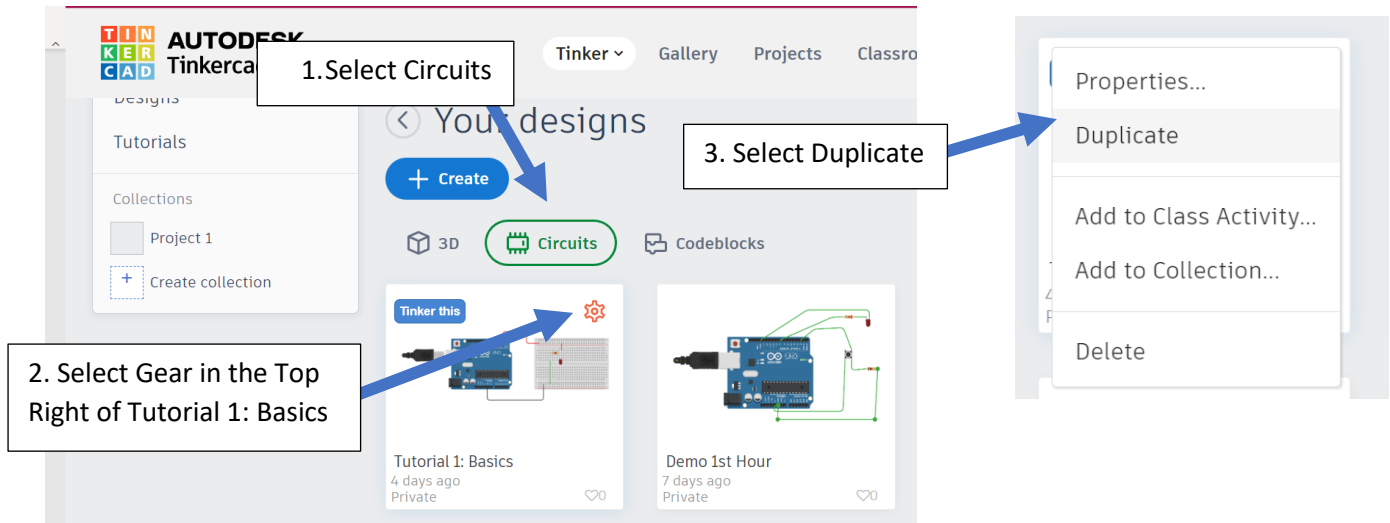


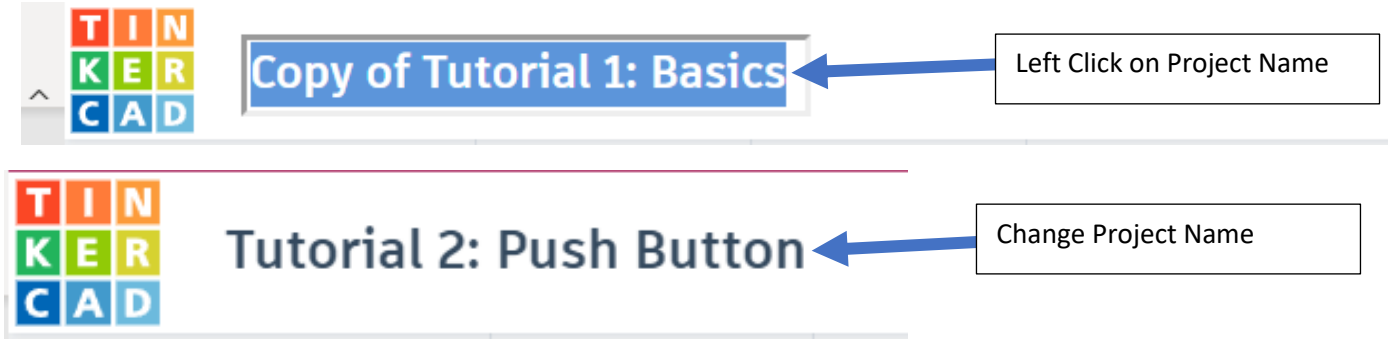
TinkerCAD Blockly Programming and Arduino

