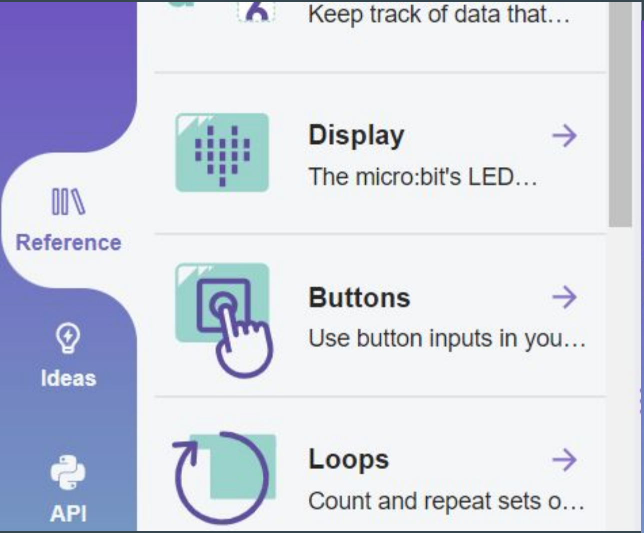
micro:bit LEDs

In the python editor, the led References are under Display

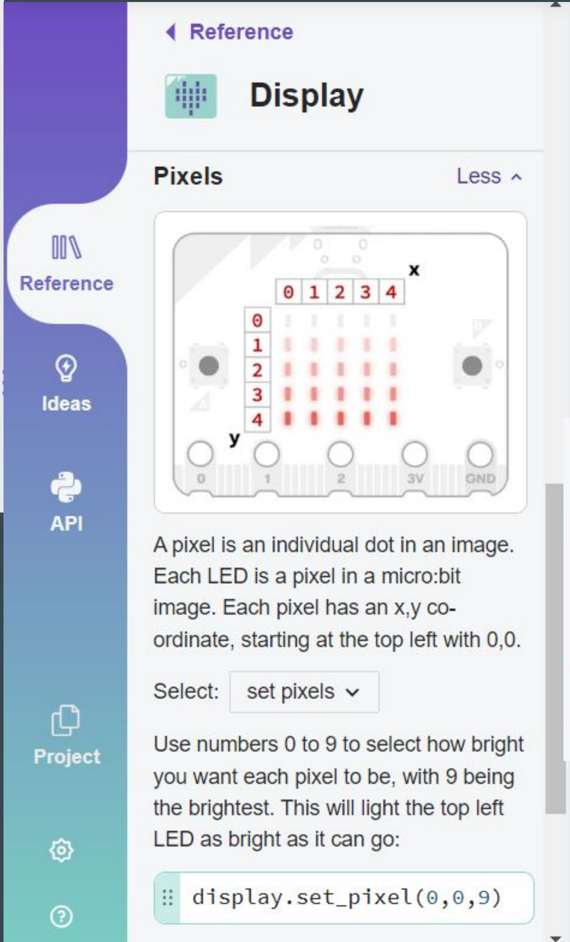


Keep track of data that…

Display →
The micro:bit's LED…

Buttons →
Use button inputs in you…

Loops →
Count and repeat sets o…



◂ Reference

Display

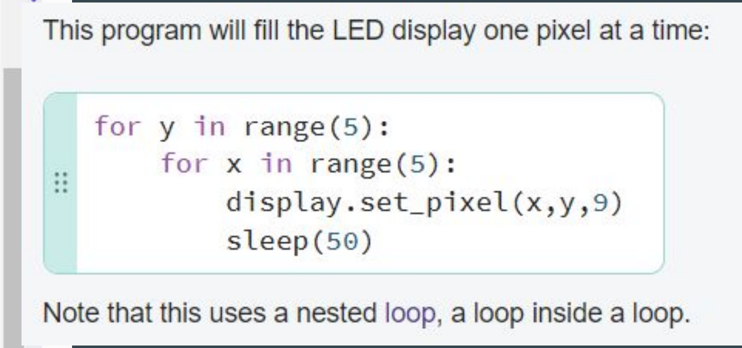
Pixels Less ⌃

A pixel is an individual dot in an image. Each LED is a pixel in a micro:bit image. Each pixel has an x,y co-ordinate, starting at the top left with 0,0.

