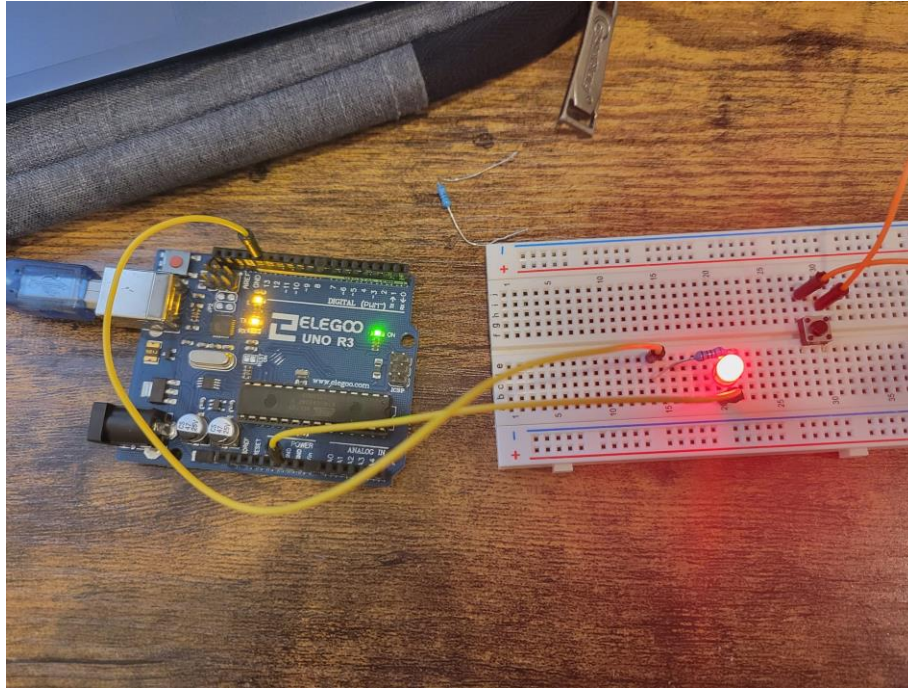


Exercise 1: Make the LED blink faster at a rate of 2 Hz. Include the Code used and explain what was changed.



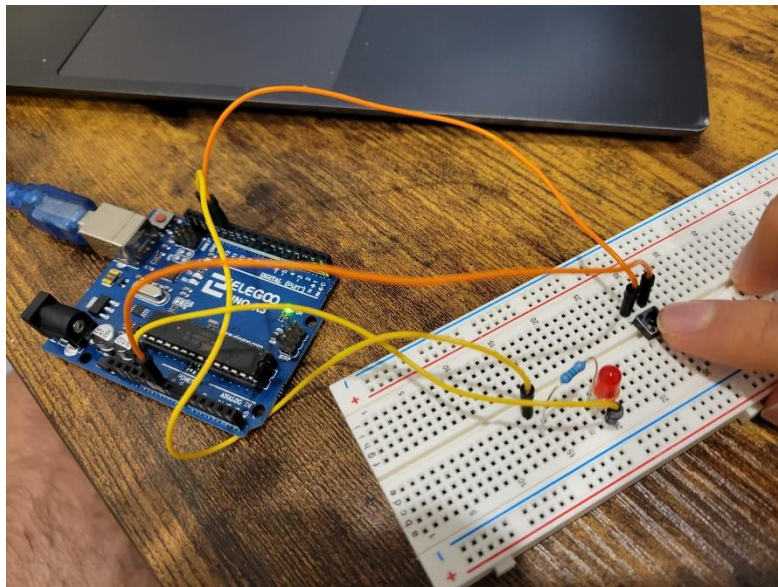
lab4_ex_1_1

```
1
2 int ledPin = 13;
3
4 void setup() {
5   pinMode(ledPin, OUTPUT);
6 }
7 void loop() {
8
9   digitalWrite(ledPin, HIGH);
10  delay(250);
11
12  digitalWrite(ledPin, LOW);
13  delay(250);
14
15  digitalWrite(ledPin, HIGH);
16  delay(250);
17
18  digitalWrite(ledPin, LOW);
19  delay(250);
20 }
```

