Arduino C++ Programming

Advanced Concepts

https://www.arduino.cc/reference/en/

Basic Sketch-setup() and loop()

setup()

-called once at the start of your program

Use it to initialize variables, pin modes, start using libraries etc.

loop()

-it loops over and over again executing its code over and over again

Use it to actively control the microcontroller

Tinkercad and Wokwi

We will use both the Arduino IDE, Tinkercad and Wokwi to create and code microcontroller based projects in this class.

Arduino IDE→search for Arduino in your windows search bar

https://www.tinkercad.com/

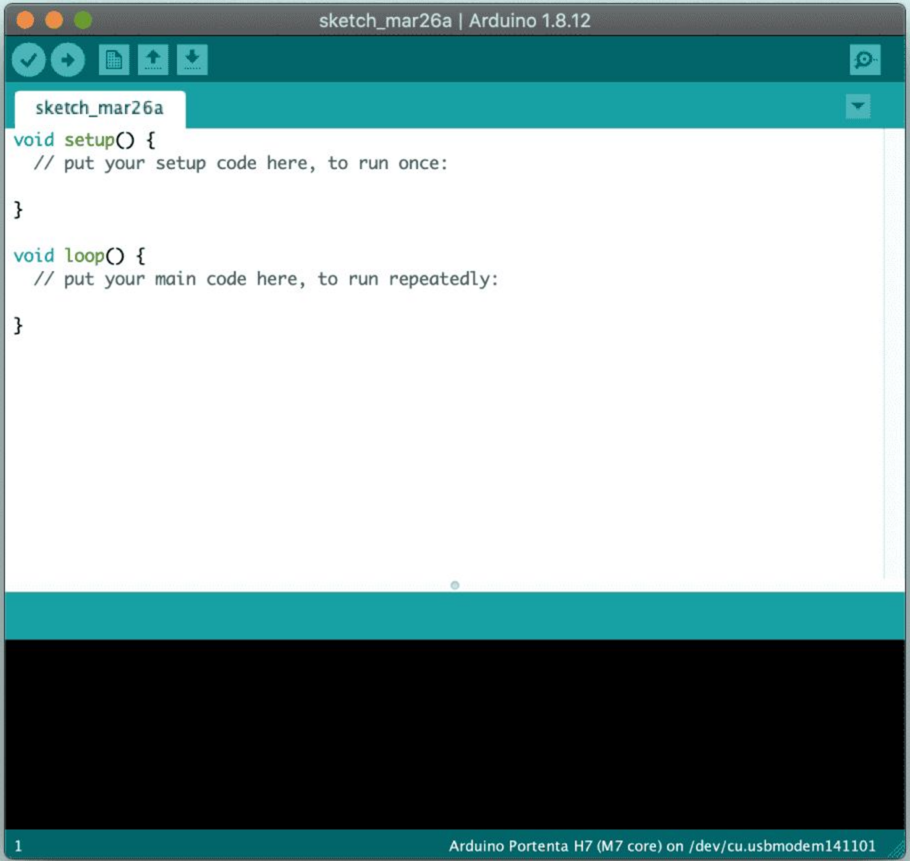
https://wokwi.com/

Tinkercad is used mainly for work with the Arduino Uno.

Wokwi will be used mainly for work with the ESP32 microcontroller.

Make sure to create accounts at both places and to join my classroom at Tinkercad. Ask for my class link code if you haven't joined the class yet.

