Rico Garcia
RBT 173
Simple Simon

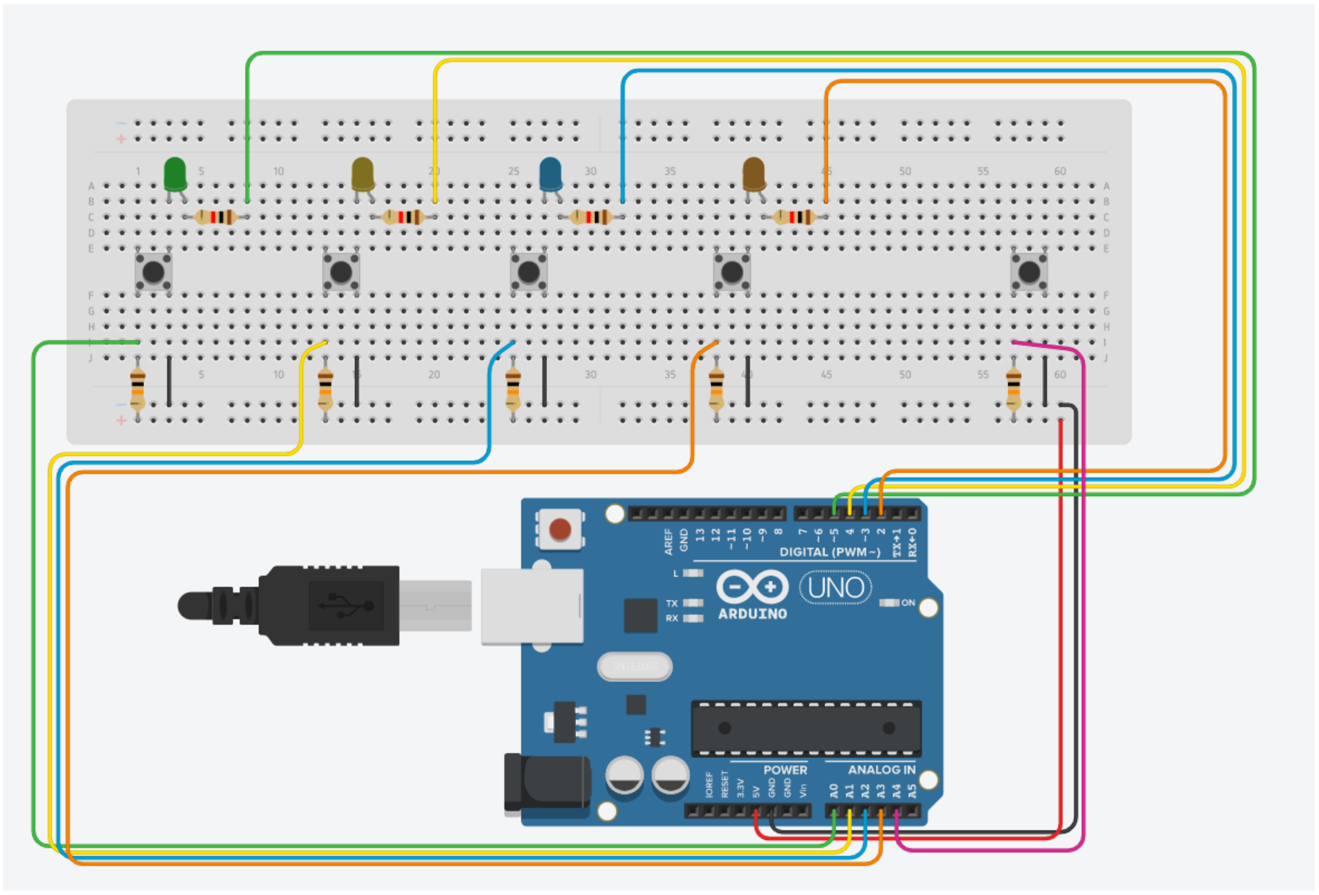
Intro

This assignment is intended to help gain familiarity with digital input and output while using a microcontroller. So, being able to enter a command of some sort, in this case, pressing a button, and then getting a calculated and planned reaction from the device. In this case, the user will be able to press a button and see a direct response to the button being pressed in the form of a memory game where the microcontroller will flash a number of lights in a certain pattern. The user is to mimic those lights by pressing the appropriate buttons in the appropriate sequence. With each proper sequence of buttons pressed, the game gets harder.