2 Hz means 2 blinks per second. One blink consists of ledPin HIGH – delay – ledPin LOW - delay. In the code, I follow this algorithm. I change the delay between the led blinks and reduce it to 250ms as well as added an extra pair of digitalWrite statements to make the delay sum up to 1 second

Exercise 2: Print “Hello World!” 10 times on the serial monitor and then “Echo” continuously using Serial.println(). Include the Code used and explain what construct you used. Hint: It is important to set up the Serial communication using Serial.begin(baud_rate) function in the setup. Set the baud rate to 9600. You should also set the same baud rate in the serial monitor window once you run the program by selecting the baud rate from a dropdown list at the bottom right-hand corner of the window.

```
lab4_ex_1_2
1 void setup() {
2     Serial.begin(9600);
3
4     for (int i=0; i < 10; i++){
5         Serial.println("Hello world!");
6     }
7 }
8
9 void loop() {
10    Serial.println("Echo");
11    delay(100);
12 }
```

[illegible]

As mentioned in the hint, we need to set the baud rate to 9600 to access the Serial monitor.

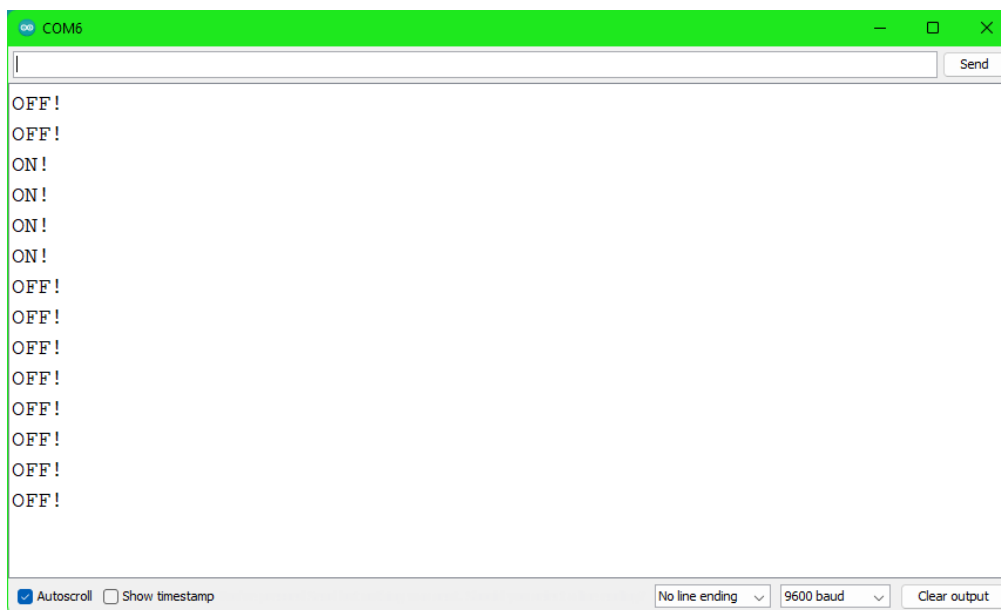
We want “Hello world!” to get printed out 10 times and once and prioritized on init so we must use a for loop inside the setup function to have it run once.

For the rest of the times we print “Echo” which can be printed in the loop with some added delay to make it visible in the Serial monitor.

Exercise 3: Using a wire as a switch, connect pin D12 to GND to control when the serial print monitor shows the text (“ON!”) or (“OFF!”). Explain your observations and describe

your Code. Hint: utilize INPUT_PULLUP for pin 12 to ensure that no floating-point condition exists. <https://www.arduino.cc/en/Tutorial/DigitalInputPullup>

```
lab4_ex_1_3
1 void setup() {
2   Serial.begin(9600);
3   pinMode(12, INPUT_PULLUP);
4 }
5
6 void loop() {
7
8   int sensorVal = digitalRead(12);
9
10  if (sensorVal == 0) {
11    Serial.println("ON!");
12  } else if (sensorVal == 1) {
13    Serial.println("OFF!");
14  }
15  |
16  delay(1000);
17 }
```