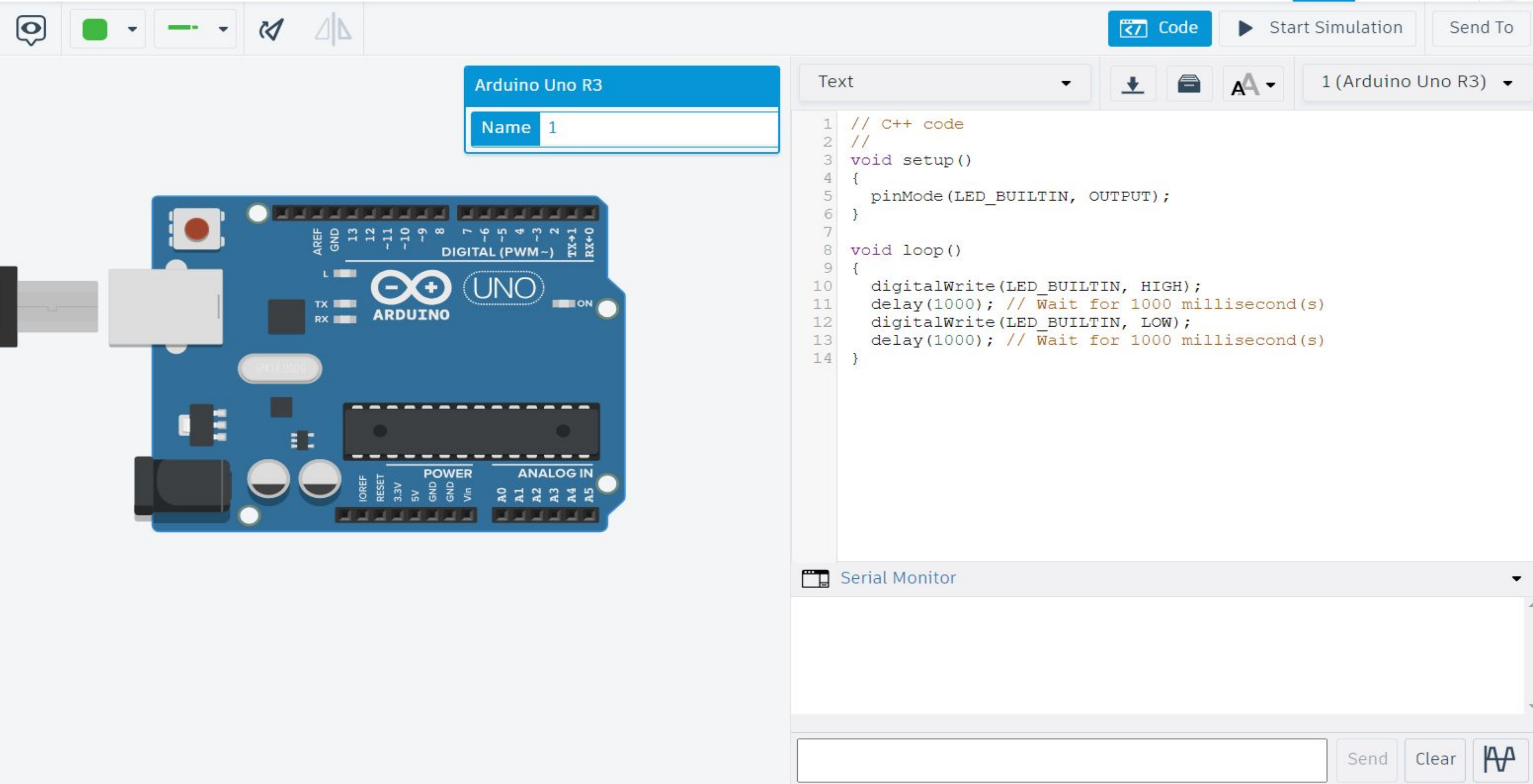
Arduino IDE



```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

1 Arduino Portenta H7 (M7 core) on /dev/cu.usbmodem141101

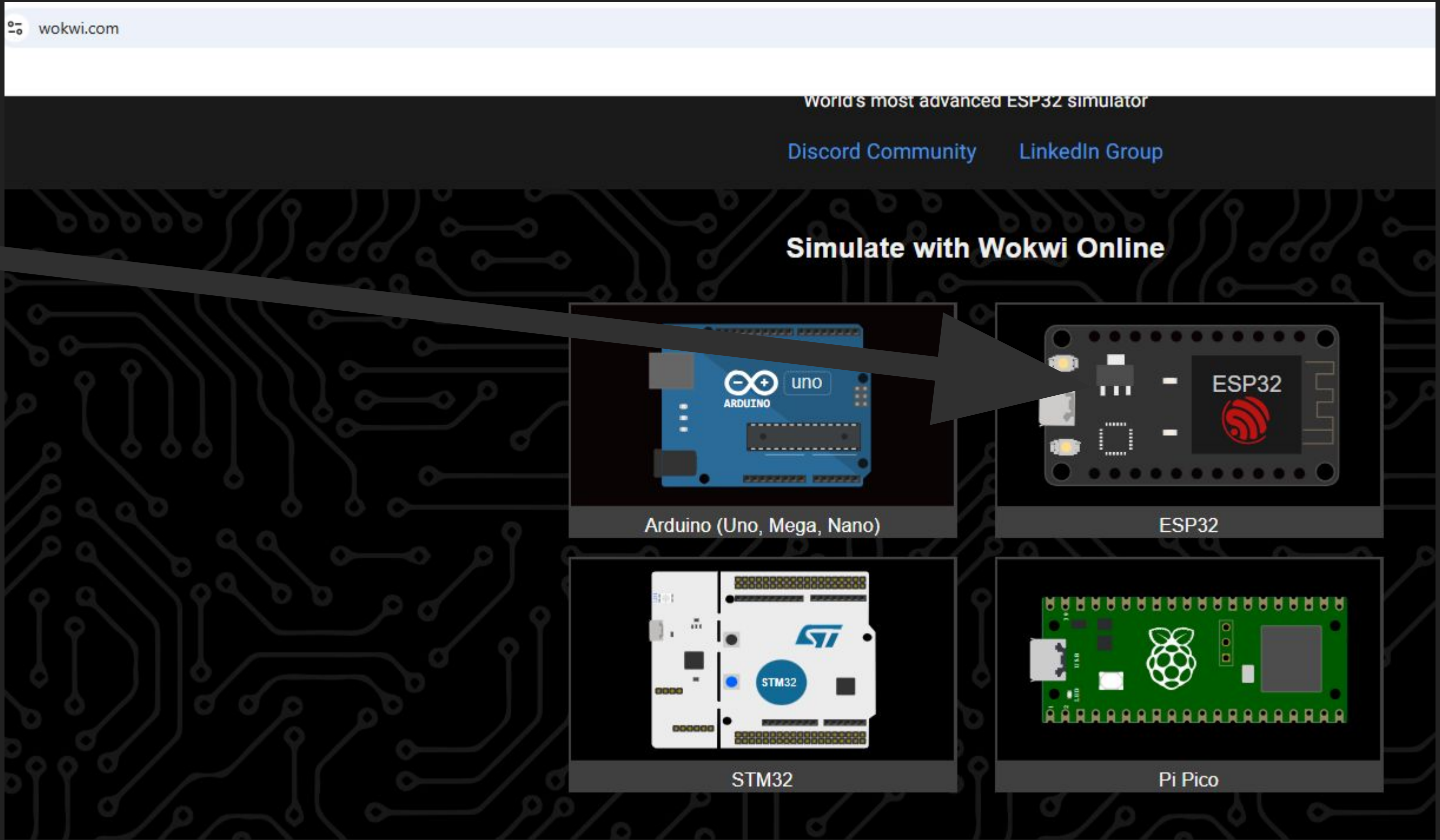
Tinkercad Code