Program: Push Button

1. Navigate to TinkerCAD.com > Click Sign-In Icon (Top Left Side of Screen > Select Students, Join your Classroom > Classroom CODE: See Teacher for Code > Nickname: Student First Name (all lower case)
2. Select Circuits > Place Cursor on Tutorial 1: Basic File > Select Gear (Settings) Icon in the Top Right Corner > Select Duplicate



3. Copy of Tutorial 1 will open > Select Project Name at the Top > Change the Name to Tutorial 2: Push Button



Part 1: Wire Push Button

4. Add the following Wires, Push Button and Resistor

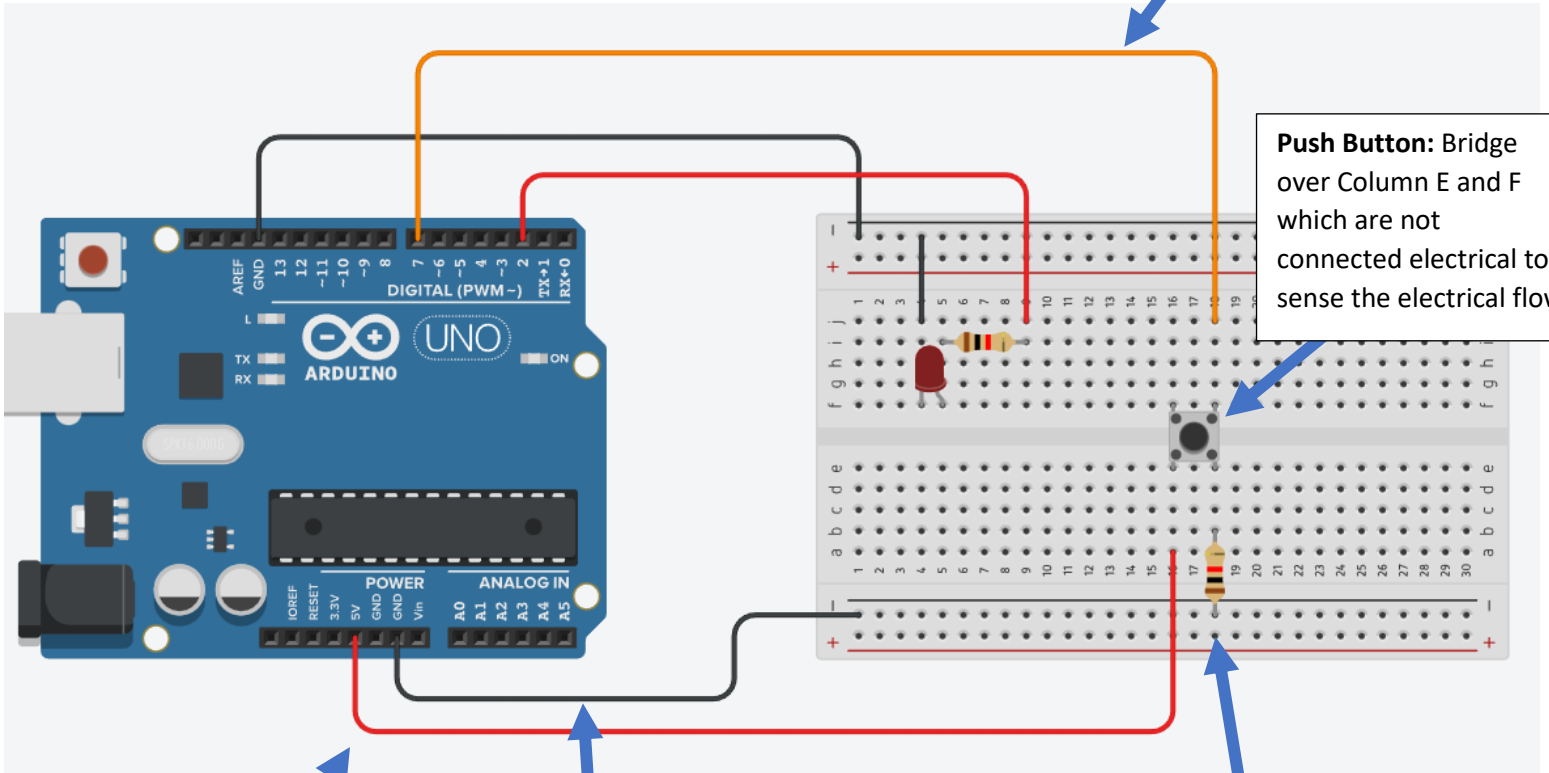
NOTE: a 1K Ohm Resistor is required to use up all of the Voltage (Electrical Potential) without the resistor the Arduino Board will crash