Select: set pixels ⌄

Use numbers 0 to 9 to select how bright you want each pixel to be, with 9 being the brightest. This will light the top left LED as bright as it can go:

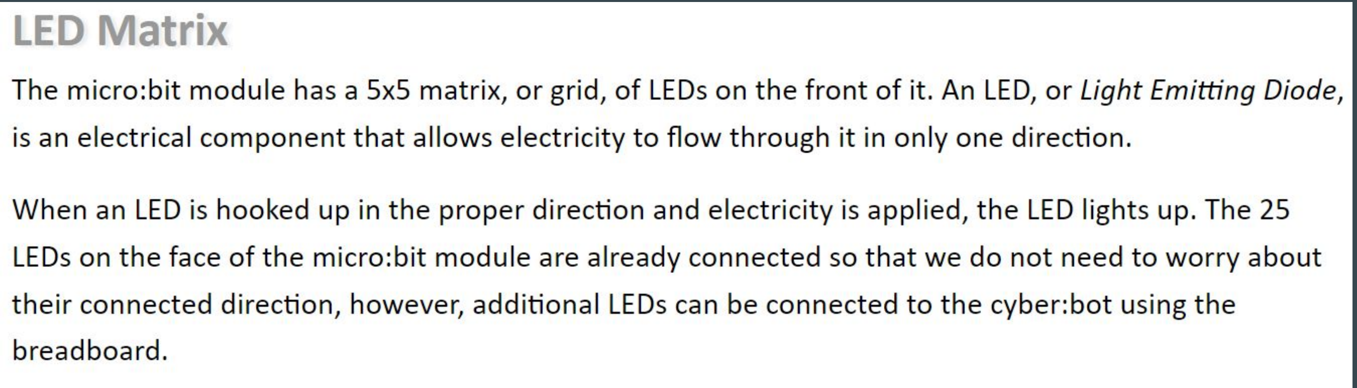
```
display.set_pixel(0,0,9)
```



This program will fill the LED display one pixel at a time:

```
for y in range(5):
    for x in range(5):
        display.set_pixel(x,y,9)
        sleep(50)
```

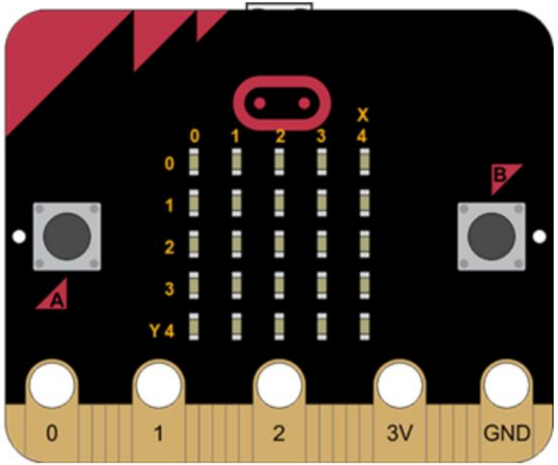
Note that this uses a nested loop, a loop inside a loop.

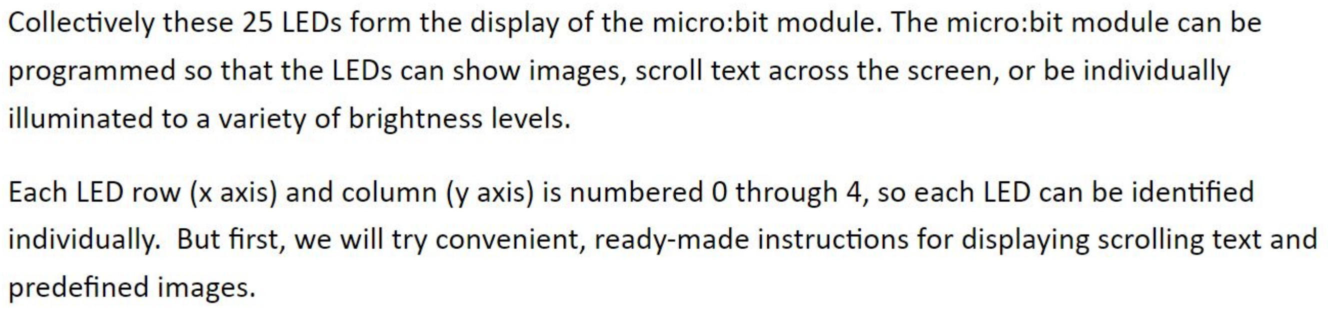


LED Matrix

The micro:bit module has a 5x5 matrix, or grid, of LEDs on the front of it. An LED, or *Light Emitting Diode*, is an electrical component that allows electricity to flow through it in only one direction.