Demo

https://youtu.be/CwPJSWyannc

Schematic



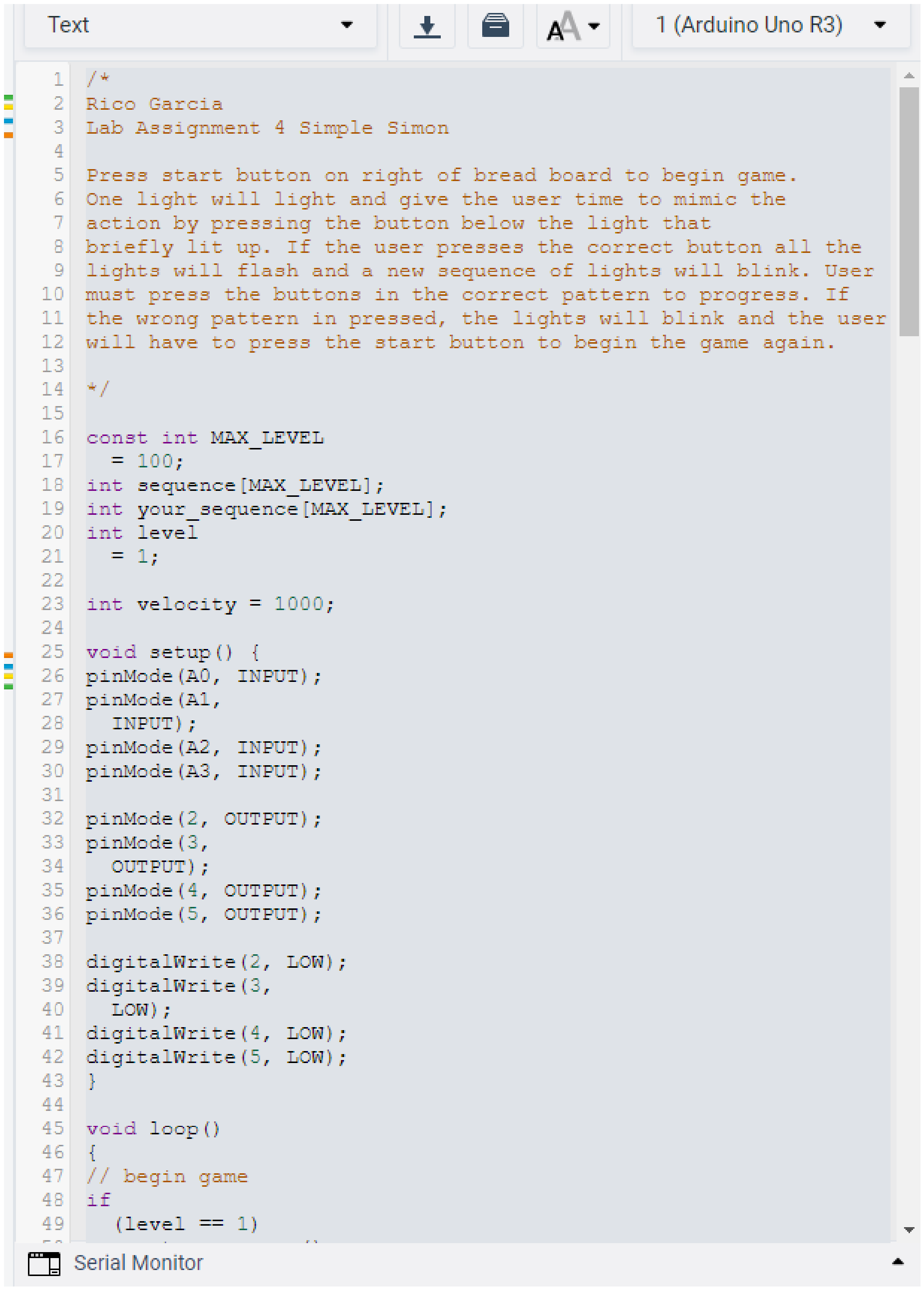
Rico Garcia
RBT 173
Simple Simon

Explanation

The game starts by pressing the start button, the button on the far-right side of the breadboard. This initializes the game and random sequence the game will follow when prompting the user with the blink of a random light. From here the user will press the button directly under the light that flashed. If done correctly, all lights will flash and the sequence will begin again but one extra light will blink after the first. Again the user must press each button in the proper order that the lights blinked. This cycle will repeat a few times or until the user presses a button that does not match up with the sequence of lights. In the event that the sequence is not followed, the lights will all flash once and the game will need to be restarted by pressing the right button.