Signal Wire: Senses electrical flow

Not Pressed: Low Electricity Flow thru the push button

Pressed: High Electricity Flow thru the push button

Push Button: Bridge over Column E and F which are not connected electrical to sense the electrical flow



5V Wire: Passes voltage to the Push Button.

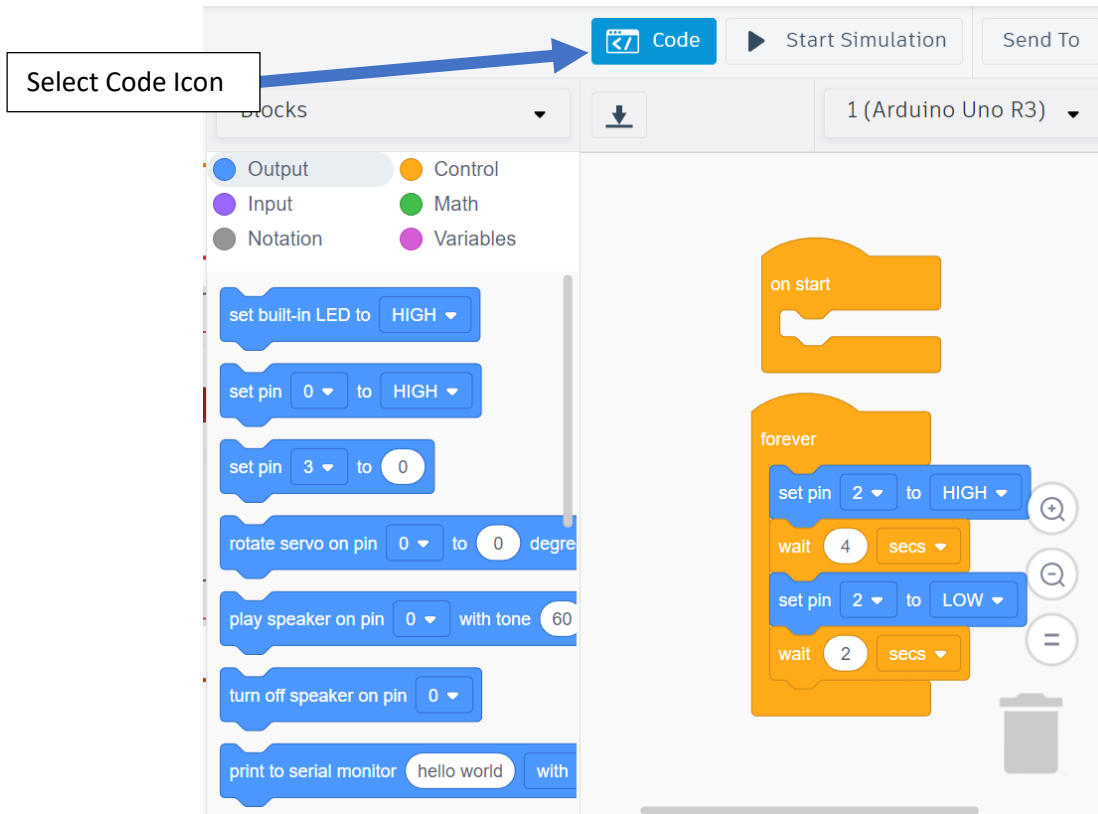
GND (Ground) Wire: Completes the Circuit

1K Ohm Resistor: Reduces the amount of voltage passed to it from the circuit to a manageable level on the Arduino.

