Python 5: Iteration Part 1

Teaching Resource

Resources

- Slides **Python 5: Iteration 1**
- You will need to either have Python IDLE installed or have access to an online Python IDE. We have used https://editor.raspberrypi.org/en
- We have added a walk-through video below

Prior Knowledge

- Printing, mathematical operators, data types and input using Python
- Selection in Python using `if..elif..else`

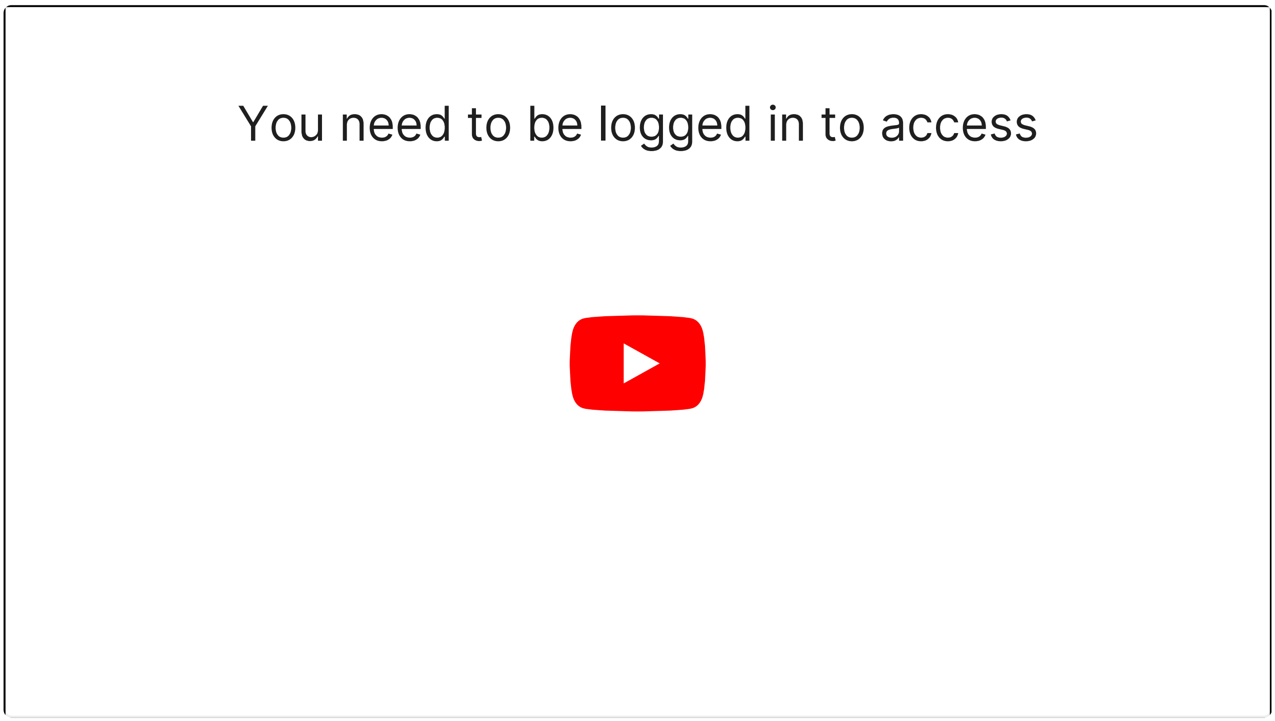
Python Den extends the block code of Rapid Router. If your pupils have not used Rapid Router before, it is recommended that they spend some time on this before starting this lesson. Some of levels 13 - 18 (no loops) and 19 - 28 (loops) will give them some useful background.

Vocabulary

- **Boolean expression** - an expression that evaluates to True or False, e.g. `age < 15` will give True if your age is under 15, but False if it is 15 or more.
- **increment** - we usually use the word to mean increase a value by 1, e.g. `count = count + 1`
- **indentation** - moving words or program code (usually) four spaces to the right; used for code blocks in Python.
- **iteration** (also called **looping**) - repeating a block of code a specific number of times.
- **loop counter** - a variable used to control a counted loop.