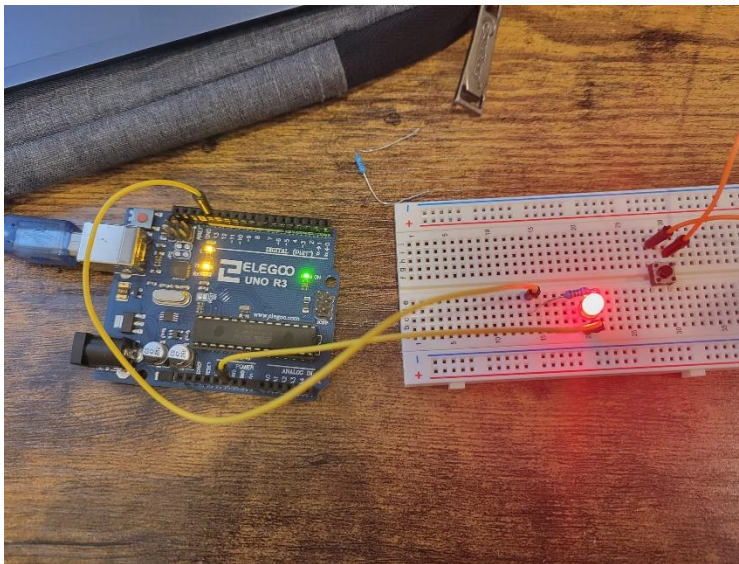
As mentioned in the hint, we need to set the baud rate to 9600 to access the Serial monitor.

INPUT_PULLUP monitors the state of a switch by establishing serial communication between your Arduino and your computer over USB.

When I press the button (and hold) shown in the above circuit, the sensorVal is read as 0 and we print "ON!" in the serial monitor. When I leave the button, the sensorVal is read as 1 and the serial monitor prints "OFF!" We delay by 1 second to see the output clearly.

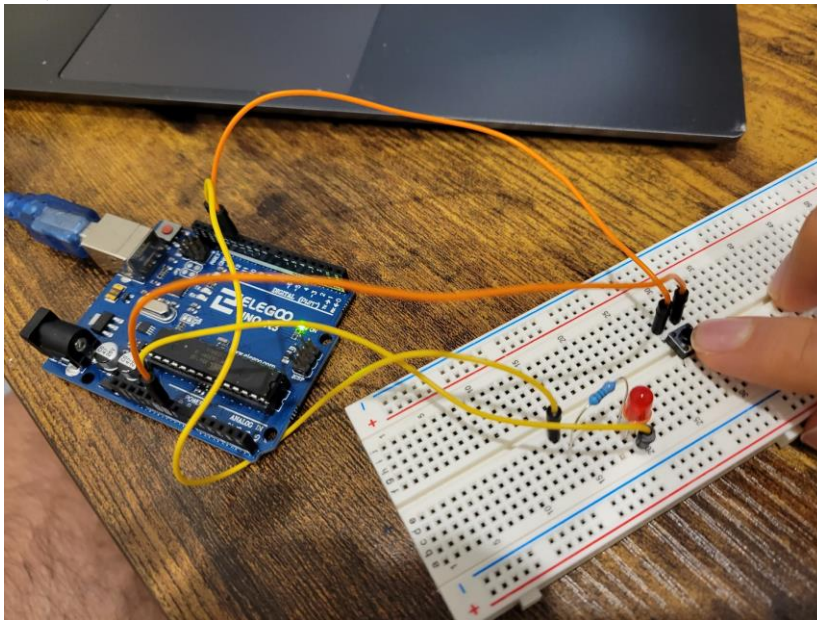
Exercise 4: Make the LED blink 5 times at the rate of 0.5 blinks/sec and then 10 times at 1 blink/sec. Also, display the blink rate on the Serial monitor at the beginning of every loop. Include and explain the code.