```
// C++ code
//
void setup()
{
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop()
{
  digitalWrite(LED_BUILTIN, HIGH);
  delay(1000); // Wait for 1000 millisecond(s)
  digitalWrite(LED_BUILTIN, LOW);
  delay(1000); // Wait for 1000 millisecond(s)
}
```

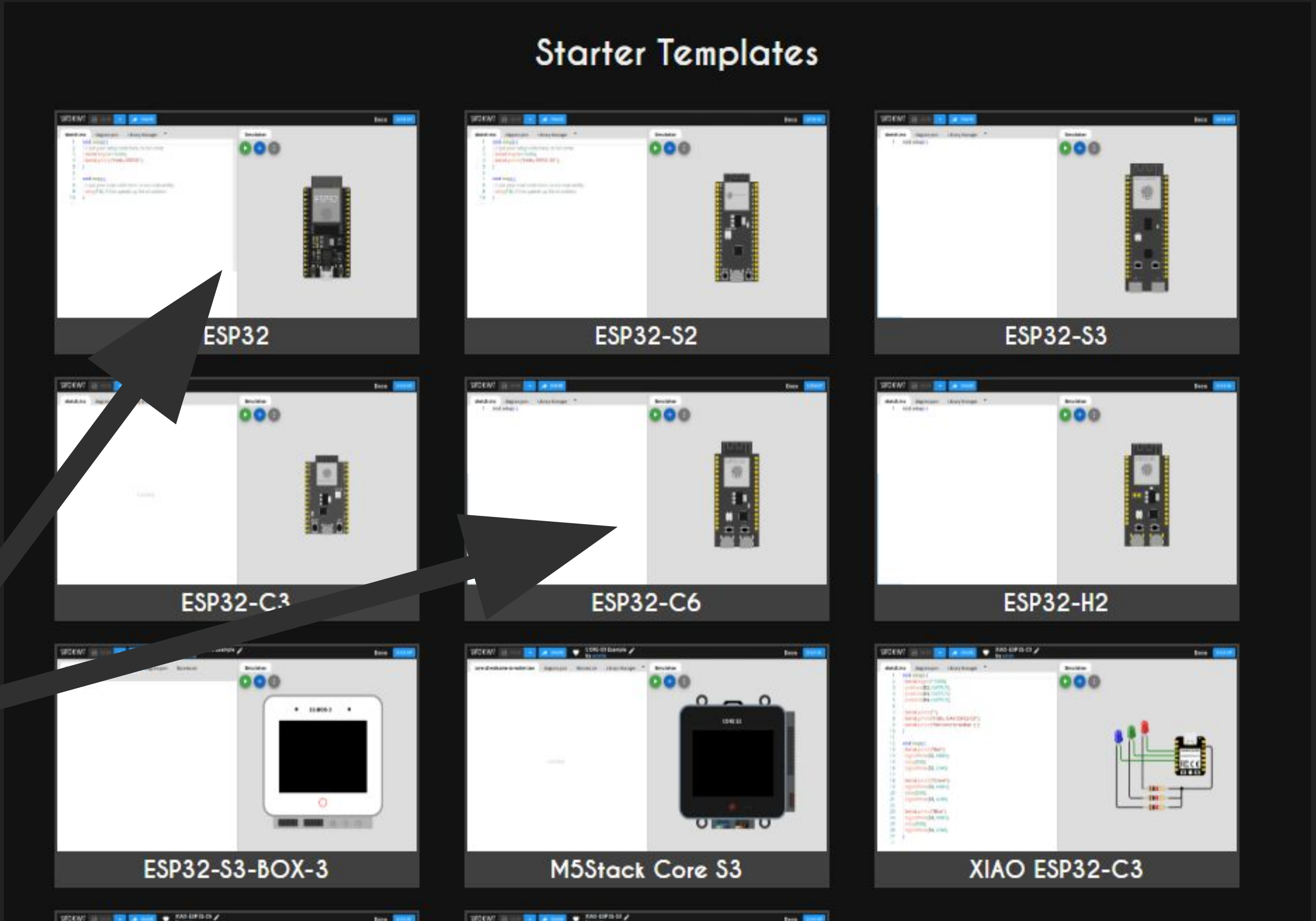
Wokwi Code

Go to wokwi.com and select the ESP32 templates.

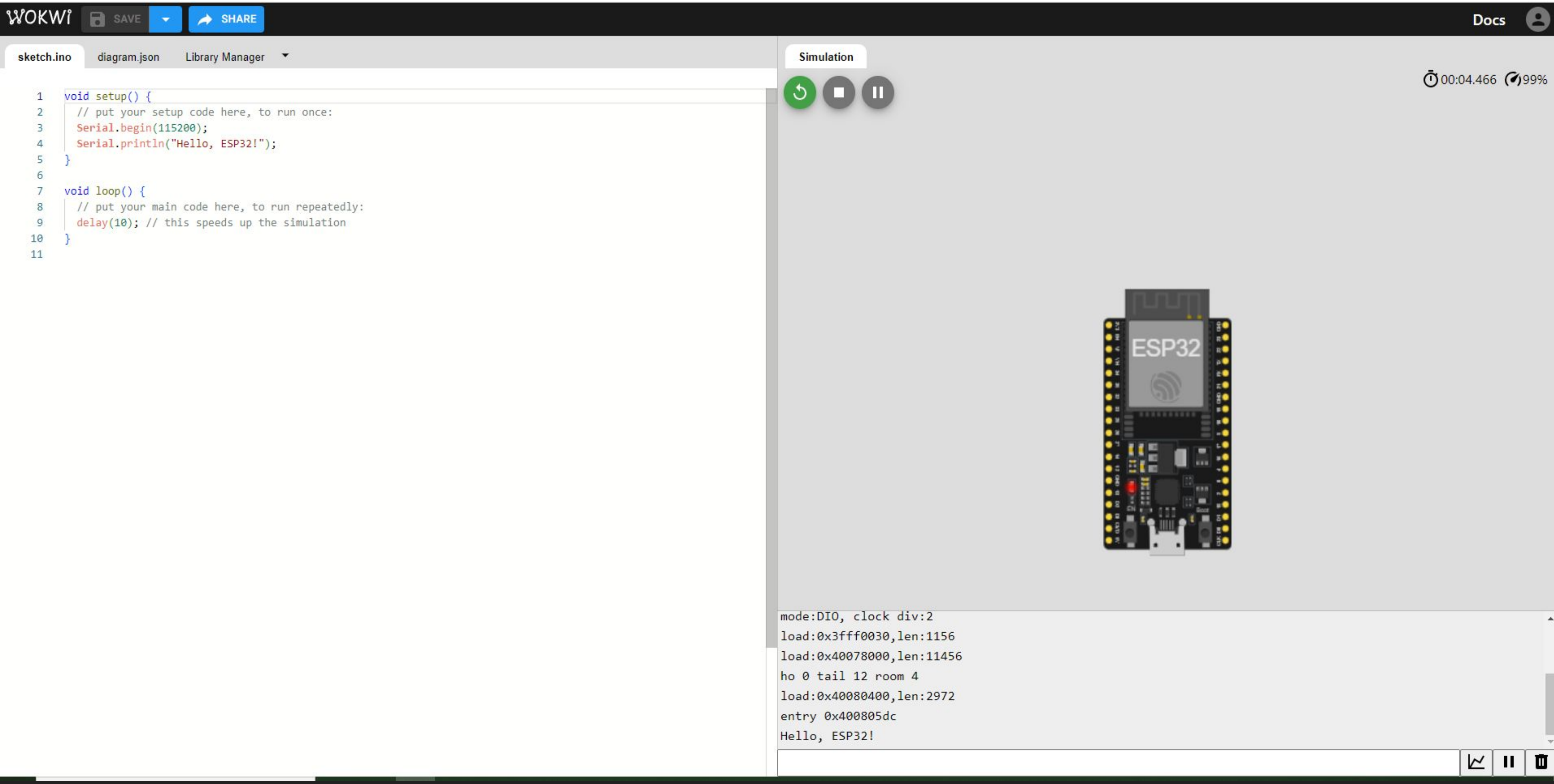

wokwi.com
World's most advanced ESP32 simulator
Discord Community
LinkedIn Group
Simulate with Wokwi Online
Arduino (Uno, Mega, Nano)
ESP32
STM32
Pi Pico

Starter Templates

We will be working with the base ESP2 and ESP32-C6 microcontrollers so you can pick either depending on what we are using at the time. For now the base ESP32 should be selected



Wokwi Code



