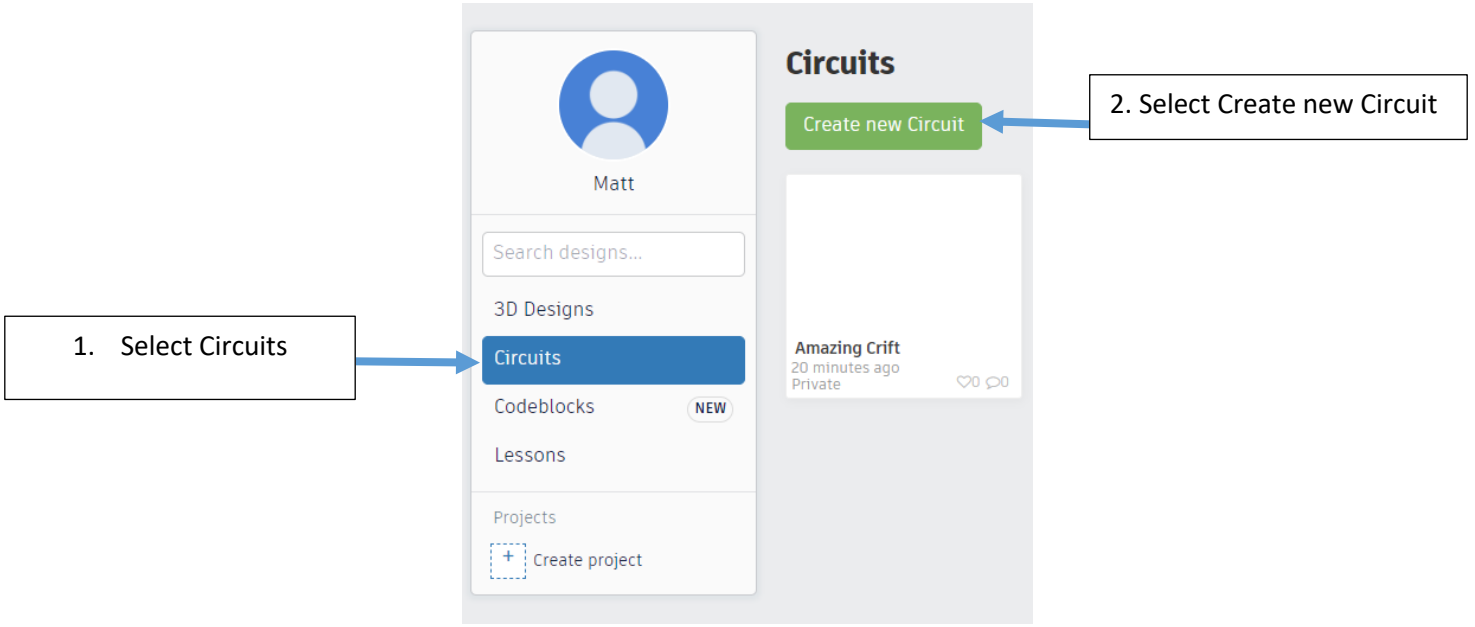
Tutorial TinkerCAD Electrical Series Circuit

By: Matthew Jourden

Brighton High School
Brighton, MI

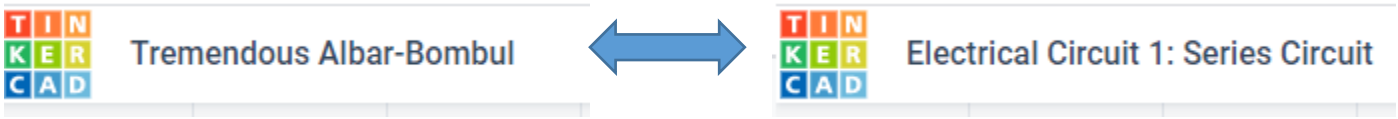
1. Navigate to TinkerCAD.com > Click Sign In Icon (Top Left Side of Screen > Select Students, Join your Classroom > Classroom CODE: HEQ2VA3WQT4N > Nickname: Student First Name (all lower case)

2. Select Circuits > Select Create New Circuit



3. Change Name of Circuit to Electrical Circuit 1: Series Circuit

   Select Default File name in the Top Left Corner



4. Adding Electrical Components. TinkerCAD is a Drag and Drop interface
5. Rename File > TinkerCAD Electrical 3: Arduino 1

Arduino: is an open-source hardware and software that allows users to wire a circuit using resistors, buttons, sensors, etc. and write a program that can receive and transmit data to control the circuit.