STOP

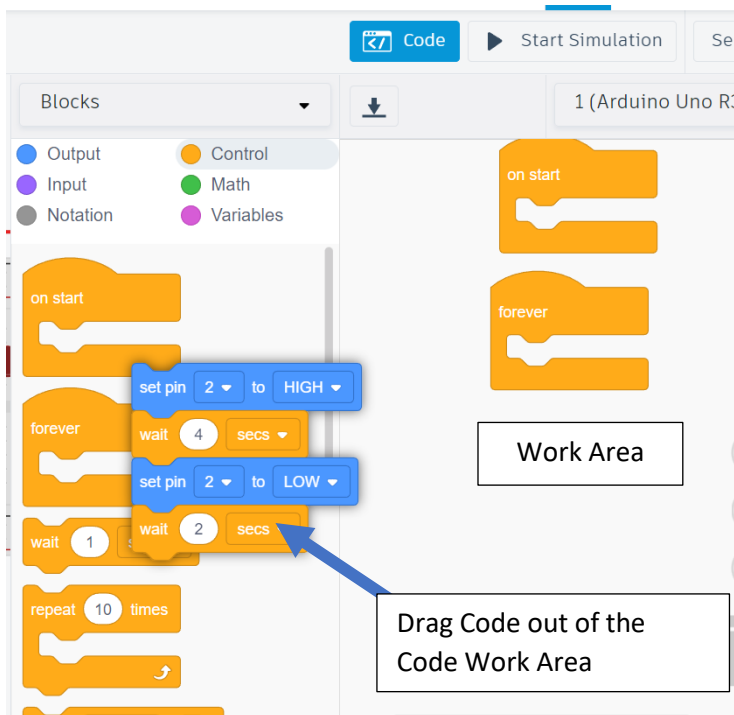
Part 2: Coding Push Button

5. Select Code Icon



6. Remove Wait Statements from the Code Except for On Start and Forever Function

- Hold Left Mouse Button on the Wait Statement > Drag out of the Coding Screen > Let Go of Left Mouse Button



NOTE: Code may have to be moved around to pull individual Code Blocks pieces apart

b. Drag and Drop the Following Code.

NOTE: Code Blocks are Color Coded into groupings

IF Statement allows for code to run when the If Condition is True; Else will run any code when the If Condition is False

The image shows a Scratch code editor with the following blocks:

- on start** block.
- forever** loop containing:
 - if** block: `read digital pin 7 = 1` then:
 - set pin 2 to HIGH** block.
 - else** block:
 - set pin 2 to LOW** block.

Annotations:

- A box labeled "Signal: Arduino Pin 7" points to the pin number in the if condition.
- A box labeled "Digital Ports have Two States" with sub-points "0: Low/ Push Button Not Pressed" and "1: High/ Push Button Pressed" points to the value 1 in the if condition.

c. Test the Code

i. Press Start Simulation Button > Left Click on the Push Button

The screenshot shows the TinkerCAD interface for "Tutorial 2: Push Button".