```
void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
  Serial.println("Hello, ESP32!");
}

void loop() {
  // put your main code here, to run repeatedly:
  delay(10); // this speeds up the simulation
}
```

```
mode:DIO, clock div:2
load:0x3fff0030,len:1156
load:0x40078000,len:11456
ho 0 tail 12 room 4
load:0x40080400,len:2972
entry 0x400805dc
Hello, ESP32!
```

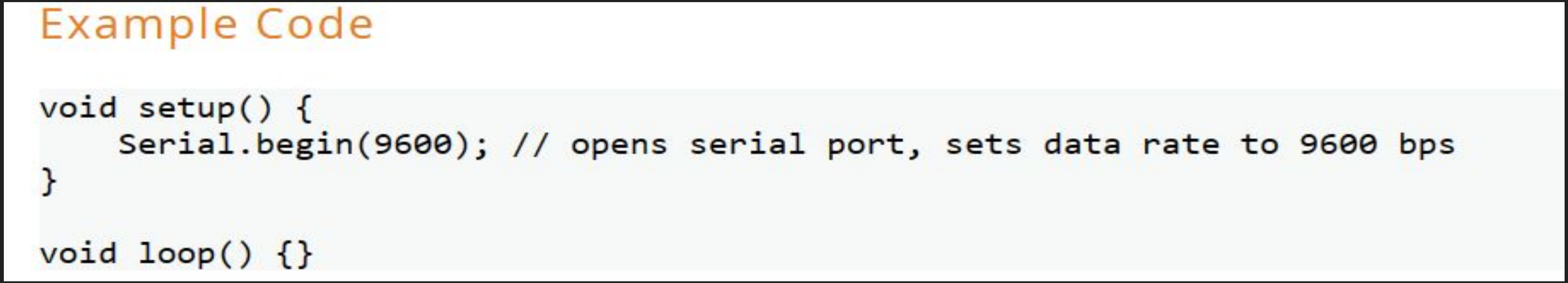
Serial (https://www.arduino.cc/reference/en/language/functions/communication/serial/begin/)

-used to communicate with the microcontroller over a serial port i.e. COM ports

-has many functions to allow for data to be sent to and read from the microcontroller

-the most common function is begin() which sets the data rate for communication between the computer and the microcontroller

-for the Arduino this rate is 9600 and for the ESP32 its 115200



Example Code

```
void setup() {
    Serial.begin(9600); // opens serial port, sets data rate to 9600 bps
}