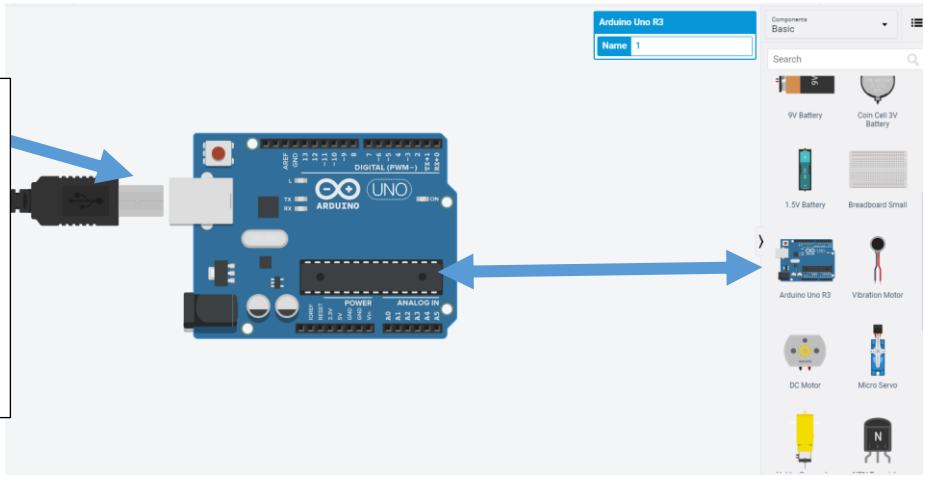
Arduino has its own IDE Software where the user programs then downloads to the Arduino Board. TinkerCAD offers a simulation, that allows the user to create a simulated Arduino board and its circuit and then the user can write a program to receive and/or transmit data to control said circuit.

Arduino uses a C++ based programming language. Arduino Programming Syntax is very similar to C++, so things like comparison (IF/THEN Statement), Mathematical Computations, Loops (FOR, DO, DO/WHILE), Ending of Line Statements (;), etc. are written the same. What differs is how to INPUT/OUTPUT data varies.
See Reference Document: Arduino Common Syntax on the class website for common coding commands.

Objective: First Tutorial is designed to simply wire a single LED and then Program its use.

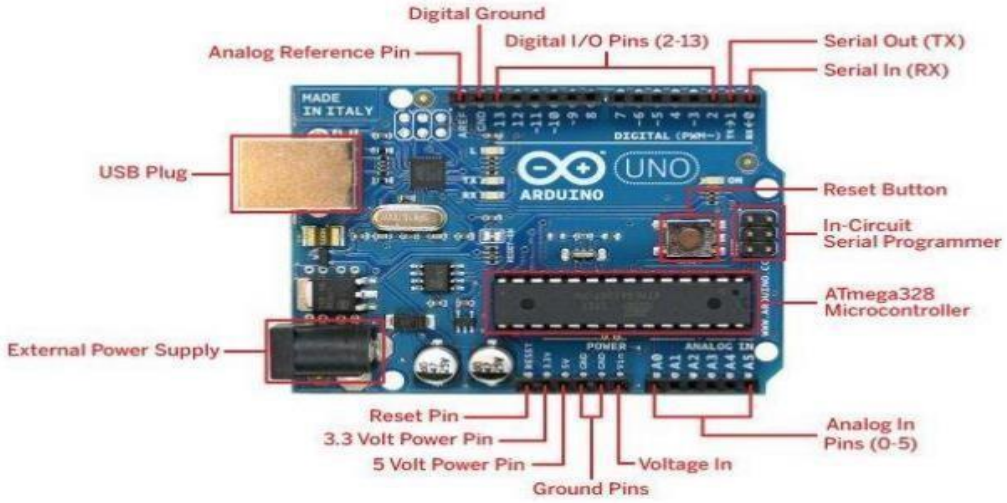
1. Electrical Component Toolbox > Find Arduino Uno > Drag and Drop Arduino Uno onto screen

Power Cord to Computer. Power Cord will take the place of a battery. When simulation runs then the power cord will plug into the Arduino board automatically.



Scroll Down in the Basic Category

2. Arduino Board Overview



Digital I/O Pins 0-13: Operate in an On/OFF State. Meaning what is being controlled in these ports is either On or OFF. Example would be a push button: it is either pressed or it is not, there is no in-between state.

Analog In Pins 0-5: Operate within a range of values, which allows the user to bring in data that is constantly changing. Example: Thermocouple (Thermometer) is constantly sensing temperature and has a range of values that it falls into.