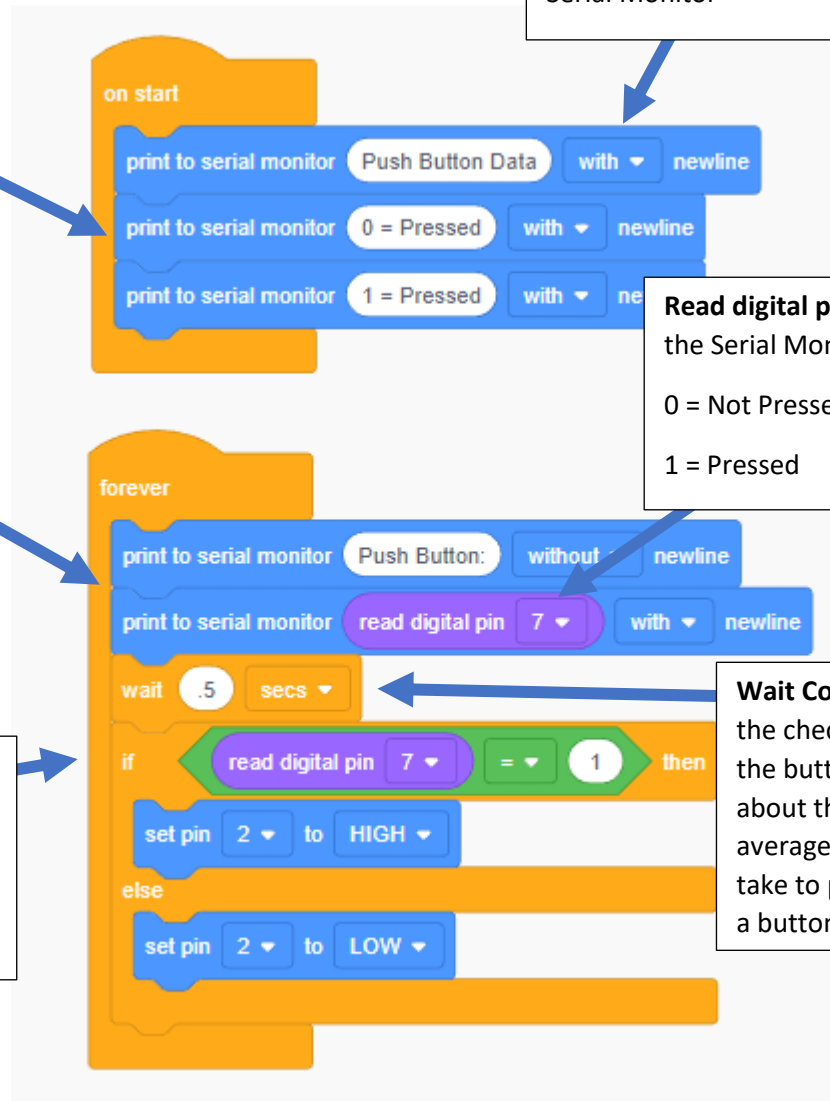
- Hardware:** An Arduino Uno is connected to a breadboard with a push button. The button's ground is connected to the Arduino's GND, and its signal pin is connected to digital pin 7.
- Code Editor:** The code blocks are identical to the previous image, but the `read digital pin` block is set to pin 0. A callout box "Left Mouse Button on the Push Button" points to the button in the simulation.
- Simulation Controls:** The "Start Simulation" button is highlighted with a callout box "Press Start Simulation".
- UI Elements:** The top right shows "All changes saved" and a user profile icon. The bottom right has search and zoom controls.

d. When finished press Stop Simulation

Part 4: Reporting Push Button Status on Screen

7. Place the following Code

Print to Serial Monitor: Prints Message in Serial Monitor 1 Time only



With/Without Newline: Allows the user to drop text down to a new line or stay on the current line in the Serial Monitor