When an LED is hooked up in the proper direction and electricity is applied, the LED lights up. The 25 LEDs on the face of the micro:bit module are already connected so that we do not need to worry about their connected direction, however, additional LEDs can be connected to the cyber:bot using the breadboard.





Collectively these 25 LEDs form the display of the micro:bit module. The micro:bit module can be programmed so that the LEDs can show images, scroll text across the screen, or be individually illuminated to a variety of brightness levels.

Each LED row (x axis) and column (y axis) is numbered 0 through 4, so each LED can be identified individually. But first, we will try convenient, ready-made instructions for displaying scrolling text and predefined images.

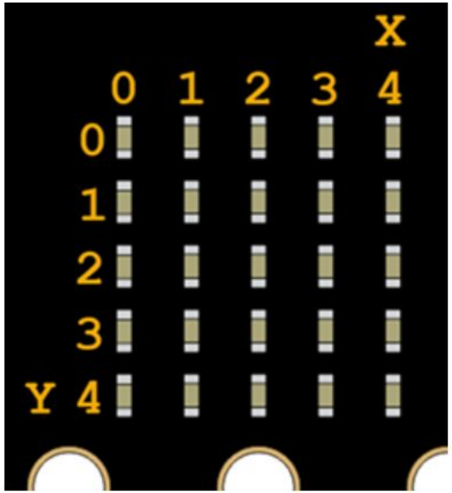
Custom Images

You can create custom images from individual pixels on the micro:bit module's display by using **`display.set_pixel`**.

The **`set_pixel`** method has three parameters: ***x***, ***y***, and ***b***:

```
    display.set_pixel(x, y, b)
```

The arguments for ***x*** and ***y*** indicate the pixel's position in the micro:bit module display's coordinate plane. The ***b*** argument controls the brightness (0-9) of the pixel.



Example script: two_pixels

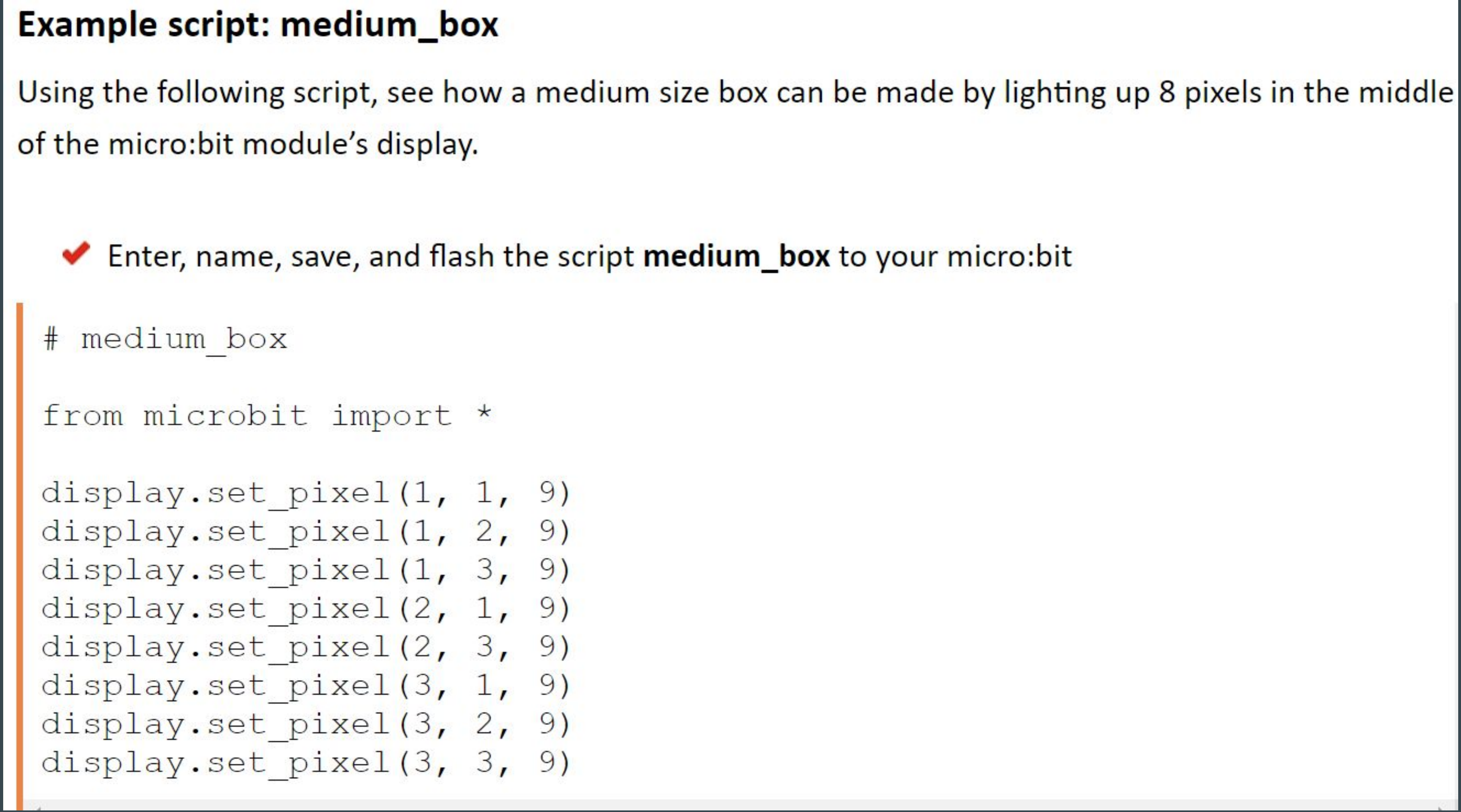
To see how this works, try the following program which lights up two LEDs.