3. Wire the following Circuit (Change Electrical Component Ratings and Wire Color as shown)

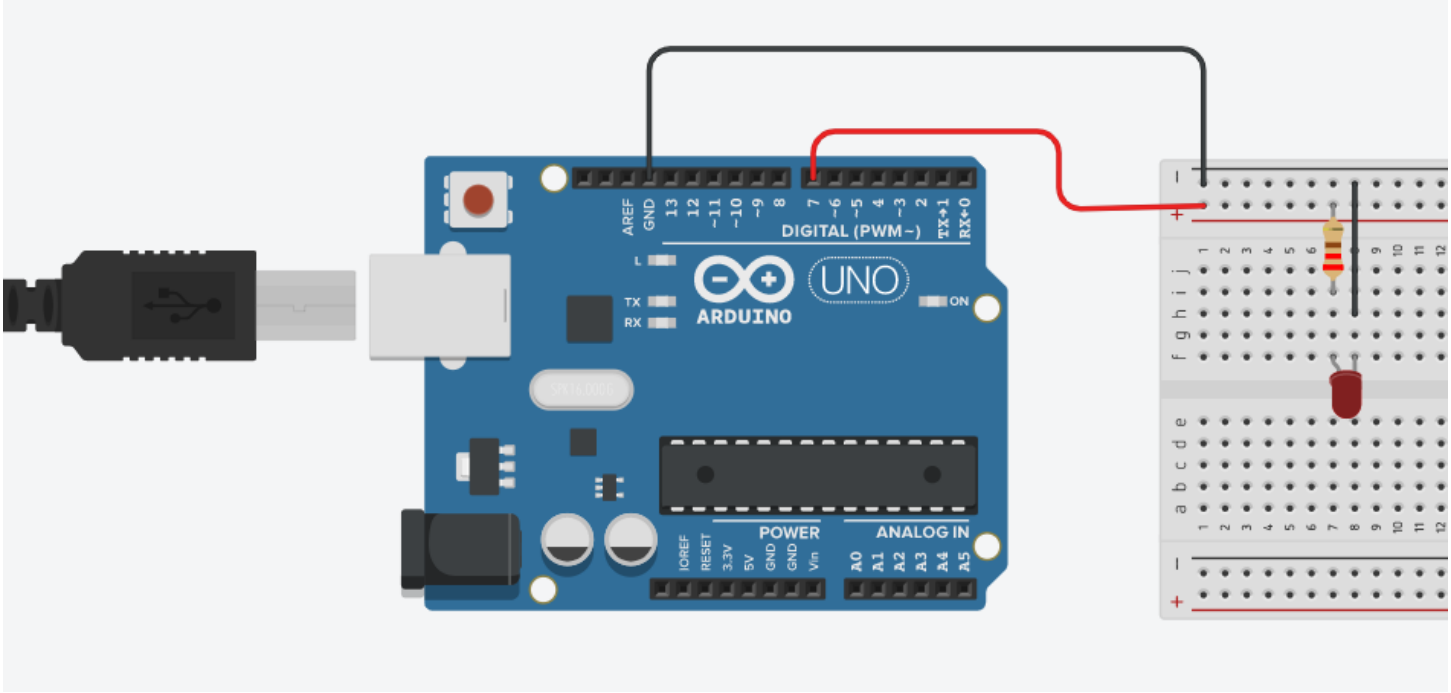
   a. Start Digital Pin 7: The digital pins act as a switch to turn on/off the flow of electricity to the circuit. On the Arduino Board is a 5V and 3.3V Port that can be used to apply direct electricity to a circuit without
   b. Drag Wire to + Row on Breadboard
   c. Place Resistor = 220 Ω starting from + row on breadboard to adjacent row

   Notice: The end of the Resistor is not touching the Red Wire from Arduino. Once a wire is in either the + or – row the whole row is charged with electricity

   d. Place LED with the pins in different columns. NOTE: Anode (Bent Wire) should be on the Resistor Side and the Cathode (Straight Wire) should be leading towards the – on the board.

   Notice: The end of the Resistor is not touching the end of the LED. Once a wire is placed in the column a-e or f-j the whole column (vertically) is charged with electricity