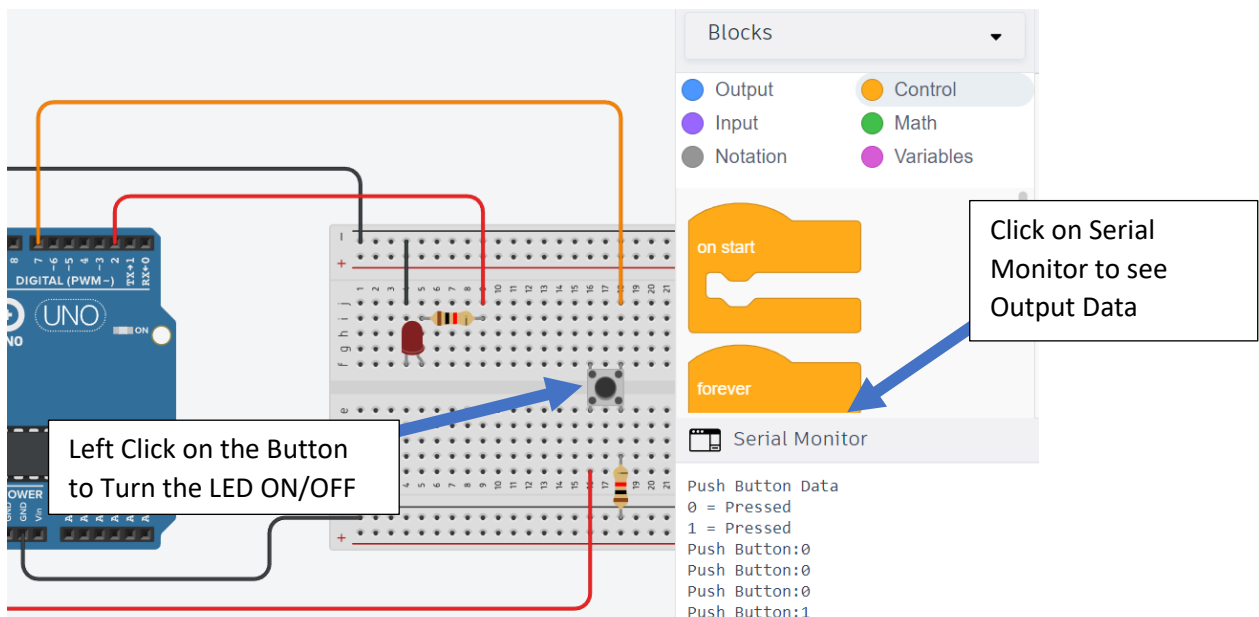
Read digital pin 7: Prints out in the Serial Monitor
0 = Not Pressed
1 = Pressed

Print to Serial Monitor: Prints Message in Serial Monitor Infinitely. Data will Change from 0 to 1 when the Pushbutton is pressed or unpressed

Wait Command: Slows the checking speed of the button. ½ second is about the time an average person would take to press and release a button

If/Else Statement: Checks to see if the Button as been Pressed
If True (Pressed) > LED Turns On
Else (Unpressed/False) > LED Turns Off

8. Run Simulation > Open Serial Monitor at the bottom of the Code Area to See Data Output > Place Cursor on the Push Button + Left Click to simulate a button press > When Finished Select Stop Simulation

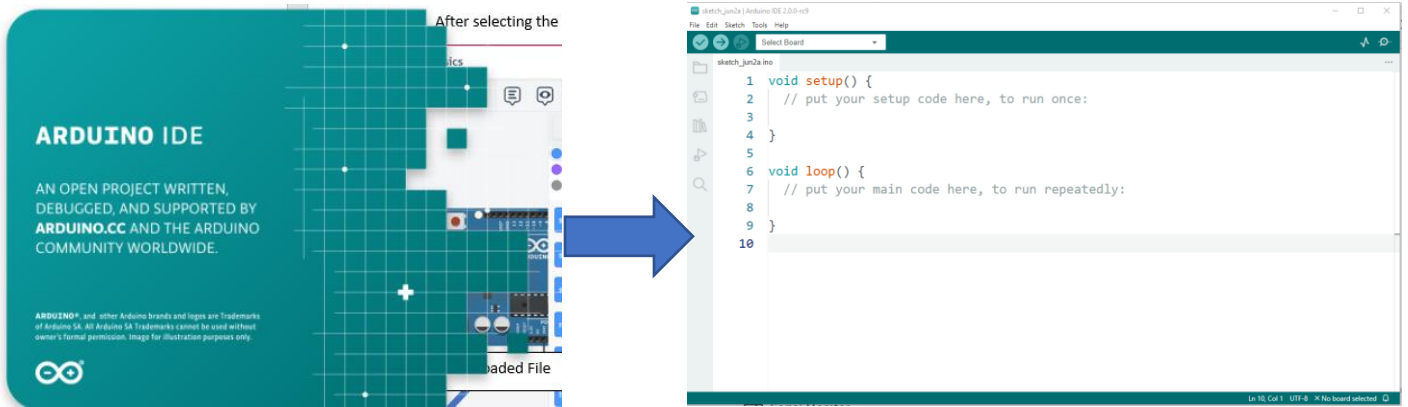
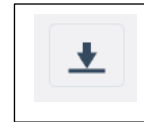