Hint - Use 2 for loops for blinking the LED's at different rates.



lab4_ex_1_4

```
1
2 int ledPin = 13;
3 void setup() {
4   pinMode(ledPin, OUTPUT);
5   Serial.begin(9600)
6 }
7
8 void loop() {
9   for (int i = 0; i < 5; i++) {
10    Serial.println("Blinking at 0.5 times per second");
11    digitalWrite(ledPin, HIGH);
12    delay(1000);
13    digitalWrite(ledPin, LOW);
14    delay(1000);
15  }
16
17  for (int i = 0; i < 10; i++) {
18    Serial.println("Blinking at 1 times per second");
19    digitalWrite(ledPin, HIGH);
20    delay(500);
21    digitalWrite(ledPin, LOW);
22    delay(500);
23  }
24 }
```



As mentioned in the hint, we need to set the baud rate to 9600 to access the Serial monitor.

In order to blink 5 times at the rate of 0.5 times per second, we can run a for loop with delay between high and low of 1 second which counts as one blink. Essentially we blink once every 2 seconds.

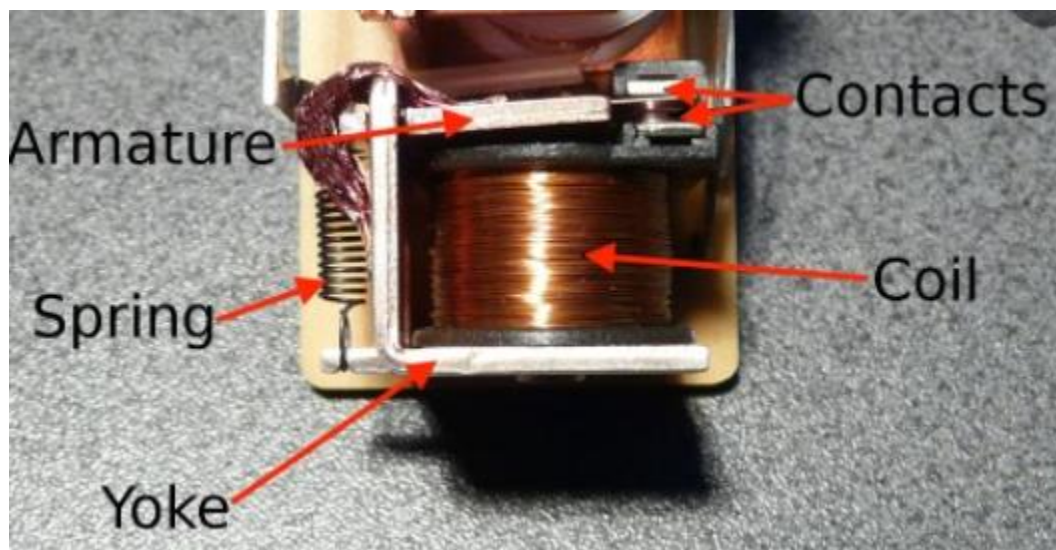
In order to blink 10 times at the rate of 1 time per second, we can run a for loop with delay between high and low of 500ms second, which counts as one blink. Essentially we blink once every seconds.

Exercise 5: Describe 2 devices (electronic components) that can be used as a switch, which is triggered electronically (i.e., the switch is electronically controlled by Arduino rather than manually controlled like push button). Explain how each device works.

1. Relay

A relay is an electrically operated switch. They commonly use an electromagnet (coil) to operate their internal mechanical switching mechanism (contacts). But instead of a manual operation, a relay uses an electrical signal to control an electromagnet, which in turn connects or disconnects another circuit. When a relay contact is open, this will switch power ON for a circuit when the coil is activated. And when the contact closes, the circuit loses power. Relays are used where it is necessary to control a circuit by an independent low-power signal, or where several circuits must be controlled by one signal.