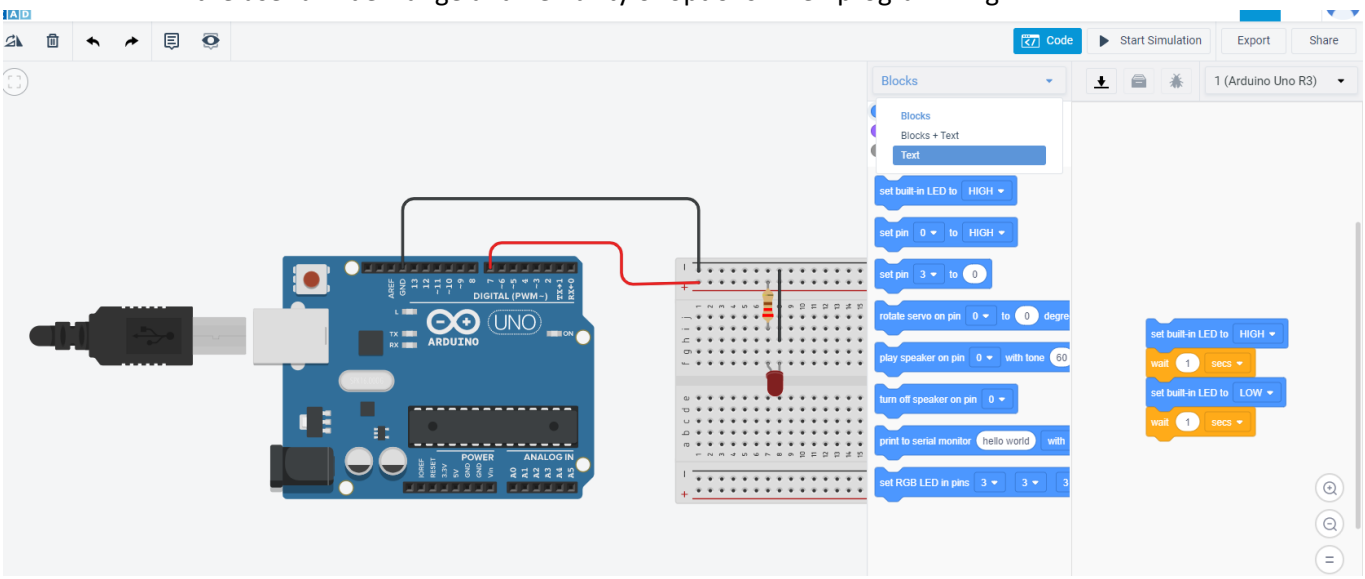
   e. Drag a wire from Cathode side of the LED to the – Row on breadboard
   f. Drag a wire from – row on breadboard to GND (Ground) on the Arduino (NOTE: the Arduino Board has 3 GND (Ground Ports). It does not matter which of these ports the user places the wire.



4. Select the Code Button on the Top Right Hand Side of the Screen > Select Drop Down Menu (Defaults at Blocks) > Change to Text > Pop-Up Message will appear > Press Continue

   TinkerCAD allows the user to program in 1 of 2 ways

   a. Block: Which is designed with blocks of prebuilt code and link together like puzzle pieces. Similar to Scratch or Snap. Block is an easier form of programming and is a good place to start, but there is less flexibility when doing complex computations, controlling of multiple motors/sensors, etc.
   b. Text: is C++ based programming language. The user will type the code to operate the circuit. Text offers the user a wider range and flexibility of options when programming



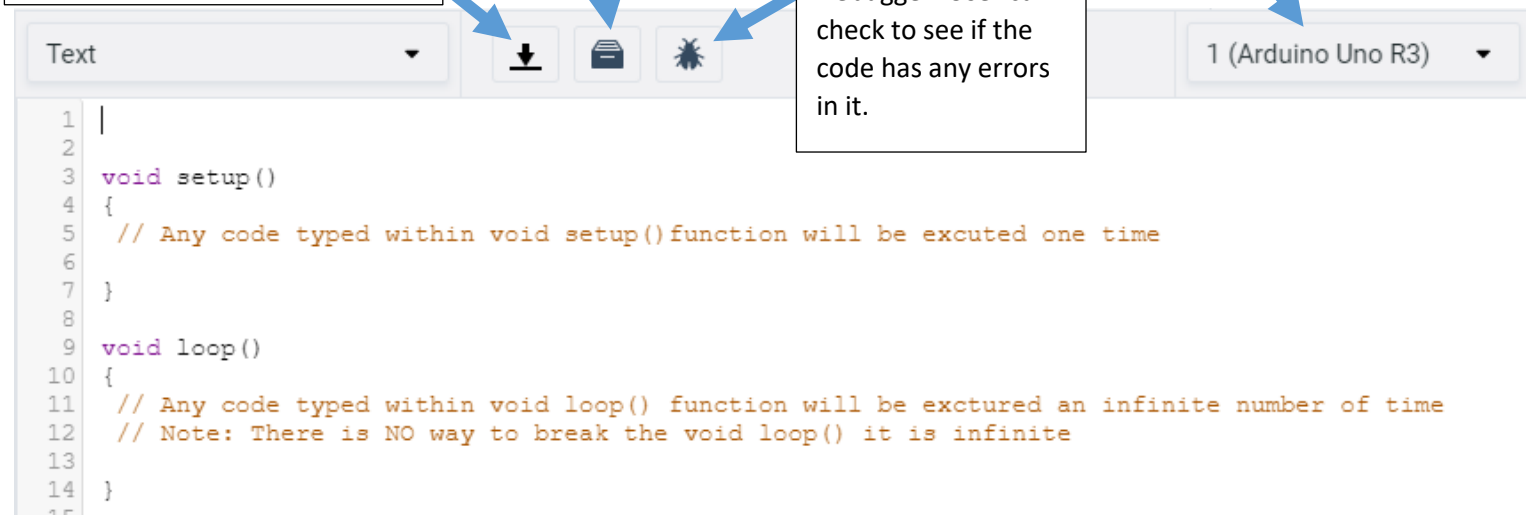
5. Highlight all code and delete
6. Type the following code

Download: Saves the code to be downloaded onto an Arduino Board for live testing.