Click on Serial Monitor to see Output Data

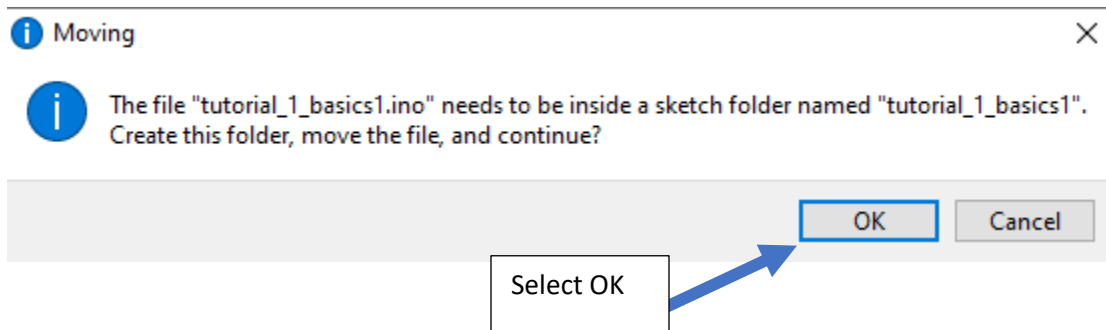
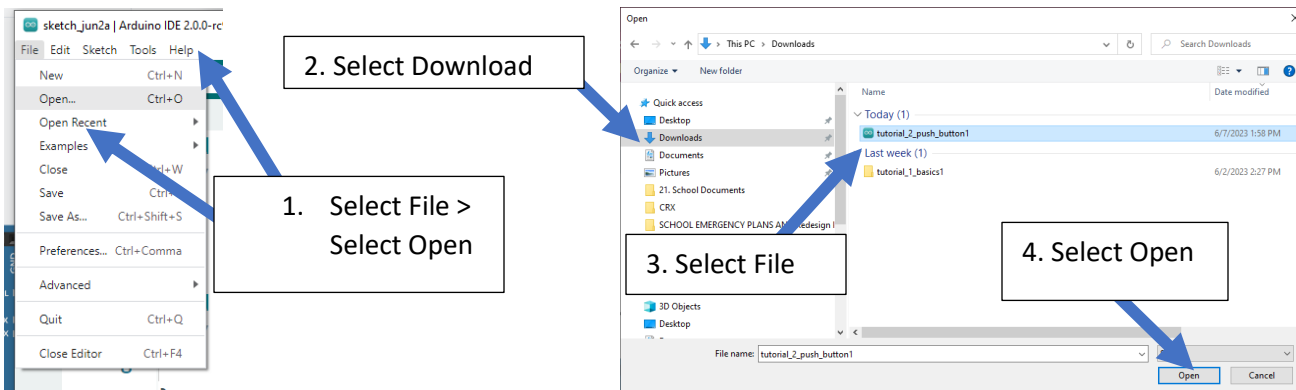
Left Click on the Button to Turn the LED ON/OFF

Part 3: Download and Build Circuit

- 9. Select Download Button in the Top Left Corner of the Programming Screen
- 10. Open Arduino Software



- 11. Select File > Open > Navigate to Downloads > Select File > Pop Menu will Appear to ask to create a folder for the file > Select YES > Arduino Software will Open