✔ Enter, name, save, and flash the script **two_pixels** to your micro:bit.

```
# two_pixels

from microbit import *

display.set_pixel(0, 0, 9)
display.set_pixel(1, 3, 4)
```



Example script: medium_box

Using the following script, see how a medium size box can be made by lighting up 8 pixels in the middle of the micro:bit module's display.

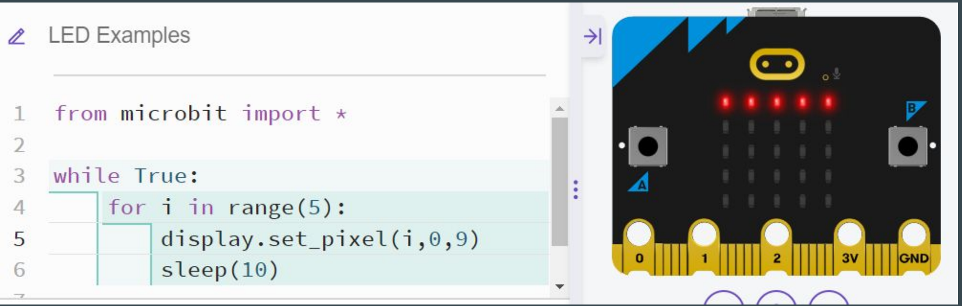
✓ Enter, name, save, and flash the script **medium_box** to your micro:bit

```
# medium_box

from microbit import *

display.set_pixel(1, 1, 9)
display.set_pixel(1, 2, 9)
display.set_pixel(1, 3, 9)
display.set_pixel(2, 1, 9)
display.set_pixel(2, 3, 9)
display.set_pixel(3, 1, 9)
display.set_pixel(3, 2, 9)
display.set_pixel(3, 3, 9)
```

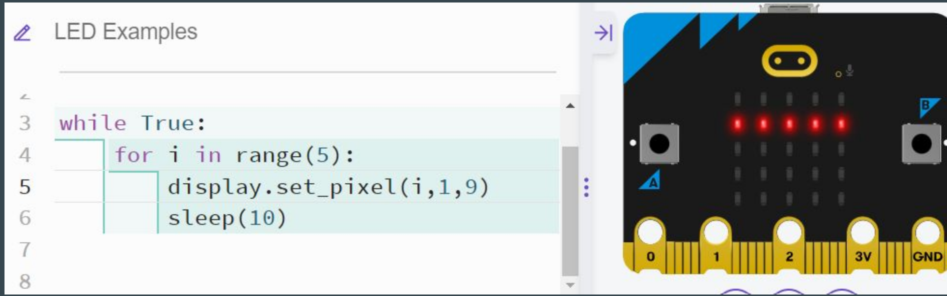
We can use for loops to turn leds on and off to create pattern.