void loop() {}
```

print() and println()

-these functions are used to send data to the Serial Monitor (**Tools>serial Monitor**)

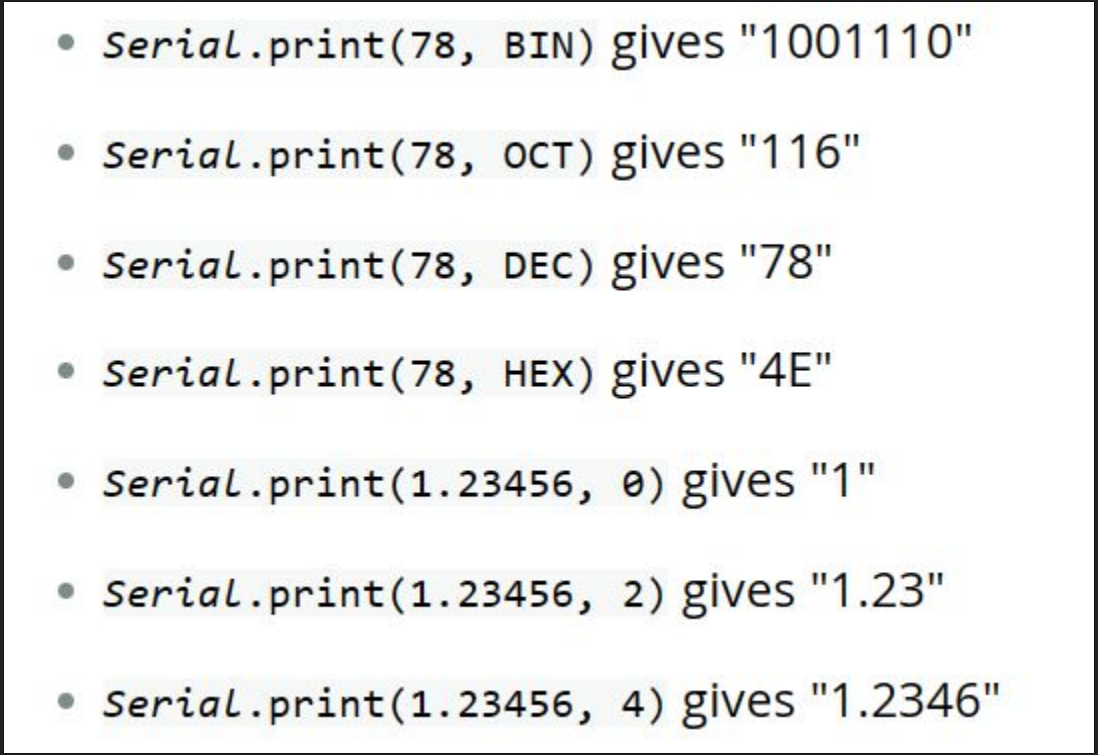
-println() sends a carriage return or line break at the end

-print() does not include a line break

-both send the data to the Serial Monitor as text or a string

-both can send data formatted using a second optional parameter

Formatting



- `Serial.print(78, BIN)` gives "1001110"
- `Serial.print(78, OCT)` gives "116"
- `Serial.print(78, DEC)` gives "78"
- `Serial.print(78, HEX)` gives "4E"
- `Serial.print(1.23456, 0)` gives "1"
- `Serial.print(1.23456, 2)` gives "1.23"
- `Serial.print(1.23456, 4)` gives "1.2346"