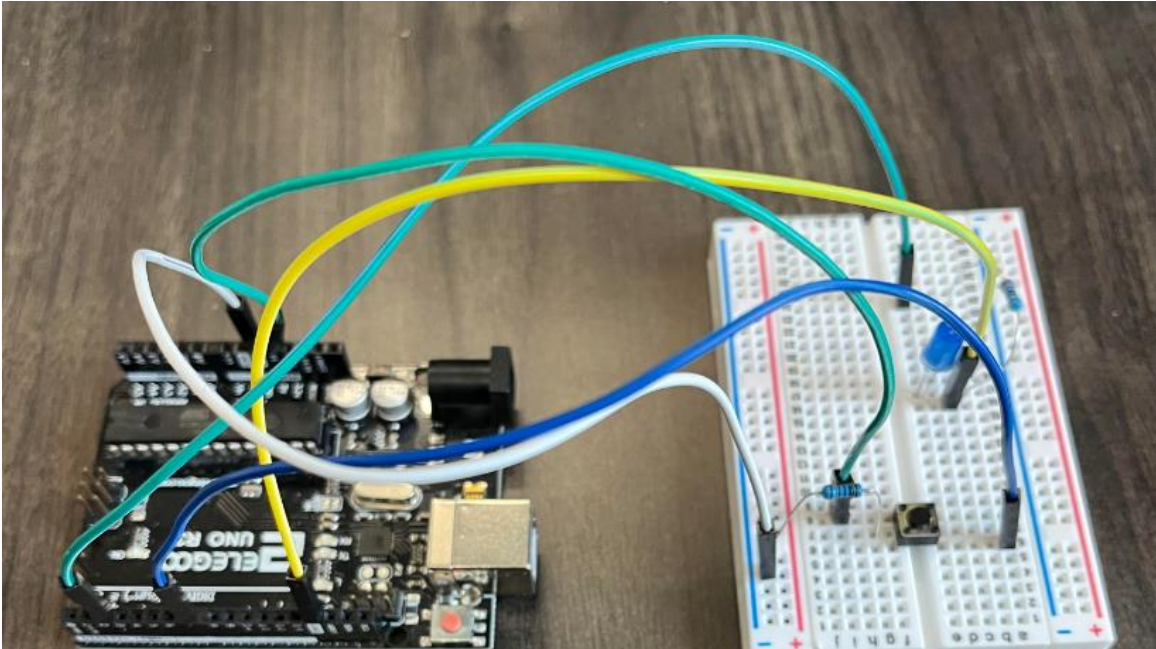
Program will appear as Structured Text

DO NOT ADJUST any of the code provided

NOTE: Comments were added (Gray Text)

```
1 // C++ code LED and Push Button
2 //
3 int LED = 0;
4
5 void setup()
6 {
7   Serial.begin(9600);           //Serial.begin (9600); connects the Serial Monitor to the Program
8   pinMode(7, INPUT);           //pinMode Declares Arduino Pin 7 can only take data signals in
9   pinMode(2, OUTPUT);          //pinMode Declares Arduino Pin 2 can only send data signals out
10
11  Serial.println("Push Button Data"); //Serial.println: \n is for the newline in the Serial Monitor
12  Serial.println("0 = Pressed");
13  Serial.println("1 = Pressed");
14 }
15
16 void loop()
17 {
18   Serial.print("Push Button:");
19   Serial.println(digitalRead(7));
20   delay(500);                  // Wait for 500 millisecond(s) 1/2 second
21   if (digitalRead(7) == 1) {   //digitalRead is monitoring the electrical signal in Arduino Pin 7
22     digitalWrite(2, HIGH);     //HIGH = ON
23   } else {
24     digitalWrite(2, LOW);      //LOW = OFF
25   }
26 }
```

12. Wire the Circuit. Circuit maybe setup exactly like TinkerCADs

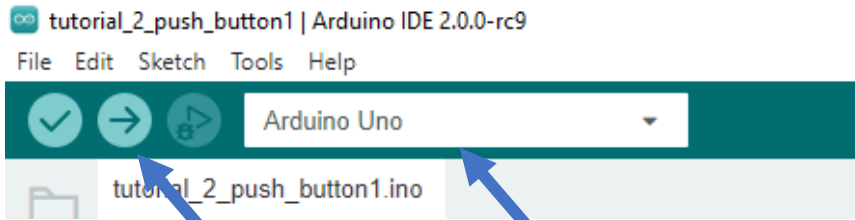


STOP

**Have a Teacher or Volunteer
inspect wiring setup**

13. Connect Arduino to the PC with the USB Cord

14. Select Arduino COM Port > Download Program > Open Serial Monitor Top Right Corner > Press the Push Button to see the LED Turn ON/OFF



Serial Monitor will open at the bottom of the Arduino Screen