LED Examples

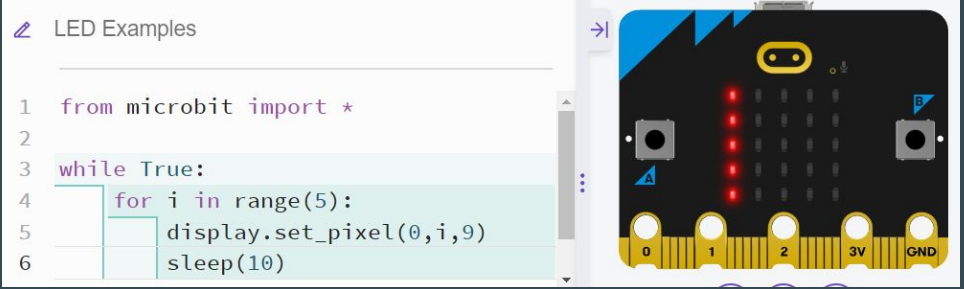
```
from microbit import *

while True:
    for i in range(5):
        display.set_pixel(i,0,9)
        sleep(10)
```



LED Examples

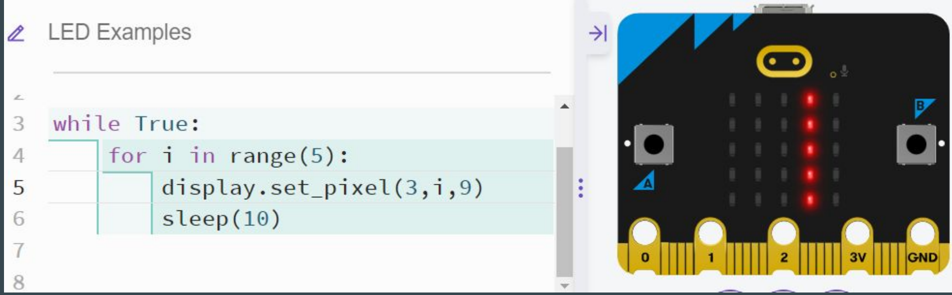
```
while True:
    for i in range(5):
        display.set_pixel(i,1,9)
        sleep(10)
```



LED Examples

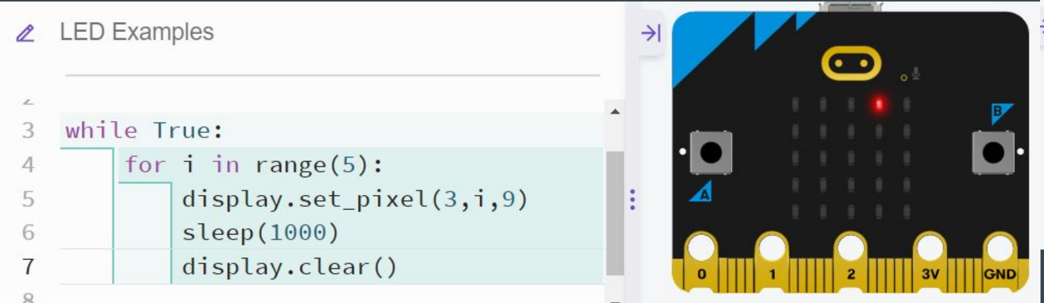
```
from microbit import *

while True:
    for i in range(5):
        display.set_pixel(0,i,9)
        sleep(10)
```



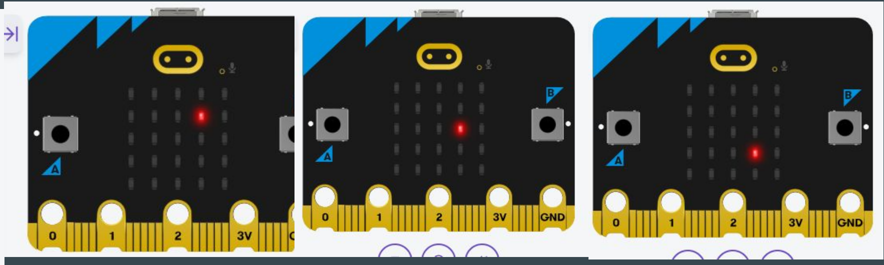
LED Examples

```
while True:
    for i in range(5):
        display.set_pixel(3,i,9)
        sleep(10)
```