Libraries: Allows users advanced functions like LCD, motors, Game controllers, etc. without have to program the base code for those components

Debugger: User can check to see if the code has any errors in it.

Choose the type of Arduino that is being used. NOTE: we are using the Arduino Uno in class




```
1
2
3  void setup()
4  {
5   // Any code typed within void setup()function will be excuted one time
6
7  }
8
9  void loop()
10 {
11  // Any code typed within void loop() function will be exctured an infinite number of time
12  // Note: There is NO way to break the void loop() it is infinite
13
14 }
```

7. Program: Blink
   Objective:
   1. Turn the LED ON/OFF
   2. Using Variables in Arduino Code
   3. Control the Number of Times the LED is turned ON/OFF
   4. Output Text using the Serial Monitor

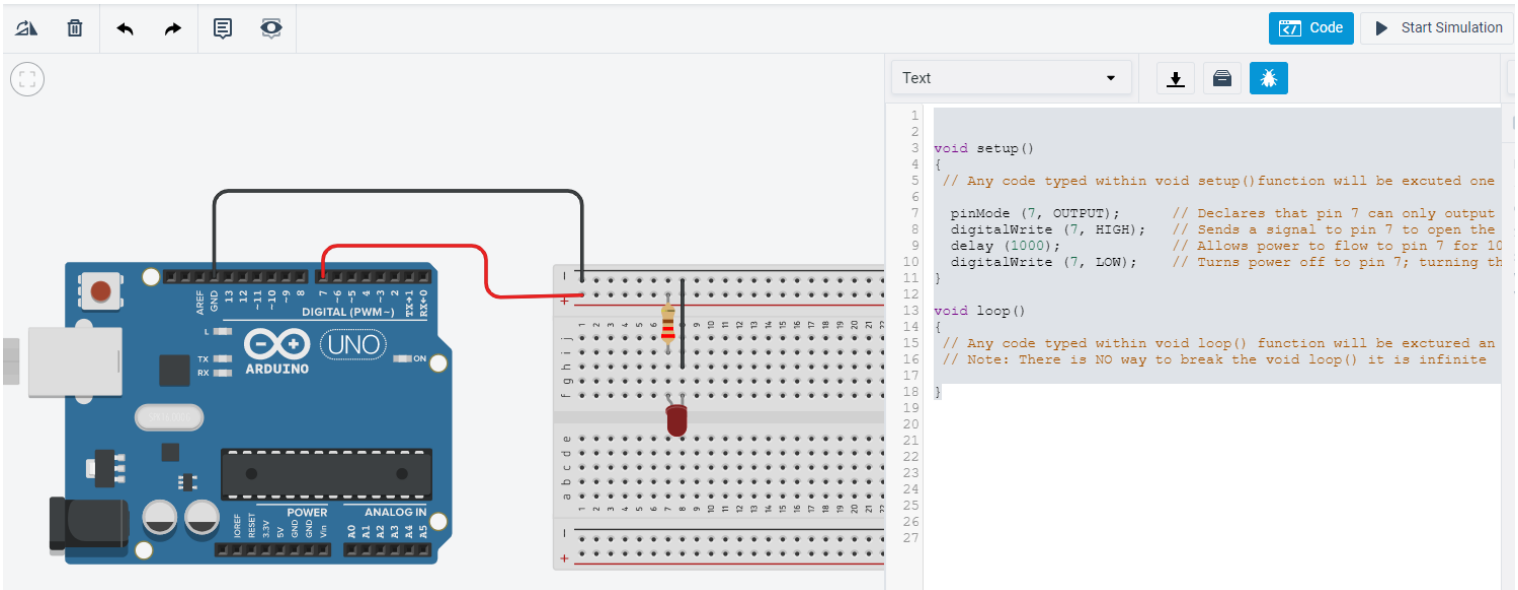
NOTE: whenever the users // before any text; the text that follows on that line becomes a comment, which the software will not read it as a command and skip over it. To comment out large sections at a time use /* at the beginning and */ and the end, this allows the user to comment large portions of code out at a time.

Objective 1: Turn LED ON/OFF

a. Type the following Code (NOTE: do not worry about the color coding. Original code is programmed in the Arduino IDE, which color codes commands. TinkerCAD only codes some of the commands.) > Select Debug to see if there are any errors > Correct errors as needed

```
void setup()
{
 // Any code typed within void setup()function will be excuted one time

  pinMode (7, OUTPUT);     // Declares that pin 7 can only output data (i.e turn light on/off
  digitalWrite (7, HIGH); // Sends a signal to pin 7 to open the flow of electricity to the circuit turning the LED ON
  delay (1000);        // Allows power to flow to pin 7 for 1000 milliseconds (1s)
  digitalWrite (7, LOW);  // Turns power off to pin 7; turning the LED OFF
}

void loop()
{
 // Any code typed within void loop() function will be exctured an infinite number of time
 // Note: There is NO way to break the void loop() it is infinite

}
```

b. Running the Code. Press Start Simulation > Notice the following
   i. Power cord plugs into the Arduino providing electricity to the board
   ii. LED turns on for 1000ms or 1s then turns off.
   iii. Simulation continues to run until user presses Stop Simulation. Theoretically the program is continue to run because the void loop() is unbreakable, but since there is nothing located in this function nothing appears to be happening.