3MB Python 5 - Iteration 1.pptx

Lesson walk-through


You need to be logged in to access

Learning Objectives

To teach pupils to

- use `while` loops to repeat a block of code a specific number of times
- transition from loops in block code to text code in Python Den

Teacher Note

It is a deliberate decision to use `while` loops before `for` loops. Iterating is a fundamental operation in programming, and using a `while` loop for iteration can help pupils to understand the concept of manual index handling and loop control.

Once this is properly understood and pupils can use it with more complex problems, then `for` loops are introduced as a useful and efficient shortcut.

Slide Notes

Activity 1: Using Python Den

Slides 4–8: Start the lesson by showing pupils how the block code is translated into Python code for them in some of the Python Den levels. These slides highlight the different parts of a simple program.

Note: The first two lines of code will already be in any Python Den levels they do, so they do not need to remember them.

Slide 10: Ask pupils to study this code and point out anything they recognise or anything they observe. The purpose of this is to encourage them to read the code carefully and hopefully avoid some common errors later. Notes follow on **slide 11**.

Activity 2: Writing Python code in Python Den

Slide 12: Ask pupils to do these Python Den levels. They follow the same pattern as the example in the slides: the Python code is generated from the block code. Encourage pupils to read the Python code and compare it to the block code.

When you are ready, show **slide 13** which reminds pupils of the Python code they need. Levels 113–115 require them to type in the Python code.

Slide 14 highlights some common errors.

Activity 3: Iteration in Python Den

Slide 16: On this slide, you are looking for pupils to identify the repeated code.

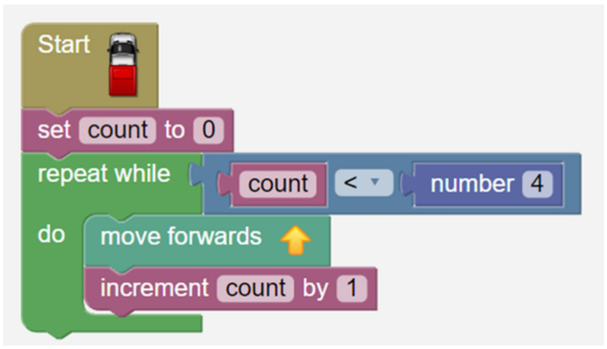
Before you show slide 17, you might like to do an unplugged exercise in counting. You could ask a pupil to come to the front of the classroom and do something ten times, e.g. touch their toes. Ask the class how the pupil will know when they have done this ten times. The answer is that they count. Explore this idea. What is counting? We start at zero and every time we do the thing we are counting we add one to count. Then we check to see if we have done enough by comparing our count to ten.

Slide 17: On this slide, the block code uses a `while` loop to repeat a single `move forwards` block four times. The algorithm is expressed in natural language on the left. Equate these to each other.

Slide 18: This slide removes the algorithm and just shows the block code and the route that it solves.

Slide 19: Pupils should now try levels 116–118 which follow the same pattern as the previous example. You might prefer to hide the example to allow pupils to try to work it out, or show them **slide 18** to help them.

Important note: please pay particular attention to the blocks used for **variables** (e.g. `count` below) and **number values** (e.g. 4 below). They are different colours and not interchangeable.



```
Start
set count to 0
repeat while count < number 4
do  move forwards
    increment count by 1
```

Once pupils have finished these levels, show **slide 20**. This shows the Python code alongside the block code. Ask pupils to comment on what they observe. These things are important to notice:

- `count = 0` comes before the while
-

- There is a colon after `while count < 4`
- The next two lines are shifted right a bit. This is called **indentation**.

These points are highlighted on slide 21.

Slide 21: Ensure that pupils understand the terminology used on this slide, and then ask them to do levels 119–122. In these levels, they will be asked to create loops using Python on its own. You might like to leave the slide on the board to help them with syntax.

Hint: if pupils have missed the indentation, the colon etc. or used the wrong brackets, rather than telling them what their error is, you might like to ask them to compare their code to the example.

Plenary

Ask pupils to define and describe the terms given to see what they remember.

Then ask them why loops are useful. The answer is that they allow us to repeat code without just copying and pasting it repeatedly.