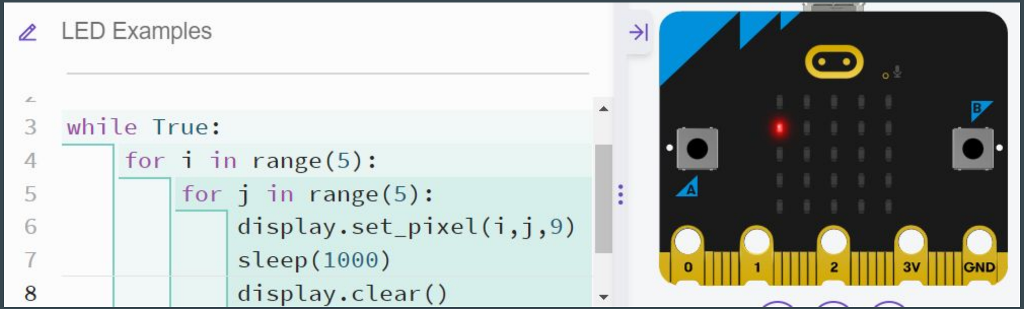


LED Examples

```
while True:
    for i in range(5):
        display.set_pixel(3,i,9)
        sleep(1000)
        display.clear()
```

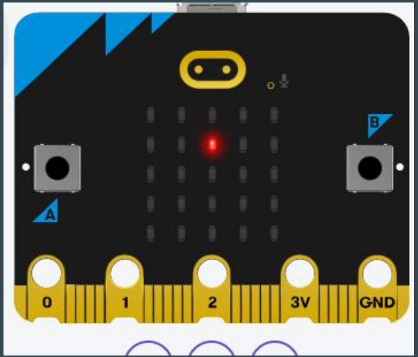


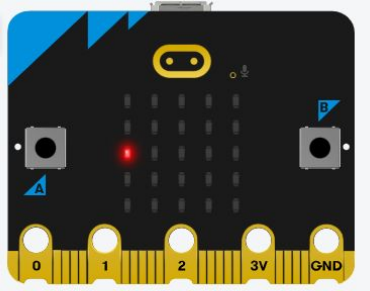
We can use nested loops to change the x and y throughout the loop.



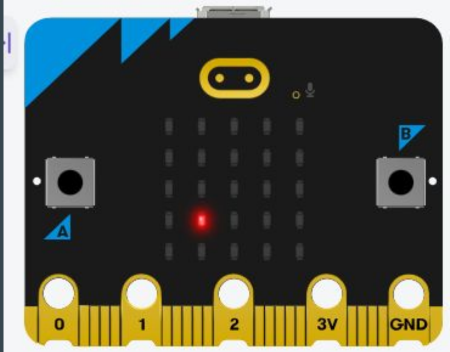
LED Examples

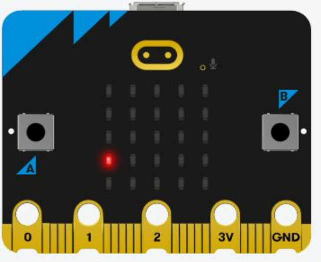
```
while True:
    for i in range(5):
        for j in range(5):
            display.set_pixel(i,j,9)
            sleep(1000)
            display.clear()
```



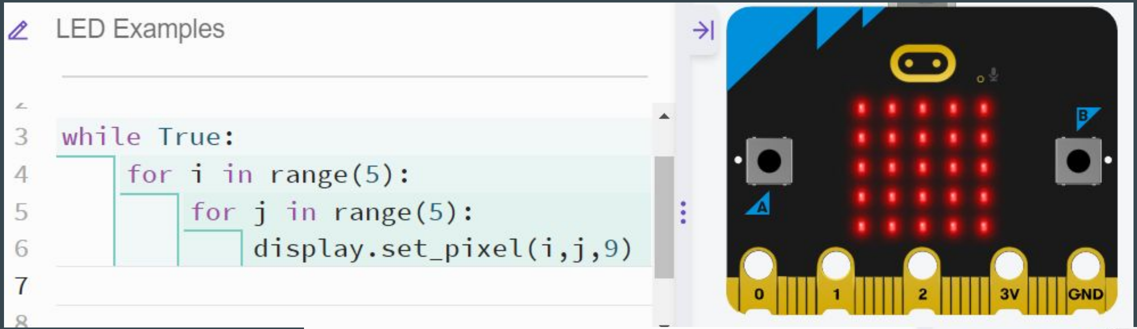


You will have to try this to see what it looks like.



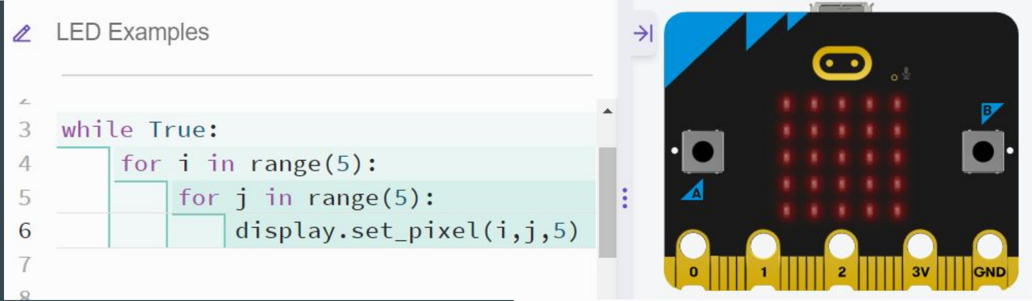


We can also change the brightness of the leds.