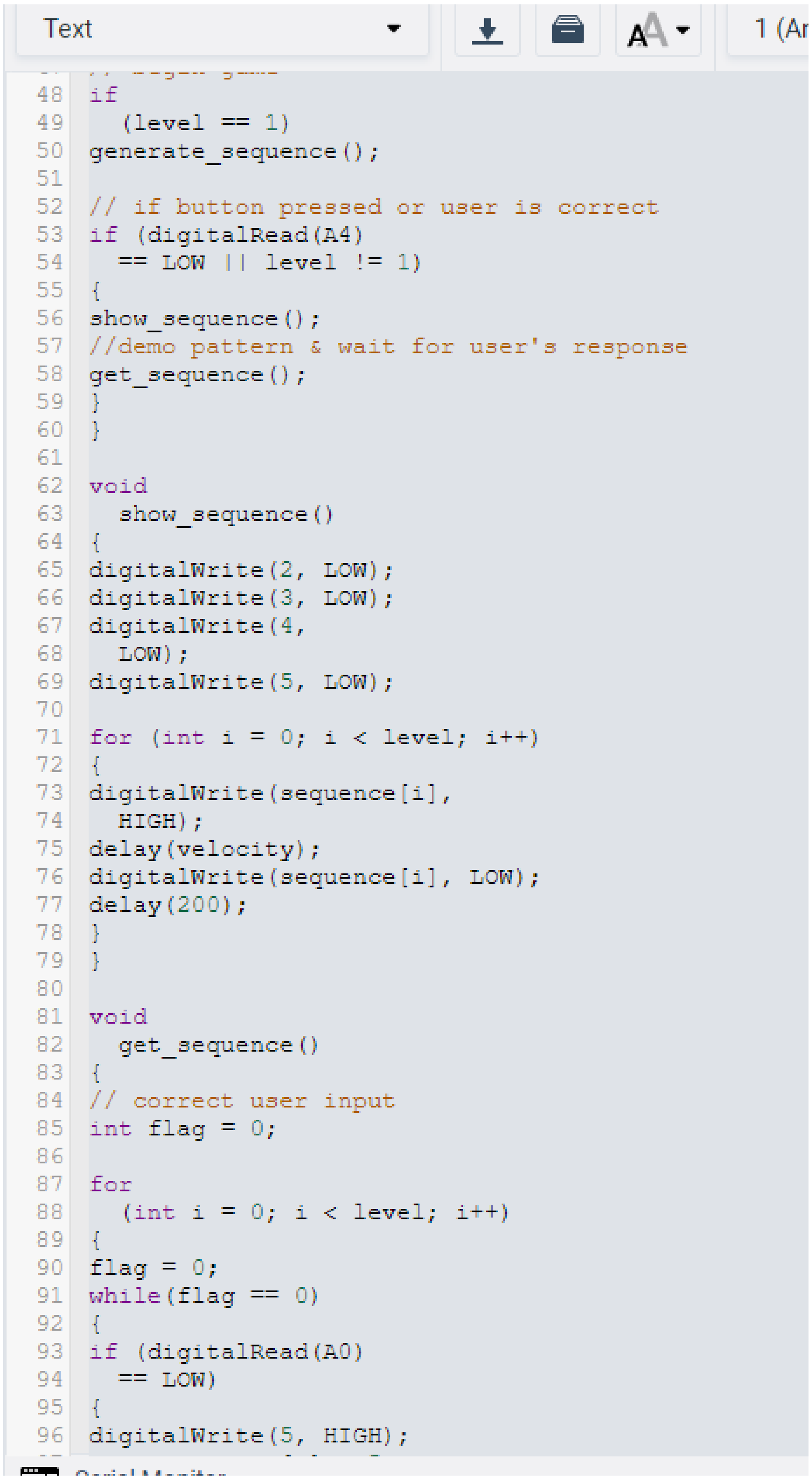
Rico Garcia
RBT 173
Simple Simon

Code

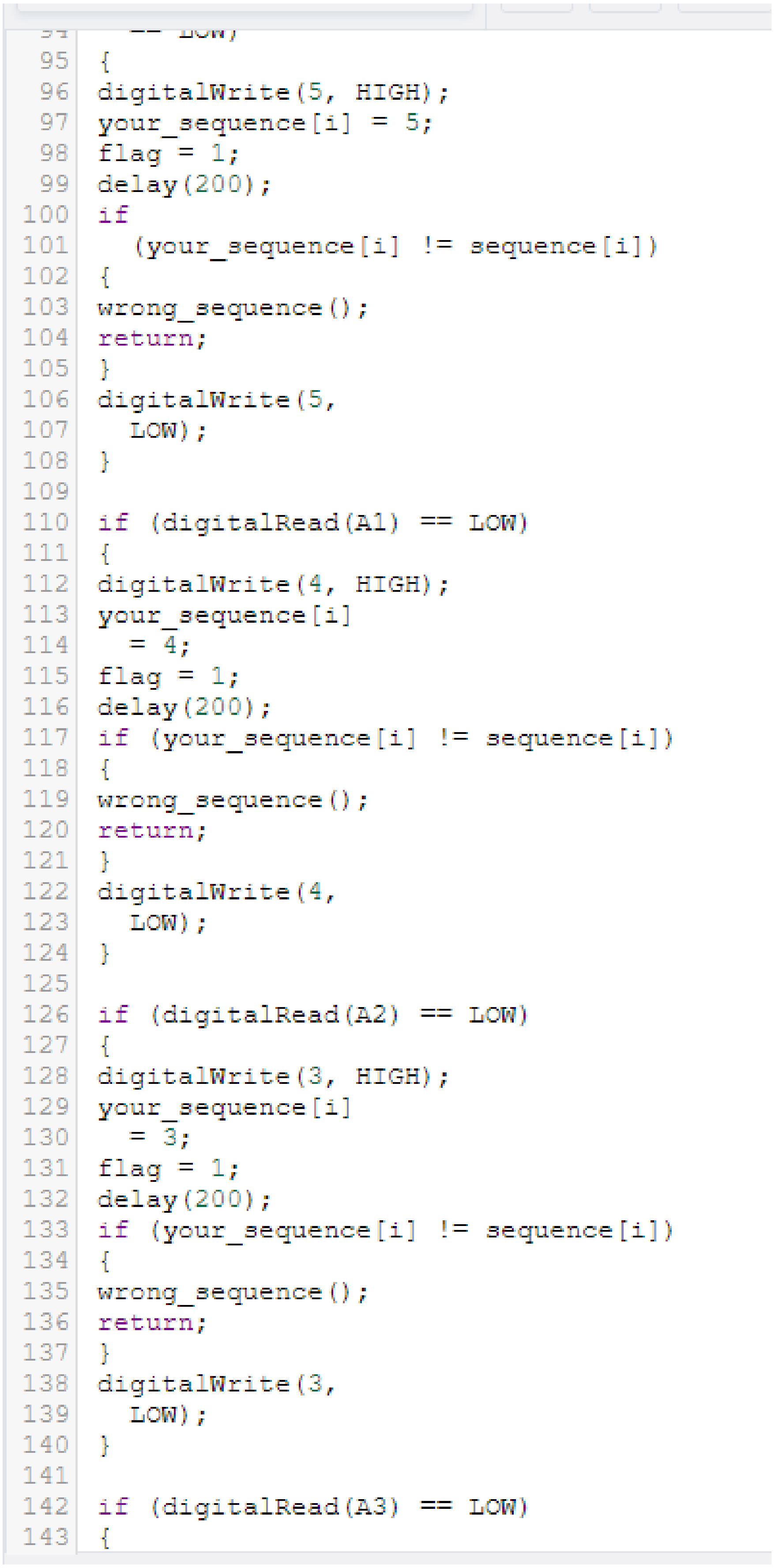


```
/*
Rico Garcia
Lab Assignment 4 Simple Simon

Press start button on right of bread board to begin game.
One light will light and give the user time to mimic the
action by pressing the button below the light that
briefly lit up. If the user presses the correct button all the
lights will flash and a new sequence of lights will blink. User
must press the buttons in the correct pattern to progress. If
the wrong pattern in pressed, the lights will blink and the user
will have to press the start button to begin the game again.

*/

const int MAX_LEVEL
  = 100;
int sequence[MAX_LEVEL];
int your_sequence[MAX_LEVEL];
int level
  = 1;

int velocity = 1000;

void setup() {
pinMode(A0, INPUT);
pinMode(A1,
  INPUT);
pinMode(A2, INPUT);
pinMode(A3, INPUT);

pinMode(2, OUTPUT);
pinMode(3,
  OUTPUT);
pinMode(4, OUTPUT);
pinMode(5, OUTPUT);

digitalWrite(2, LOW);
digitalWrite(3,
  LOW);
digitalWrite(4, LOW);
digitalWrite(5, LOW);
}

void loop()
{
// begin game
if
  (level == 1)
```

