Name: Answer Key Date: ____________ Class: ____________

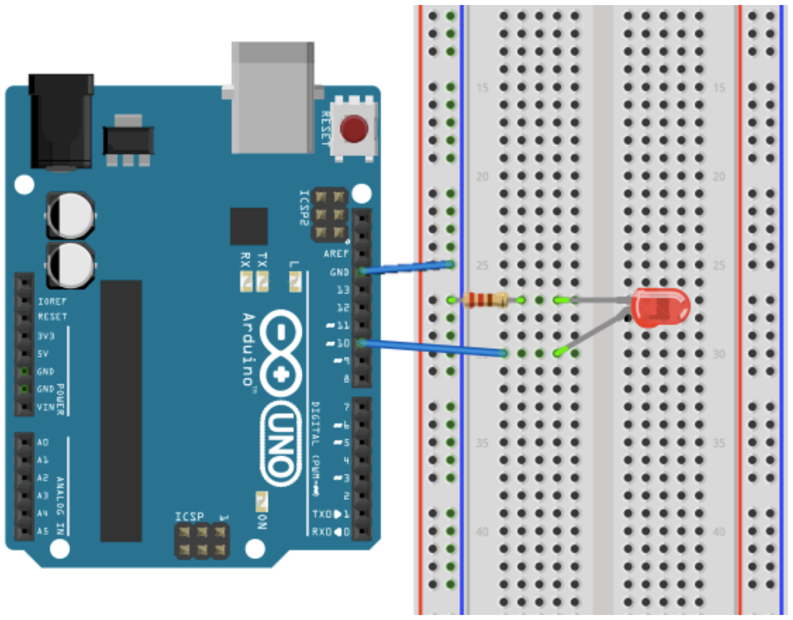
Activity 2: Making an LED Blink Worksheet

Overview

In this activity, we will connect an (external) LED to an Arduino using jumper wires and a breadboard, and then program the Arduino to cause the LED to blink.

Construct the Hardware

1. Gather the following materials:
   - 1 Arduino and USB cable
   - 1 breadboard and 2 jumper wires
   - 1 LED and 1 220Ω resistor
2. Construct the following setup:



   A. Connect 1 jumper wire from GND ("ground") to one of the (-) columns.
   B. Place 1 leg of the resistor in the same column as the GND wire. Place the other leg of the resistor in a new row.
   C. Place the short leg of the LED to the same row as the resistor leg. Place the long leg in a new row.
   D. Connect 1 jumper wire from the previous row to digital pin 10.
   E. Check that your setup matches the setup in the diagram above or the one at the front of the room.

Create the Program

3. Type up the associated program for Activity 2 (see the next page).

   Tip: Capitalization and spelling matter. Note: most lines of the code end in a semi-colon.

4. Save the file as **LED_Blink**.

5. To check that the Arduino can compile your code, "Verify" it. Do this by either pressing **Command + R** or by clicking on the Verify button (it looks like a check mark).

Note: This does not upload the program to the Arduino board. This operation only checks to make sure that the Arduino will be able to execute it.

Note: If the file does compile without errors, this does not necessarily mean that there are no issues with your code.

```
/*
 Activity 2: LED_Blink
  Turns on an external LED on for one second, then off for one second, repeatedly.
*/

// Identify the digital pin to which the LED is connected:
int led = 10;

// The setup() routine runs only once:
void setup() {

        // Set the digital pin as an output.
        pinMode(led, OUTPUT);

}

// The loop routine runs over and over again forever:
void loop() {

        digitalWrite(led, HIGH);
        delay(1000);

        digitalWrite(led, LOW);
        delay(1000);

}
```

Troubleshoot Errors

6. Error messages pop up in orange text the bottom of the IDE window. The computer tells you where (what lines) it believes the errors in the program are located (though it is not always right).
7. Start by reading the first error message and checking that line of code. Compare what you have written to the original program. Common errors:
   - If you get the message "_________ was not declared in this scope," it is possible you misspelled one of the variables or functions.
   - If you get the message "expected ';' before ____" —that means you forgot to add a semi-colon at the end of the previous line.