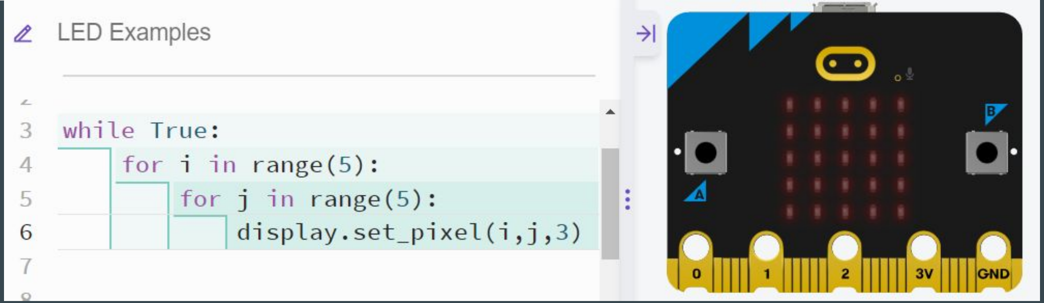
LED Examples

```
while True:
    for i in range(5):
        for j in range(5):
            display.set_pixel(i,j,9)
```



LED Examples

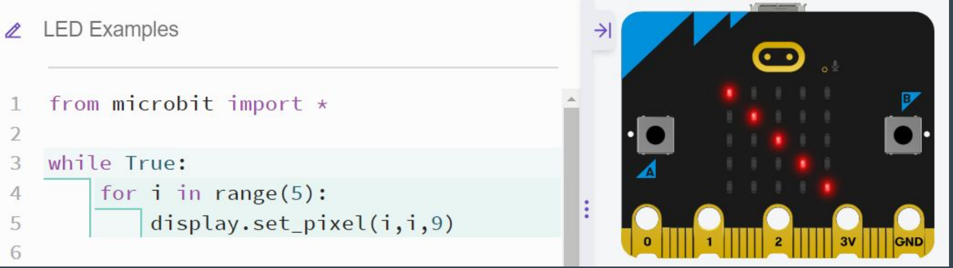
```
while True:
    for i in range(5):
        for j in range(5):
            display.set_pixel(i,j,5)
```



LED Examples

```
while True:
    for i in range(5):
        for j in range(5):
            display.set_pixel(i,j,3)
```

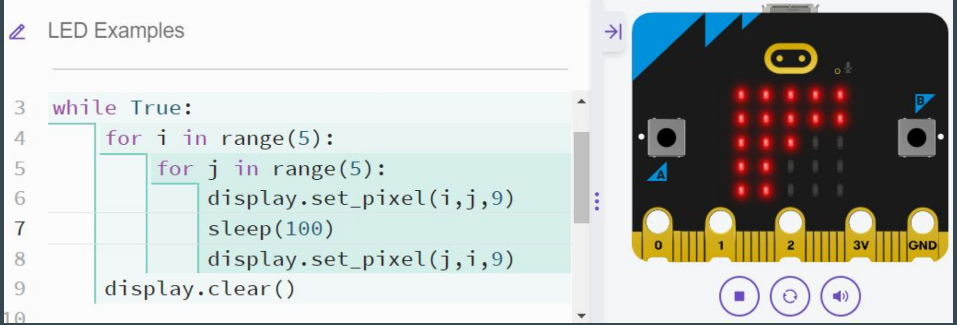
There are lots of possibilities with this.