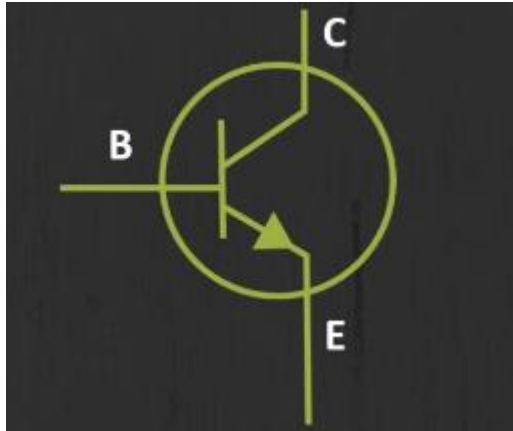
https://www.electronicshub.org/what-is-relay-and-how-it-works/#Poles_and_Throws



2. Bipolar junction transistors

Bipolar junction transistors are used as amplifiers, filters, oscillators or even switches. A bipolar transistor allows a small current injected at one of its terminals to control a much larger current flowing between the terminals, making the device capable of amplification and switching.

A transistor's collector current is limited proportionally by its base current. A small amount of electron flow through the base of the transistor has the ability to exert control over a much larger flow of electrons through the collector.



Exercise 6: Assume that you are designing a 4-way traffic light; what electronic components would you need? How could you control the system using Arduino? What electronic components would you use? In addition, at late nights, you want the system to prioritize oncoming traffic. (i.e., if 3 of the traffic lights had no cars at all, and one of them had a car on a red light, the traffic lights for that lane would automatically turn green. What sensors could you use to add this functionality?

Components:

The components you will be required for Arduino traffic light controller are as follows

- Arduino Uno
- Red LEDs (4 pieces) (1 for each way)
- Yellow LEDs (4 pieces) (1 for each way)
- Green LEDs (4 pieces) (1 for each way)
- 220 ohm resistors (12 pieces)
- Jumper cables
- Breadboards

There are total of 12 LEDs used in this project. Each signal has 3 LEDs (Red, Yellow and Green) connected to it through the 220 ohm resistors.

The resistors are used to limit the current that is going to pass through the LEDs. If we don't use the resistors, then the LEDs may burn due to excessive current.

Control system:

There are 4 traffic lights so each light must spend some time in green light, brief instances in yellow light, and long periods in red light.

In the beginning, green light of signal 1 and red lights at other signals will light up to give time to the vehicles at signal 1 to pass.

After 8 seconds, the yellow light at signal 1 will light up to give an indication that the red light at signal 1 is about to come up and also to give an indication to the vehicles at signal 2 that the green light is about to light up.

After 2 seconds, red light at signal 1 will turn on and green light at signal 2 will turn on meaning vehicles at signal 1 must stop and vehicles at signal 2 can move.

After 8 seconds, the yellow light at signal 2 will light up to give an indication that the red light at signal 2 is about to come up and also to give an indication to the vehicles at signal 3 that the green light is about to light up.

After 2 seconds, red light at signal 2 will turn on and green light at signal 3 will turn on meaning vehicles at signal 1 must stop and vehicles at signal 3 can move.

After 8 seconds, the yellow light at signal 3 will light up to give an indication that the red light at signal 3 is about to come up and also to give an indication to the vehicles at signal 4 that the green light is about to light up.

After 2 seconds, red light at signal 3 will turn on and green light at signal 4 will turn on meaning vehicles at signal 1 must stop and vehicles at signal 4 can move.

And so on.

Additional sensors

In order to detect traffic at a signal, we will have to setup ultrasonic sensors that emit sound waves which get reflected back if they encounter objects. Each of the 4 signals have ultrasonic sensors (4 total) to measure the amount of traffic at that signal.