c. Adjust the code as shown below

```
void setup()
{
 // Any code typed within void setup()function will be excuted one time

  pinMode (7, OUTPUT);    // Declares that pin 7 can only output data (i.e turn light on/off

}

void loop()
{
 // Any code typed within void loop() function will be exctured an infinite number of time
 // Note: There is NO way to break the void loop() it is infinite
  digitalWrite (7, HIGH); // Sends a signal to pin 7 to open the flow of electricity to the circuit turning the LED ON
  delay (1000);           // Allows power to flow to pin 7 for 1000 milliseconds (1s)
  digitalWrite (7, LOW);  // Turns power off to pin 7; turning the LED OFF
  delay (2000);           // Maintains power off to pin 7 for 2000ms (2s) before returning to the top of the void loop()
                          // Which turns the LED back ON (HIGH State)

}
```

d. Running the Code. Press Start Simulation > Notice the following
   i. Power cord plugs into the Arduino providing electricity to the board
   ii. LED turns ON for 1000ms or 1s
   iii. LED turns OFF for 2000ms or 2s
   iv. Then void loop() repeats itself

Objective 2: Variables in Arduino Code

In Arduino Code the same variable types are used as C++

- int = integer
- double or float = real number
- char = character
- string = 286 consecutive characters
- among others

One issue we may run into when programming and wiring circuits if we need to move wires for different pins, then in our code we would have to change the pin numbers more than likely missing one. So, what we will do is use a variable name to act as our pin.

There is three four places where variable can be declared

1. before void setup(): this allows the variable to be used throughout the program (void setup(), void loop(), any other functions that maybe used
2. in the void setup(): this allows the variable only to be used within the void setup() function
3. above void loop(): this allows the variable to be used by any functions after the declaration
4. in the void loop(): this allows the variable only to be used within he void loop()

Depending on what you want the variable to do will depend on where it is declared; I prefer to option 1 so I only need to look in one location. Mind you it depends on the program the more complex the program is the more likely variable declartions will be placed throughout the program.

1. Adjust the code as follows.
   Since the pin we are using is the number 7 and the number 7 is an integer we will declare a variable called redled (my led is the color red) as a integer and assign the value of 7 to it > then adjust our code to where ever the number 7 appears to redled

```
int redled = 7;

void setup()
{
 // Any code typed within void setup()function will be excuted one time

  pinMode (redled, OUTPUT);    // Declares that redled can only output data (i.e turn light on/off

}

void loop()
{
 // Any code typed within void loop() function will be exctured an infinite number of time
 // Note: There is NO way to break the void loop() it is infinite
  digitalWrite (redled, HIGH); // Sends a signal to redled to open the flow of electricity to the circuit turning the LED ON
  delay (1000);           // Allows power to flow to redled for 1000 milliseconds (1s)
  digitalWrite (redled, LOW);  // Turns power off to redled; turning the LED OFF
  delay (2000);           // Maintains power off to redled for 2000ms (2s) before returning to the top of the void loop()
                          // Which turns the LED back ON (HIGH State)

}
```

   a. Run the simulation > Notice the program runs the same as previous

Objective 3: Control the Number of Times the LED is turned ON/OFF

Now the code will be adjusted to turn the LED ON/OFF a number of times then give the illusion that the program has ended

1. Add the following code

```
int redled = 7;

void setup()
{
 // Any code typed within void setup()function will be excuted one time

  pinMode (redled, OUTPUT);    // Declares that redled can only output data (i.e turn light on/off

}

void loop()
{
 // Any code typed within void loop() function will be exctured an infinite number of time
 // Note: There is NO way to break the void loop() it is infinite
  for (int x = 1 ; x <= 3; x++)
  {
  digitalWrite (redled, HIGH); // Sends a signal to redled to open the flow of electricity to the circuit turning the LED ON
  delay (1000);           // Allows power to flow to redled for 1000 milliseconds (1s)
  digitalWrite (redled, LOW);  // Turns power off to redled; turning the LED OFF
  delay (2000);           // Maintains power off to redled for 2000ms (2s) before returning to the top of the void loop()
                          // Which turns the LED back ON (HIGH State)
  }
```

2. Run the Simulation
    a. Notice the code runs and the LED is always flashing. The reason for this is the FOR loop is running 3 times turning the LED ON/OFF then resets itself after it falls out of the loop the 3rd time. So, we still have an infinite loop.
    b. Test to show the FOR loop ends, but the void loop () does not.