LED Examples

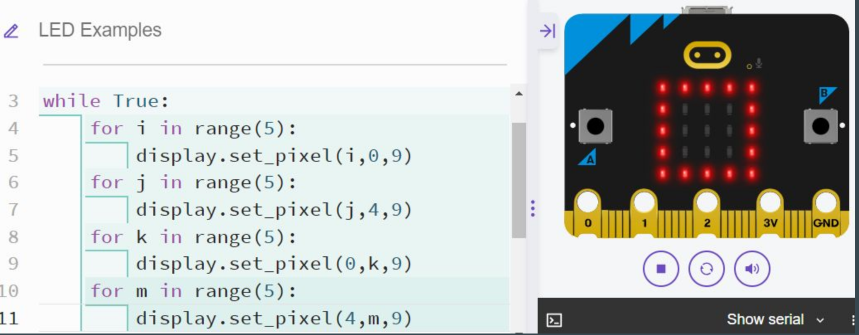
```
from microbit import *

while True:
    for i in range(5):
        display.set_pixel(i,i,9)
```



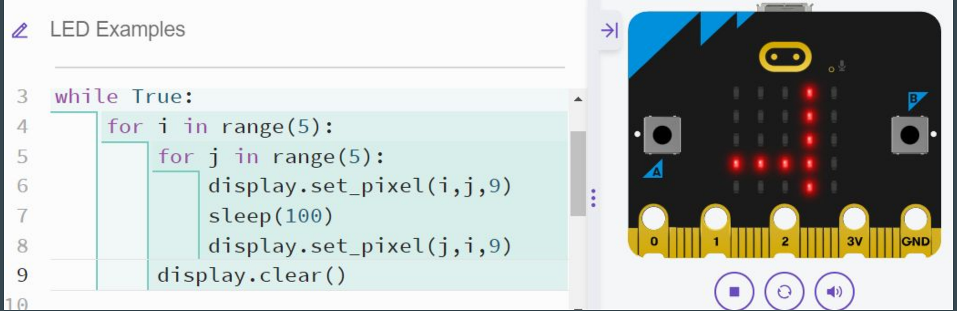
LED Examples

```
while True:
    for i in range(5):
        for j in range(5):
            display.set_pixel(i,j,9)
            sleep(100)
            display.set_pixel(j,i,9)
    display.clear()
```



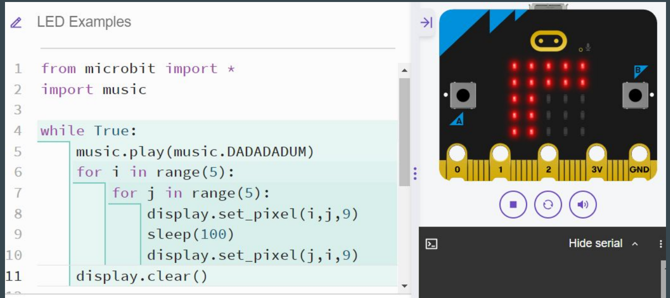
LED Examples

```
while True:
    for i in range(5):
        display.set_pixel(i,0,9)
    for j in range(5):
        display.set_pixel(j,4,9)
    for k in range(5):
        display.set_pixel(0,k,9)
    for m in range(5):
        display.set_pixel(4,m,9)
```



LED Examples

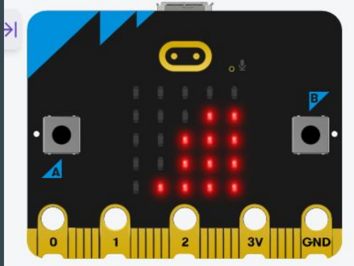
```
while True:
    for i in range(5):
        for j in range(5):
            display.set_pixel(i,j,9)
            sleep(100)
            display.set_pixel(j,i,9)
        display.clear()
```

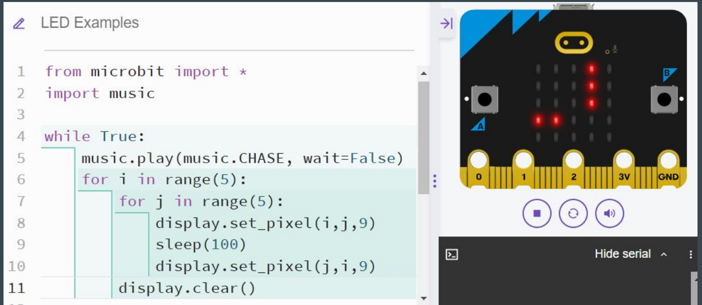


LED Examples

```
from microbit import *
import music

while True:
    music.play(music.DADADADUM)
    for i in range(5):
        for j in range(5):
            display.set_pixel(i,j,9)
            sleep(100)
            display.set_pixel(j,i,9)
    display.clear()
```





LED Examples

```
from microbit import *
import music

while True:
    music.play(music.CHASE, wait=False)
    for i in range(5):
        for j in range(5):
            display.set_pixel(i,j,9)
            sleep(100)
            display.set_pixel(j,i,9)
        display.clear()
```