Prioritize incoming traffic

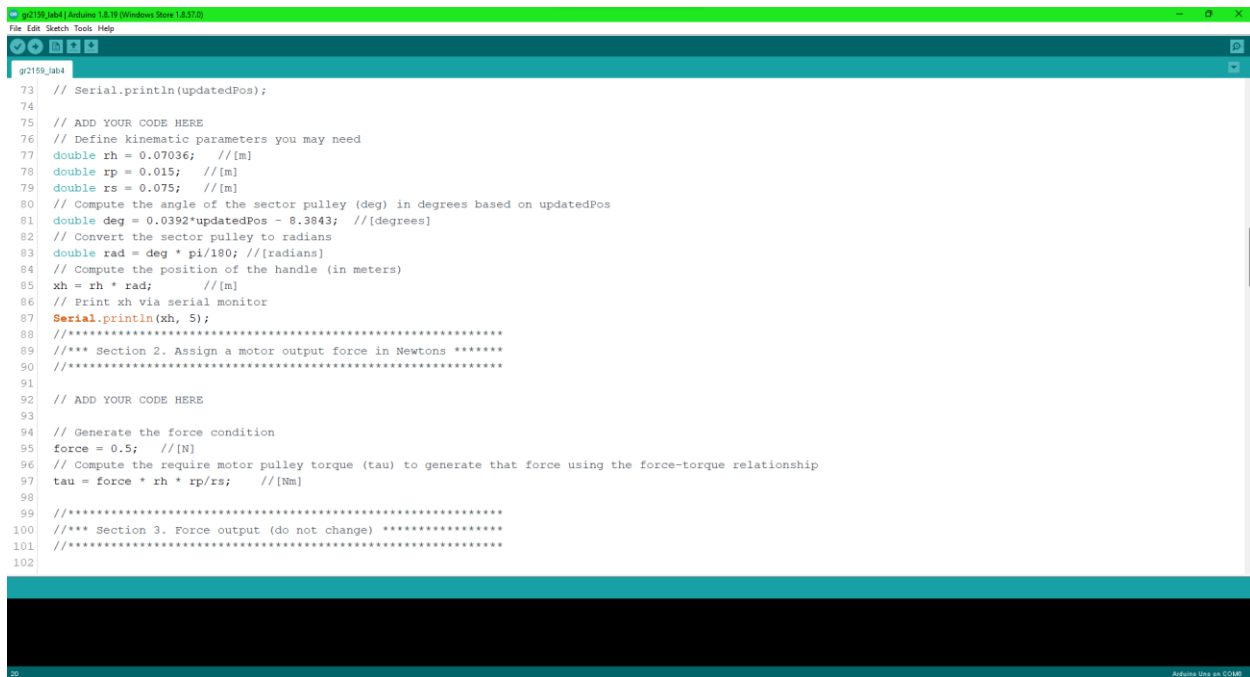
Base case is when we have equal amount of traffic. If the state of traffic at all the signals is identical, then the system will work normally by controlling the signals one by one.

If there is unequal traffic at one of the four signals, we will modify the algorithm. If there is no traffic near a signal, then the system will skip this signal during the green and yellow light phase and will move on to the next one. For example, if there is no vehicle at signal 2, 3 and currently

the system is allowing vehicles at signal 1 to pass. Then after signal 1, the system will move on to signal 4 skipping signal 2 and 3.

If there is no traffic at all the 4 signals, system will stop at the current signal and will only move on the next signal if there will be traffic at any other signal.

Exercise 2



```

g2159_lab4
73 // Serial.println(updatedPos);
74
75 // ADD YOUR CODE HERE
76 // Define kinematic parameters you may need
77 double rh = 0.07036; // [m]
78 double rp = 0.015; // [m]
79 double rs = 0.075; // [m]
80 // Compute the angle of the sector pulley (deg) in degrees based on updatedPos
81 double deg = 0.0392*updatedPos - 8.3843; //[degrees]
82 // Convert the sector pulley to radians
83 double rad = deg * pi/180; //[radians]
84 // Compute the position of the handle (in meters)
85 xh = rh * rad; // [m]
86 // Print xh via serial monitor
87 Serial.println(xh, 5);
88 //*****
89 //*** Section 2. Assign a motor output force in Newtons *****
90 //*****
91
92 // ADD YOUR CODE HERE
93
94 // Generate the force condition
95 force = 0.5; // [N]
96 // Compute the require motor pulley torque (tau) to generate that force using the force-torque relationship
97 tau = force * rh * rp/rs; // [Nm]
98
99 //*****
100 //*** Section 3. Force output (do not change) *****
101 //*****
102
  
```