    Add a delay statement after the FOR Loop of 3000 ms (3s) > Notice know that the light flashes fast then there is a long delay to reset itself.

```
int redled = 7;

void setup()
{
 // Any code typed within void setup()function will be excuted one time

  pinMode (redled, OUTPUT);    // Declares that redled can only output data (i.e turn light on/off

}

void loop()
{
 // Any code typed within void loop() function will be exctured an infinite number of time
 // Note: There is NO way to break the void loop() it is infinite
  for (int x = 1 ; x <= 3; x++)
  {
  digitalWrite (redled, HIGH);  // Sends a signal to redled to open the flow of electricity to the circuit turning the LED ON
  delay (1000);                 // Allows power to flow to redled for 1000 milliseconds (1s)
  digitalWrite (redled, LOW);   // Turns power off to redled; turning the LED OFF
  delay (2000);                 // Maintains power off to redled for 2000ms (2s) before returning to the top of the void loop()
                                // Which turns the LED back ON (HIGH State)
  }
delay (3000);                   // delay statement shows the FOR loop exits then resets itself
}
```

Now we will trap the user within a loop that does not do anything to give them the illusion that the program has ended by using a while loop.

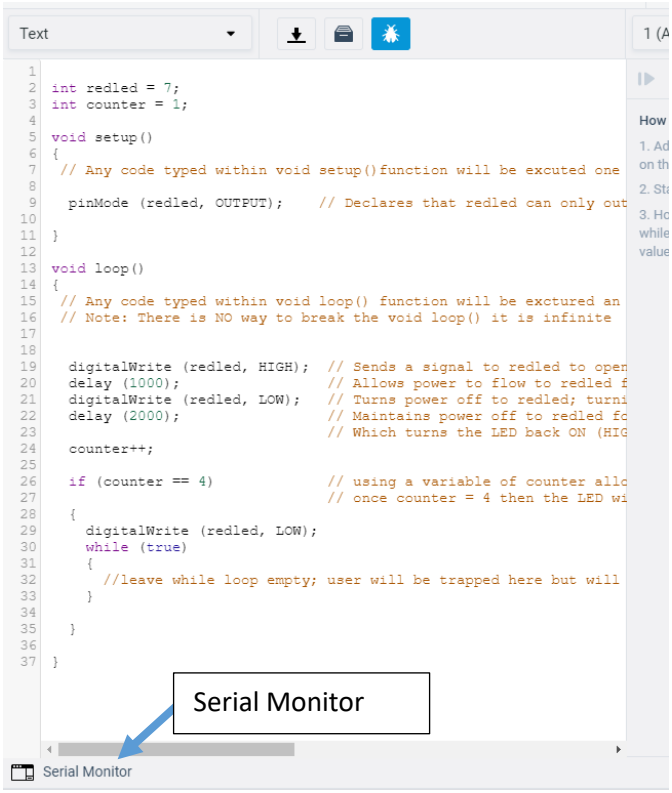
    c. Modify the code as follows

```
int redled = 7;
int counter = 1;

void setup()
{
 // Any code typed within void setup()function will be excuted one time

  pinMode (redled, OUTPUT);    // Declares that redled can only output data (i.e turn light on/off

}

void loop()
{
 // Any code typed within void loop() function will be exctured an infinite number of time
 // Note: There is NO way to break the void loop() it is infinite


  digitalWrite (redled, HIGH);  // Sends a signal to redled to open the flow of electricity to the circuit turning the LED ON
  delay (1000);                 // Allows power to flow to redled for 1000 milliseconds (1s)
  digitalWrite (redled, LOW);   // Turns power off to redled; turning the LED OFF
  delay (2000);                 // Maintains power off to redled for 2000ms (2s) before returning to the top of the void loop()
                                // Which turns the LED back ON (HIGH State)
  counter++;

  if (counter == 4)             // using a variable of counter allows the program to check each time through the loop to see how many times throught the loop
                                // once counter = 4 then the LED will stop flashing and the nested while loop will trap the user
  {
    digitalWrite (redled, LOW);
    while (true)
    {
      //leave while loop empty; user will be trapped here but will not know it
    }

  }

}
```

d. Run the Simulation > Notice
    i. Light Flashes ON/OFF 3 times checking each time with the IF/THEN if counter equals 4
    ii. Once counter equals 4 the light turns off

Objective 4: Adding Text Output

Adding text to a program to output/input information Arduino uses the Serial Monitor. The Serial Monitor is located on the Bottom Left of the Programming Screen



This is where Arduino programming varies from C++ programming when dealing with Input/Output.