Rico Garcia
RBT 173
Simple Simon

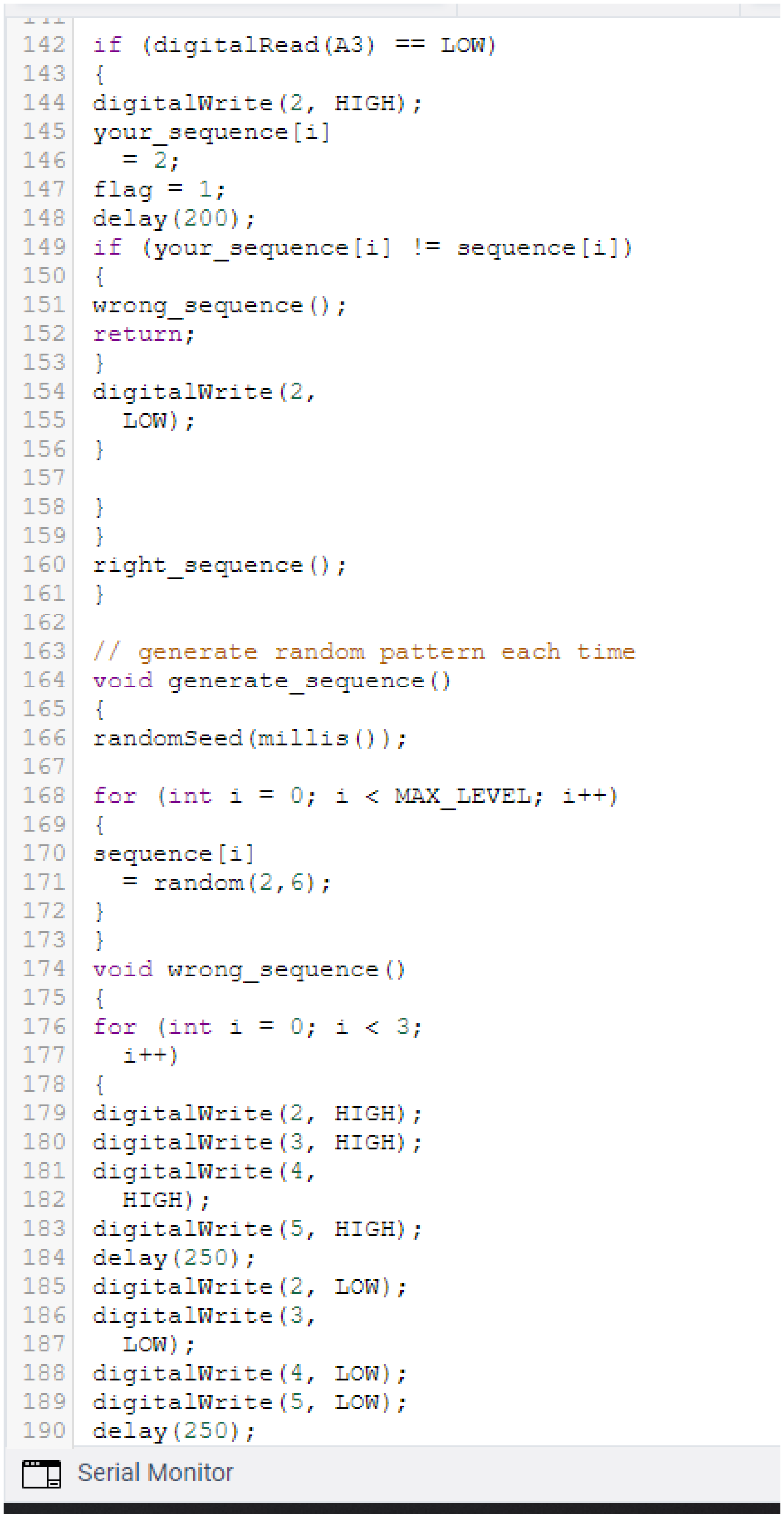


```
if
  (level == 1)
generate_sequence();

// if button pressed or user is correct
if (digitalRead(A4)
  == LOW || level != 1)
{
show_sequence();
//demo pattern & wait for user's response
get_sequence();
}
}

void
  show_sequence()
{
digitalWrite(2, LOW);
digitalWrite(3, LOW);
digitalWrite(4,
  LOW);
digitalWrite(5, LOW);

for (int i = 0; i < level; i++)
{
digitalWrite(sequence[i],
  HIGH);
delay(velocity);
digitalWrite(sequence[i], LOW);
delay(200);
}
}

void
  get_sequence()
{
// correct user input
int flag = 0;

for
  (int i = 0; i < level; i++)
{
flag = 0;
while(flag == 0)
{
if (digitalRead(A0)
  == LOW)
{
digitalWrite(5, HIGH);
```



```
  == LOW)
{
digitalWrite(5, HIGH);
your_sequence[i] = 5;
flag = 1;
delay(200);
if
  (your_sequence[i] != sequence[i])
{
wrong_sequence();
return;
}
digitalWrite(5,
  LOW);
}

if (digitalRead(A1) == LOW)
{
digitalWrite(4, HIGH);
your_sequence[i]
  = 4;
flag = 1;
delay(200);
if (your_sequence[i] != sequence[i])
{
wrong_sequence();
return;
}
digitalWrite(4,
  LOW);
}

if (digitalRead(A2) == LOW)
{
digitalWrite(3, HIGH);
your_sequence[i]
  = 3;
flag = 1;
delay(200);
if (your_sequence[i] != sequence[i])
{
wrong_sequence();
return;
}
digitalWrite(3,
  LOW);
}

if (digitalRead(A3) == LOW)
{
```

