CONTROL FLOW: FOR STATEMENTS

If you have done the exercise with the traffic lights, you will have found that you need to write a lot of code to set half the LEDs to one colour and the other half to another – we have to set them individually, one at a time. Fortunately, there's a shortcut called the **`for`** statement which is used to execute some code a fixed number of times.

If we want to make the first 8 LEDs flash RED the code would be:

```
void loop() {

      int i;

      for (i = 1; i <= 8; i++)
            EngduinoLEDs.setLED(i, RED);

      delay(1000);
      EngduinoLEDs.setAll(OFF);
      delay(1000);

}
```

There are several parts to this. The first is the statement:

```
      int i;
```

This declares a variable **`i`**. You can think of a variable as a box to put things into with a special name. Our box is called **`i`**, but you can call it what you like so long as it's unique and you're consistent.

The loop is then:

```
      for (i = 1; i <= 8; i++)
            EngduinoLEDs.setLED(i, RED);
```

What this means is:

a) Set the value of **`i`** (the number in the box called **`i`**) to be 1
b) check to see if the value of **`i`** is less than or equal to 8. If it is not, then go to step f). If **`i`** is less than or equal to 8, then
c) set LED with the number we have in the box labelled **`i`** to be RED
d) add one to **`i`**
e) go back to step b) and carry on.
f) Carry on with the rest of the program - execute whatever is next in the code after this **`for`** statement. In this case that's **`delay(1000)`**

Just as for if statements, it is possible to group more than one statement together:

```
      for (i = 1; i <= 8; i++) {
            EngduinoLEDs.setLED(i, RED);
            Serial.print(i);
            Serial.println(" switched on");
      }
```