Escape Characters

\t → tab space

\n →line break

Serial.write()

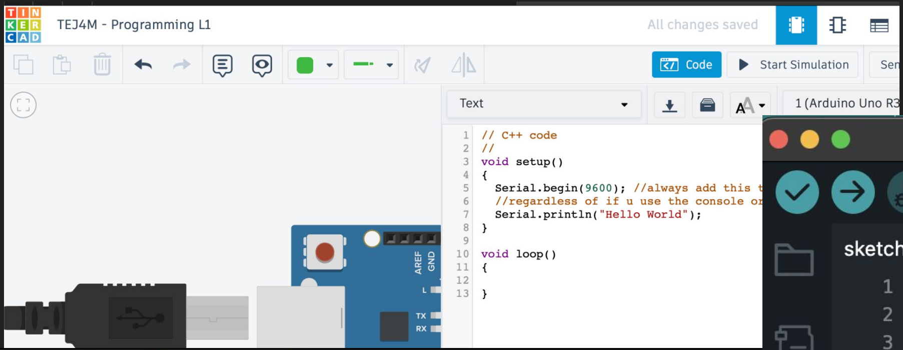
-sends the binary representation of the written value

-for example Serial.write(99); actually sends 1100011 to the Serial Monitor which in turn converts it back to its string representation

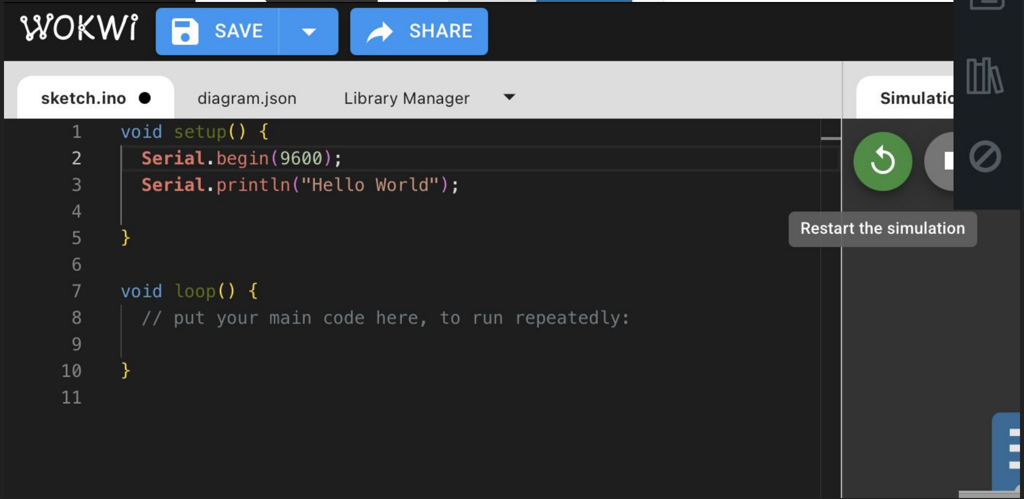
-1100011 as a string is the letter “c” (you can check this by searching up the ASCII chart)

-if you write() a string it sends the string as an array of bytes ex. Serial.write(“cat”) sends “c”, then “a” and then “t”

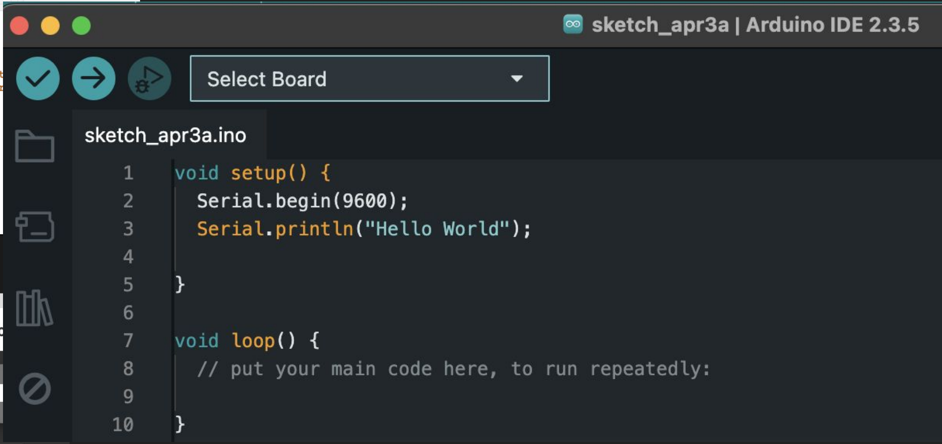
1. Create an Arduino program that says “Hello World!”. Do this at Tinkercad, Wokwi and in the Arduino IDE.