1. Connecting to the Serial Monitor and outputting to the Serial Monitor
    a. In void setup() place the following line of code
        serial.Begin (9600); This code sets the baud rate in which information is sent between the PC and the Arduino board. 9600 is the goldilocks number to transfer English data charaters.

```
int redled = 7;
int counter = 1;

void setup()
{
 // Any code typed within void setup()function will be excuted one time

  pinMode (redled, OUTPUT);     // Declares that redled can only output data (i.e turn light on/off
  Serial.begin (9600);          // Sets the baud rate; speed data is transferd between PC and Arduino Board
}
```

b. In void loop() place the following code

Two Options to print items to the Serial Monitor

Option 1: Serial.print (“TYPE INFO TO OUTPUTTED); or Serial.print (Place Variable Name); Both options will keep the cursor on the same line when outputting

```
13  void loop()
14  {
15   // Any code typed within void loop() function will be exctured a
16   // Note: There is NO way to break the void loop() it is infinite
17
18
19    digitalWrite (redled, HIGH);  // Sends a signal to redled to op
20    delay (1000);                 // Allows power to flow to redled
21    digitalWrite (redled, LOW);   // Turns power off to redled; tur
22    delay (2000);                 // Maintains power off to redled
23                                  // Which turns the LED back ON (H
24    Serial.print ("HELLO WORLD");
25    counter++;
26
27    if (counter == 4)             // using a variable of counter al
28                                  // once counter = 4 then the LED
29    {
30      digitalWrite (redled, LOW);
31      while (true)
32
```

Insert Output Line Here

Serial Monitor

```
HELLO WORLDHELLO WORLDHELLO WORLD
```

Text stays on same line

Option 2: Serial.println (“TYPE INFO TO OUTPUTTED); or Serial.println (Place Variable Name); Both options will drop the cursor down to the next line

```
13  void loop()
14  {
15   // Any code typed within void loop() function will be exctured an infinite number
16   // Note: There is NO way to break the void loop() it is infinite
17
18
19    digitalWrite (redled, HIGH);          // Sends a signal to redled to open the f
20    delay (1000);                         // Allows power to flow to redled for 100
21    digitalWrite (redled, LOW);           // Turns power off to redled; turning the
22    delay (2000);                         // Maintains power off to redled for 2000
23                                          // Which turns the LED back ON (HIGH Stat
24    Serial.println ("HELLO WORLD");
25    counter++;
26
27    if (counter == 4)                     // using a variable of counter allows the
28                                          // once counter = 4 then the LED will sto
29    {
30      digitalWrite (redled, LOW);
31      while (true)
32      {
33        //leave while loop empty; user will be trapped here but will not know it
34      }
35
```

Insert Output Line Here

Serial Monitor

```
HELLO WORLD
HELLO WORLD
HELLO WORLD
```

Text stays on drops to the next line

NOTE: There is not a way in Arduino to mix User Text (“HELLO WORLD “) with variables. Each has to be on its own Serial.print or Serial.println line

NOTE: TinkerCAD Serial Monitor does not clear the previous run. Refresh the Browser and it will clear the last test run

c. Type the following code where the arrows are located

```
int redled = 7;
int counter = 1;

void setup()
{
  pinMode (redled, OUTPUT);            // Declares that redled can only output data (i.e turn light on/off
  Serial.begin (9600);                 // Sets the baud rate; speed data is transferd between PC and Arduino Board
  Serial.println ("Hello World");     // Only prints 1 time because of location in void setup()
}

void loop()
{
  digitalWrite (redled, HIGH);  // Sends a signal to redled to open the flow of electricity to the circuit turning the LED ON
  delay (1000);                 // Allows power to flow to redled for 1000 milliseconds (1s)
  digitalWrite (redled, LOW);   // Turns power off to redled; turning the LED OFF
  delay (2000);                 // Maintains power off to redled for 2000ms (2s) before returning to the top of the void loop()
                                // Which turns the LED back ON (HIGH State)
  Serial.print ("Counter = ");  // Serial.print is used so that that value of counter appears next to the = sign which is on the proceeding line
  Serial.println (counter);     // Prints the value of counter next to previous line, then drops the cursor to the next line
  Serial.println ();            // Leaves a blank space between outputs, creates a double space effect
  counter++;

  if (counter == 4)             // using a variable of counter allows the program to check each time through the loop to see how many times throught the loop
                                // once counter = 4 then the LED will stop flashing and the nested while loop will trap the user
  {
    digitalWrite (redled, LOW);
    Serial.println ("GOOD BYE");
    while (true)
    {
      //leave while loop empty; user will be trapped here but will not know it
    }
  }
}
```

d. Run the Simulation Output Should look as follows

```
Hello World
Counter = 1

Counter = 2

Counter = 3
```

Submission

1. Take a Screenshot of the following
   a. Complete Code
   b. Final Output
   c. Preview screen of program from User Hub (similar to Tutorials 1 and 2 Electrical Circuits series/parallel)

   Place each screenshot in a Word or Google Document and send me a pdf of completed work.