
| **Step 1-2**<br>➔ Add 2 x change front led block<br>➔ Set from 'change front' to 'change all' on the drop-down menu | **   |
| **Step 3**<br>➔ Click Simulator Play OR<br>➔ Click Robot play | ** If you are using the physical MiRO ensure the IP address is correctly added to the onscreen code to allow the code to be communicated to MiRO to action.<br>** If you are using the simulator it is recommended to zoom in to see miRo clearly but not essential |

**Observe the output**

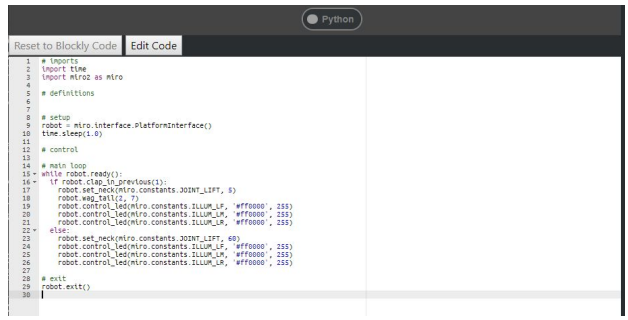
| Test | Expected outcome | Actual outcome | Action |
|---|---|---|---|
| Within 1 second of clap. | Head raises, tail wags and lights green | **The error is that the colour is red** | |
| After 1 second of clap | Head lowers, tail stops and lights red | | |

**Debugging**

From your test table students will find that the colour red is used in both blocks.

This debug is going to be completed in the python code but for some students using the block to set the colour is available.

Click the button above the block code with the word ‘Blockly’ on it and switch the screen to ‘python’.




Lick the ‘Edit Code’ button.

Identify where the colour code ‘**#ff0000**’ is used. It is used in 6 different locations.

***There are 3 individual lights on MiRo and all visible to change when the block is set to ‘all’ lights.***

The first three are the ones the student needs to edit to a new colour.

RGB stands for Red, Green, Blue and the numbers or letters identify how much of each colour is added to make up the new colour.

```
# imports
import time
import miro2 as miro

# definitions


# setup
robot = miro.interface.PlatformInterface()
time.sleep(1.0)

# control

# main loop
while robot.ready():
  if robot.clap_in_previous(1):
    robot.set_neck(miro.constants.JOINT_LIFT, 5)
    robot.wag_tail(2, 7)
    robot.control_led(miro.constants.ILLUM_LF, '#ff0000', 255)
    robot.control_led(miro.constants.ILLUM_LM, '#ff0000', 255)
    robot.control_led(miro.constants.ILLUM_LR, '#ff0000', 255)
  else:
    robot.set_neck(miro.constants.JOINT_LIFT, 60)
    robot.control_led(miro.constants.ILLUM_LF, '#ff0000', 255)
    robot.control_led(miro.constants.ILLUM_LM, '#ff0000', 255)
    robot.control_led(miro.constants.ILLUM_LR, '#ff0000', 255)

# exit
robot.exit()
```

Edit the colour code to:

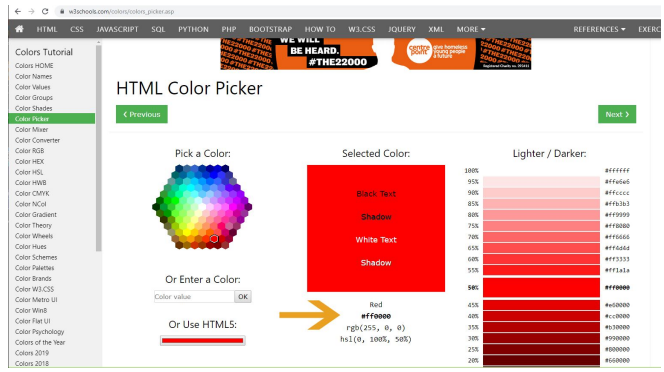
**#00ff00**

The first pair represent red, 2nd pair represent green and final pair represent blue. 0 is off and F is the highest amount of colour it can be.

The range of each is 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

Test the program - Click ‘Play’ from python view (if you click back to blockly the code added is not retained)

```
# imports
import time
import miro2 as miro

# definitions


# setup
robot = miro.interface.PlatformInterface()
time.sleep(1.0)

# control

# main loop
while robot.ready():
  if robot.clap_in_previous(1):
    robot.set_neck(miro.constants.JOINT_LIFT, 5)
    robot.wag_tail(2, 7)
    robot.control_led(miro.constants.ILLUM_LF, '#00ff00', 255)
    robot.control_led(miro.constants.ILLUM_LM, '#00ff00', 255)
    robot.control_led(miro.constants.ILLUM_LR, '#00ff00', 255)
  else:
    robot.set_neck(miro.constants.JOINT_LIFT, 60)
    robot.control_led(miro.constants.ILLUM_LF, '#ff0000', 255)
    robot.control_led(miro.constants.ILLUM_LM, '#ff0000', 255)
    robot.control_led(miro.constants.ILLUM_LR, '#ff0000', 255)

# exit
robot.exit()
```

| | |
|---|---|
| https://www.w3schools.com/colors/colors_picker.asp<br><br>Find another RGB colour code and try it out by editing the python code.<br><br>Try changing each code to a different RGB colour code. |  |

Remember there is no fail in computing only debugging, fixing and learning!

## Summary

Have a discussion with the class about what they have learnt in the lesson. Discuss the new word learnt: ***debugging*** and talk through any difficulties they had.

Ask students to complete the self-assessment and can be done by thumbs up, down and centre or using the images; on the following 3 questions

| Questions |
|---|
| Can you describe what debugging is? |
| Can you give an example of how debugging is used? |
| Can you debug a given program and fix it? |