**Lesson\_X\_Notes\_and\_Journal**

**2024.04.XX**

* Look at Code
* Look at Circuit
* Draw Circuit
* Build Circuit
* Start Lesson

**Code**

* Glance at Code

import machine

import utime

import urandom

# Initialize LEDs

led\_pins = [12, 13]

leds = [machine.Pin(pin, machine.Pin.OUT) for pin in led\_pins]

# Initialize buttons

button\_pins = [14, 15]

buttons = [machine.Pin(pin, machine.Pin.IN, machine.Pin.PULL\_UP) for pin in button\_pins]

# This function will flash the given LED number

def flash\_led(led\_number, duration=0.5):

leds[led\_number].value(1)

utime.sleep(duration)

leds[led\_number].value(0)

utime.sleep(duration)

# This function waits for a button press and returns the button number

def wait\_for\_button():

while True:

for i, button in enumerate(buttons):

if button.value() == 0:

# Debounce the button press

utime.sleep\_ms(20)

while button.value() == 0:

pass

return i

# Main game loop

sequence = []

while True:

print("\nNew round! Watch the LED sequence...")

# Add a new step to the sequence

sequence.append(urandom.randint(0, 1))

# Show the sequence to the player

for led\_number in sequence:

flash\_led(led\_number)

print("Now, replicate the sequence using the buttons!")

# Wait for player's input and check if it's correct

for led\_number in sequence:

button\_number = wait\_for\_button()

if button\_number != led\_number:

# Wrong sequence, flash both LEDs as an error signal and restart the game

flash\_led(0, 0.2)

flash\_led(1, 0.2)

print("Wrong sequence! Let's start over.")

sequence = []

break

else:

print(f"Correct button {button\_number+1}!")

# Inform player about the correct sequence input

if len(sequence) > 0 and button\_number == led\_number:

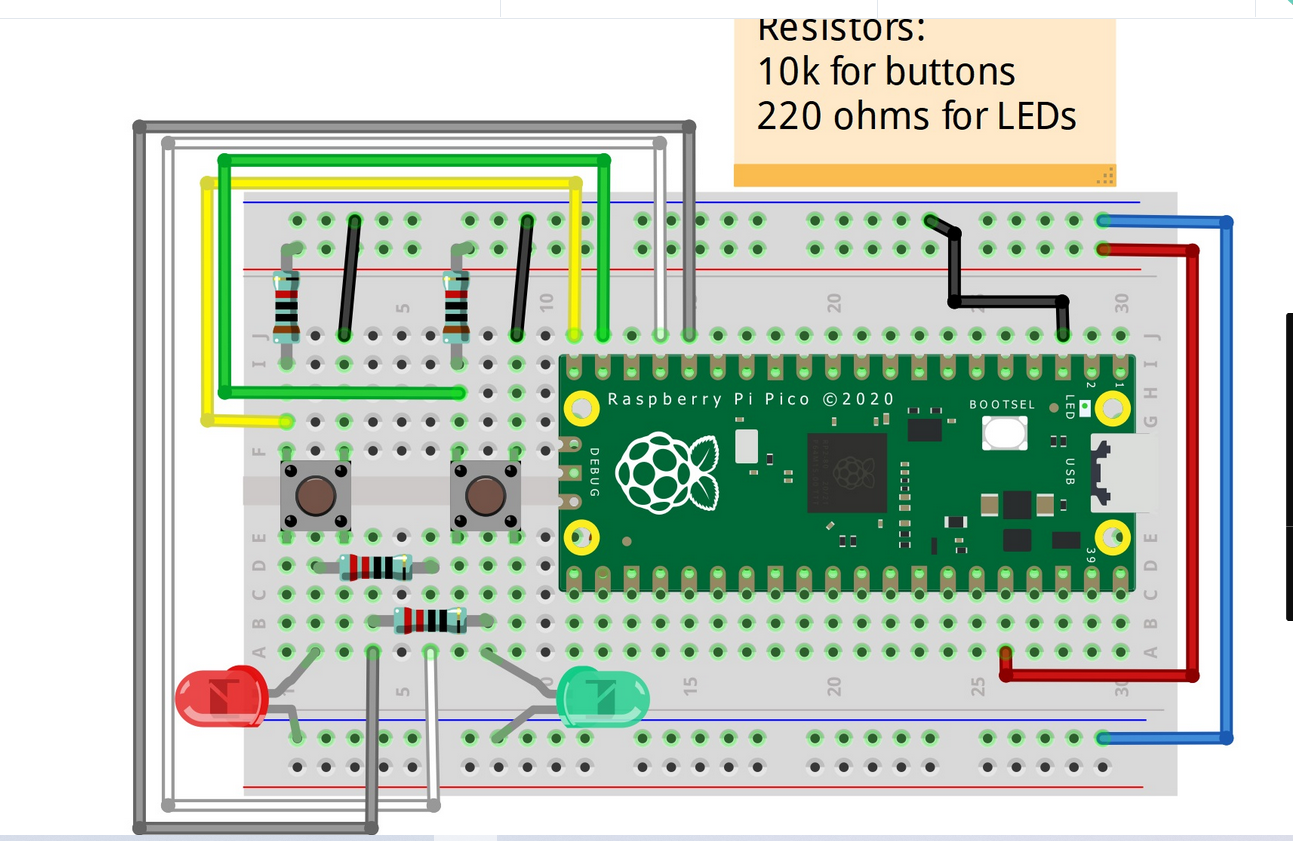
print(f"Good job! Sequence length is now {len(sequence)}")

# Small delay before next sequence

utime.sleep(1)

**Circuit**

* Theirs



* Mine
* As built