```
// C++ code
//
void setup()
{
  Serial.begin(9600); //always add this t
  //regardless of if u use the console or
  Serial.println("Hello World");
}

void loop()
{

}
```

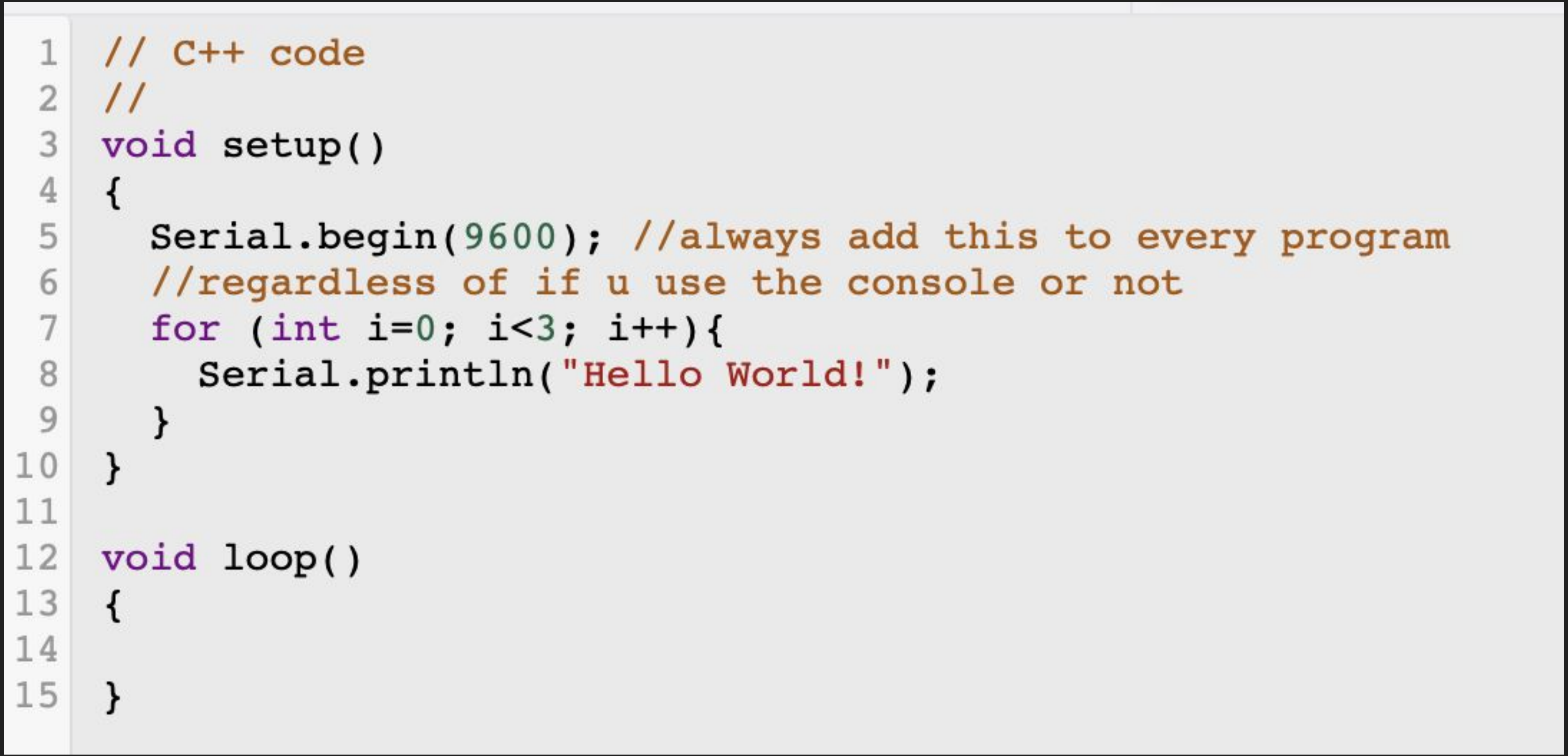


```
void setup() {
  Serial.begin(9600);
  Serial.println("Hello World");

}

void loop() {
  // put your main code here, to run repeatedly:

}

```



```
void setup() {
  Serial.begin(9600);
  Serial.println("Hello World");

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

2Create a program that says “Hello World!” 3 times on separate lines.



```
// C++ code
//
void setup()
{
  Serial.begin(9600); //always add this to every program
  //regardless of if u use the console or not
  for (int i=0; i<3; i++){
    Serial.println("Hello World!");
  }
}

void loop()
{

}
```

3. Using tab spaces create a program that display something like the following in the Serial Monitor:

```
     Date          Time          Result
```

```
// C++ code
//
void setup()
{
  Serial.begin(9600);
  Serial.println("Date\t\tTime\t\tResult");
}

void loop()
{

}
```

4. Write a program using only Serial.write and only numbers to display the word “cat” on the Serial Monitor (hint: use the ASCII chart)

```
// C++ code
//
void setup()
{
  Serial.begin(9600);
  Serial.println(" 000 \t  0  \t00000\n0   0\t 0 0 \t  0  \n0    \t00000\t  0  \n0   0\t0   0\t  0  \n 000 \t0   0\t  0  \n");
}

void loop()
{

}
```

5. Write a program to display a table of decimal numbers and their binary, octal, and hex equivalents. Show the results for decimal numbers 5,10,15, and 20. Use tab spaces to make the table look nice.

```
void setup()
{
  Serial.begin(9600);
  Serial.println("Decimal\tBin\t\tOct\tHex");
  Serial.println("5\t0101\t\t5\t5");
  Serial.println("10\t1010\t\t12\tA");
  Serial.println("15\t1111\t\t17\tF");
  Serial.println("20\t0001 0100\t24\t14");
}
```

6. Find and list at least 3 other functions that belong to Serial.

Serial.print(data)
Sends data to the serial port as human-readable text (e.g., Serial.print("Hello");).

Serial.read()
Reads incoming serial data (returns the first byte of incoming data).

Serial.available()
Returns the number of bytes available for reading from the serial buffer.

```
void setup() {
  Serial.print(7);
  Serial.print("\t\t");
  Serial.print(7, DEC);
  Serial.print("\t");
  Serial.print(7, HEX);
  Serial.print("\t");
  Serial.print(7, OCT);
  Serial.print("\t");
  Serial.println(7, BIN);
}
```