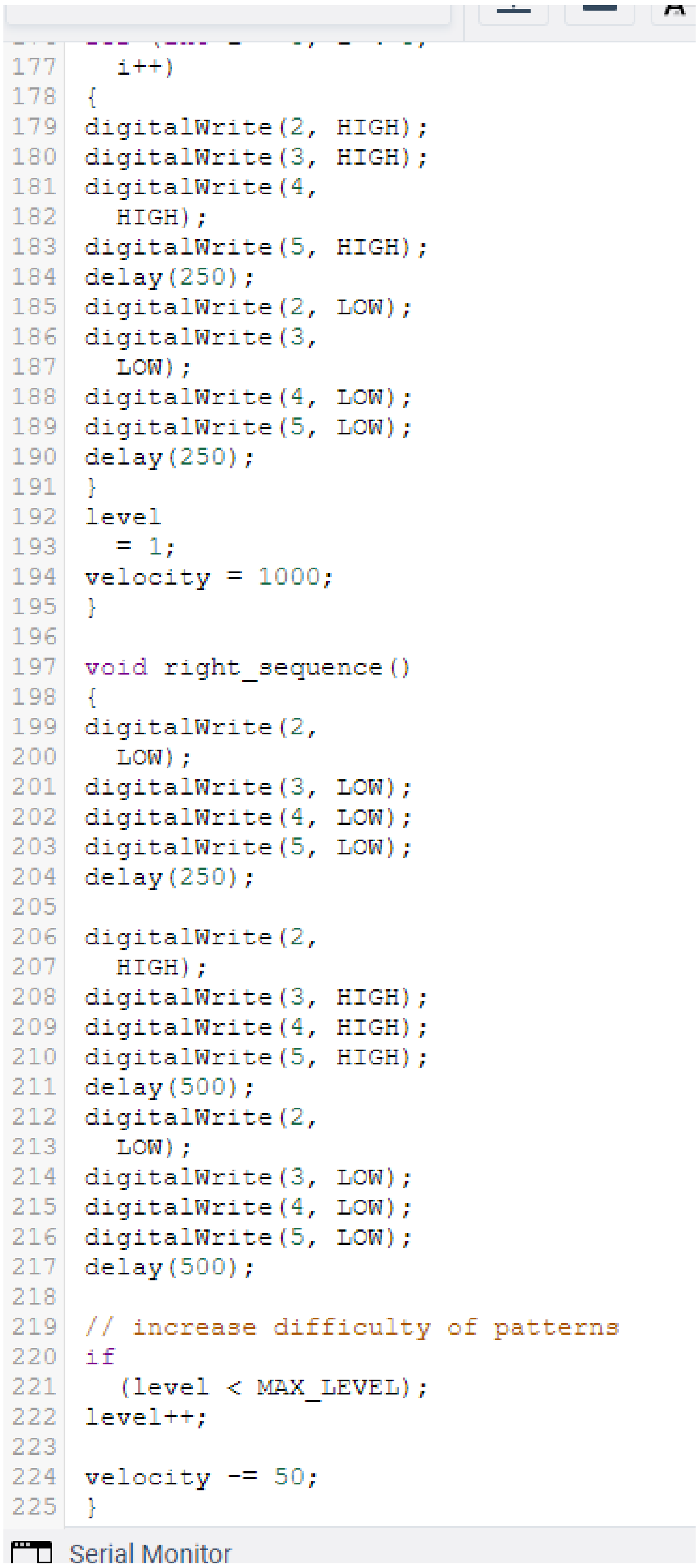
Rico Garcia
RBT 173
Simple Simon



```
if (digitalRead(A3) == LOW)
{
digitalWrite(2, HIGH);
your_sequence[i]
  = 2;
flag = 1;
delay(200);
if (your_sequence[i] != sequence[i])
{
wrong_sequence();
return;
}
digitalWrite(2,
  LOW);
}

}
}
right_sequence();
}

// generate random pattern each time
void generate_sequence()
{
randomSeed(millis());

for (int i = 0; i < MAX_LEVEL; i++)
{
sequence[i]
  = random(2,6);
}
}
void wrong_sequence()
{
for (int i = 0; i < 3;
  i++)
{
digitalWrite(2, HIGH);
digitalWrite(3, HIGH);
digitalWrite(4,
  HIGH);
digitalWrite(5, HIGH);
delay(250);
digitalWrite(2, LOW);
digitalWrite(3,
  LOW);
digitalWrite(4, LOW);
digitalWrite(5, LOW);
delay(250);
```

Serial Monitor

Rico Garcia
RBT 173
Simple Simon



```
  i++)
{
digitalWrite(2, HIGH);
digitalWrite(3, HIGH);
digitalWrite(4,
  HIGH);
digitalWrite(5, HIGH);
delay(250);
digitalWrite(2, LOW);
digitalWrite(3,
  LOW);
digitalWrite(4, LOW);
digitalWrite(5, LOW);
delay(250);
}
level
  = 1;
velocity = 1000;
}

void right_sequence()
{
digitalWrite(2,
  LOW);
digitalWrite(3, LOW);
digitalWrite(4, LOW);
digitalWrite(5, LOW);
delay(250);

digitalWrite(2,
  HIGH);
digitalWrite(3, HIGH);
digitalWrite(4, HIGH);
digitalWrite(5, HIGH);
delay(500);
digitalWrite(2,
  LOW);
digitalWrite(3, LOW);
digitalWrite(4, LOW);
digitalWrite(5, LOW);
delay(500);

// increase difficulty of patterns
if
  (level < MAX_LEVEL);
level++;

velocity -= 50;
}
```