Run the Program

8. Using the USB cable, connect your Arduino to your computer. Expect to see some lights flash on the Arduino board. Then the Arduino starts to execute the last program that was uploaded to it.
9. From the "Tools" menu in the Arduino IDE, make sure the appropriate serial port is selected.
10. Now upload the "LED_Blink" sketch. To do this, hit **Command + U.** (Or, click on the Upload Button in the Arduino IDE, which looks like an arrow pointing to the right.)
11. After a few seconds, expect the LED to start to blink! If it is not blinking, check your hardware first, and then your code.

Name: ____ **Date:** ____________________ **Class:** ________________

Reflection Questions

1. What do you think the function **digitalWrite( )** does? What information must go in its parentheses for it to work?

   This function is used to turn a digital I/O pin on (HIGH) or off (LOW). In order for it to work, this function requires two arguments.

   digitalWrite(arg1, arg2)

   The first argument is the number of the digital pin you're looking to control, and the second is the state to which you want to set that pin (either HIGH or LOW).

2. What do you think the function **delay( )** does? What does the number inside its parentheses represent?

   The delay function tells the Arduino to wait for a certain amount of time before moving onto and executing the next line of code. The number inside the parentheses is the number of milliseconds that the Arduino will wait before continuing on.

Alternatives

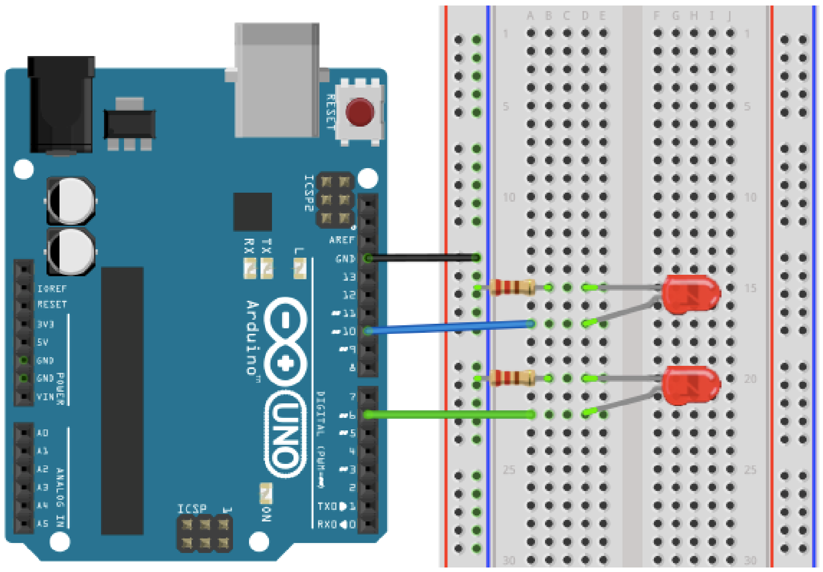
1. Figure out how to change the blink-rate of the LED. For example, try to get the LED to blink twice as fast, twice as slow, or with a completely new pattern. Describe what you changed in the program to do this.

   An easy way to change the blink rate of the LED is to change the numbers in the delay functions. In order to get the LED to blink twice as fast, change the number in each delay to 500. In order to make the LED blink twice as slow, change each delay to 2000.

2. Figure out how to add a second LED and get it to blink opposite the first LED (when the first one is on, the second one should be off). Sketch the new setup below and describe the changes you made to the program to accomplish this.

   Add an additional LED to the original setup as follows. Place the two legs of the new LED into two new rows. Connect the shorter leg to a resistor that on the other end connects to the GND column. Using a wire, connect the long leg of the LED to another digital pin, such as pin 6. Then modify the code as follows (the differences are in bold).

Name: ____ **Date:** ____________________ **Class:** ________________



```
/*
Activity 2B: LED_Blink_2
Turns on an external LED on for one second, then off for one second, repeatedly.  At the same time, a
second LED is turned off and on opposite the first LED.
*/

// Identify the digital pin to which the LED is connected:
int led = 10;
int led2 = 6;

// The setup() routine runs only once:
void setup() {

        // Set the digital pin as an output.
        pinMode(led, OUTPUT);
        pinMode(led2, OUTPUT);

}

// The loop routine runs over and over again forever:
void loop() {

        digitalWrite(led, HIGH);
        digitalWrite(led2, LOW);
        delay(1000);

        digitalWrite(led, LOW);
        digitalWrite(led2, HIGH);
        delay(1000);

}
```