BBC micro:bit
Shut The Matrix

Introduction

Here is some example code for a game which is loosely inspired on the Shut The Box classic dice game. Here, you roll a die and add it onto your total. Your aim is to reach 25 in as few rolls as possible. You have to land on 25 exactly, otherwise your total is lowered.

You have code enough to get you started here with more work needed to make the game play as you would like.

Programming

```
from microbit import *
import random

faces = [Image('00000:00000:00900:00000:00000:'),
        Image('00009:00000:00000:00000:90000:'),
        Image('00009:00000:00900:00000:90000:'),
        Image('90009:00000:00000:00000:90009:'),
        Image('90009:00000:00900:00000:90009:'),
        Image('90009:00000:90009:00000:90009:')]

def nleds(value):
    img = Image('00000:'*5)
    sp = img.set_pixel
    counter = 0
    for row in range(0,5):
        for col in range(0,5):
            if counter<value:
                sp(col,row,9)
            else:
                sp(col,row,0)
            counter += 1
    return img

def RandomImages(n, delay):
    for i in range(0,n):
            display.show(random.choice(faces))
            sleep(delay)
            display.clear()
            sleep(delay)

def PlayGame():
    counter = 0
    while counter!=25:
        if button_a.was_pressed():
            display.clear()
            sleep(250)
            roll = random.randint(1,6)
            RandomImages(10,75)
            display.show(faces[roll-1])
            sleep(500)
            if counter+roll==25:
                # won
                counter = counter + roll
            elif counter+roll<25:
                # add on
                counter = counter + roll
            else:
            # go to end and come back
                counter = 50 - (counter + roll)
        display.show(nleds(counter))
        sleep(10)
    for i in range(0,10):
        display.show(nleds(25))
        sleep(200)
        display.clear()
        sleep(200)

# Start The Game
PlayGame()
```

Challenges

1. Get all of the sleep statements delaying things for as much as you feel comfortable with. This will help you to focus in on different parts of the code for the game.
2. The example program doesn't keep a score. That is easy enough to do if you add another variable after counter and set it to 0 too. Add one to it each time that the a button is pressed and display it at the end of the game.
3. The game could do with a home screen, something to show before and after games that encourages the pressing of the button to get going.
4. You could play this against the micro:bit and see who wins. You need to work with 2 counter variables. You could copy or adjust the nleds function to allow you to display the LEDs at a different brightness for the AI player. If the player always goes first, you can add the AI logic directly after the player makes a turn (when the a button is pressed). Make sure to check for the AI winning

BBC Microbit

Collapse All Expand All

+ Block Editor - The Basics
+ Block Editor - Components
+ Kodu - micro:bit Worlds
+ JavaScript Blocks
+ JavaScript Blocks - Exercises
+ Blocks - Bit:Bot
+ Blocks - Bit:Commander
+ MicroPython - Starting Off
- MicroPython - Examples
Dice Rolling
Shut The Matrix
Encoding Morse Code
Encoding Ciphers
Drawing A Maze
Scrolling Race Track
Vertical Scroller
Concentration Game
Text Entry - Accelerometer
Charlie's Python Game
Lights Out Game
Sam's French Number Game
Bryn's Concentrated Clocks
Lattice Paths
Knight Moves
+ MicroPython - Components
+ MicroPython - Breakout Boards
+ MicroPython - Exercises
+ MicroPython - Pi Accessories
+ MicroPython - Bit:Bot
+ MicroPython - Bit:Commander
+ MicroPython - Projects
+ MicroPython - Visual Basic
+ Other - Odds & Ends

after its move and use the return statement to quit the function when you have played your failure messages to the loser.