Practical No. 17

Aim: To interface PIR sensor with Arduino and write a program to turn on and off LED depending on motion detection.

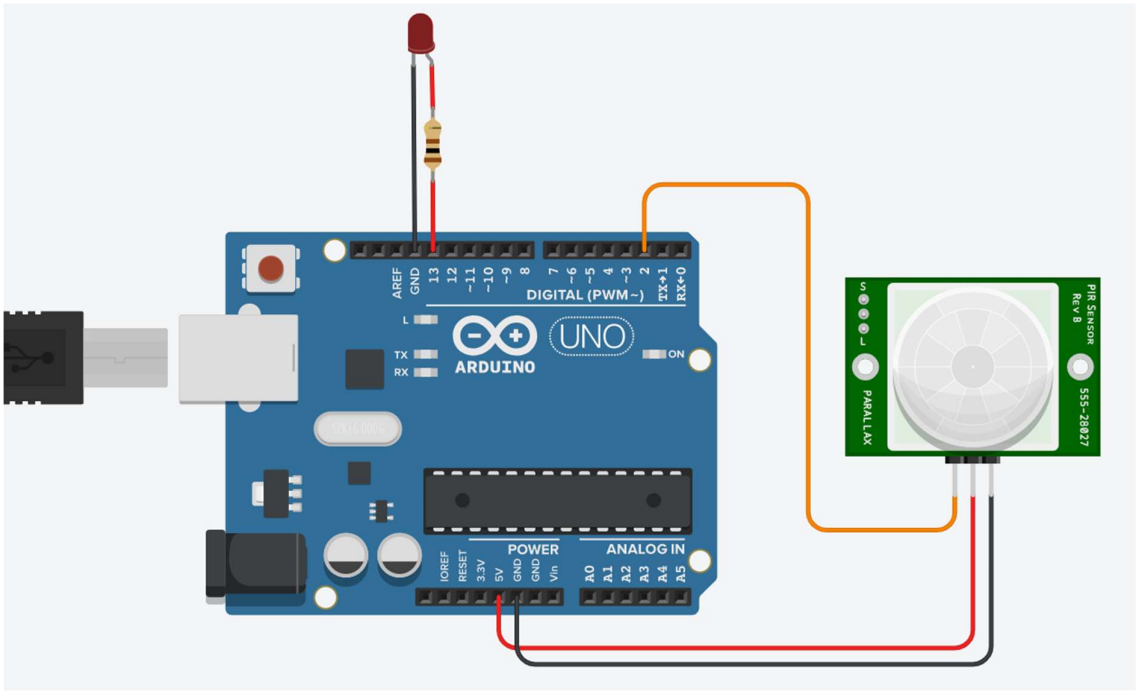
Component: Arduino Uno, PIR Sensor, Red LED, 100 Ω Resistor

Theory:

PIR sensors are more complicated than many of the other sensors explained in these tutorials (like photocells, FSRs and tilt switches) because there are multiple variables that affect the sensors input and output. To begin explaining how a basic sensor works, we'll use this rather nice diagram

The PIR sensor itself has two slots in it, each slot is made of a special material that is sensitive to IR. The lens used here is not really doing much and so we see that the two slots can 'see' out past some distance (basically the sensitivity of the sensor). When the sensor is idle, both slots detect the same amount of IR, the ambient amount radiated from the room or walls or outdoors. When a warm body like a human or animal passes by, it first intercepts one half of the PIR sensor, which causes a positive differential change between the two halves. When the warm body leaves the sensing area, the reverse happens, whereby the sensor generates a negative differential change. These change pulses are what is detected.

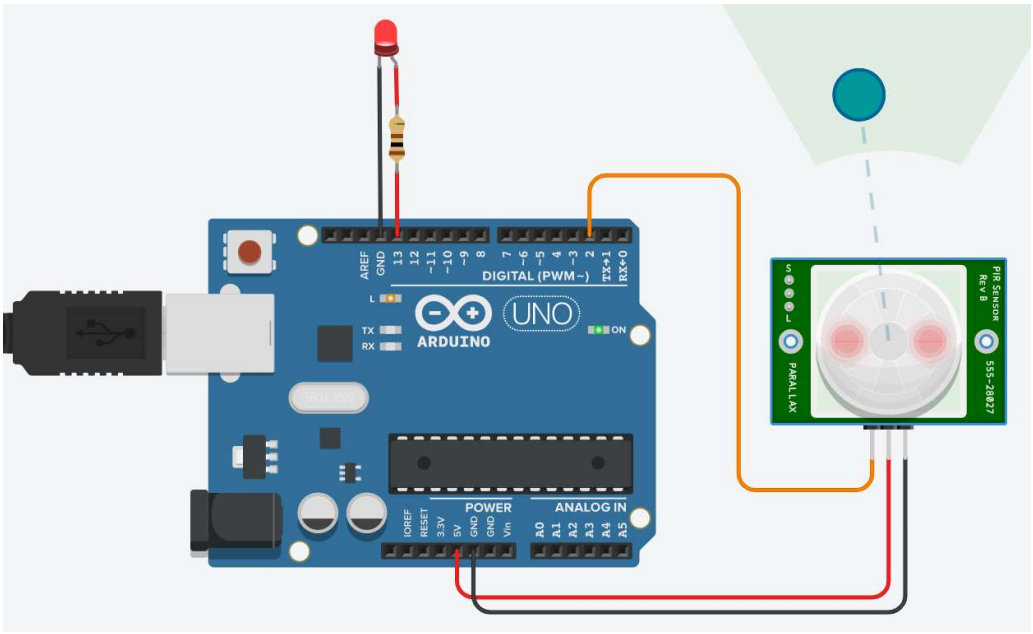
Circuit Diagram:

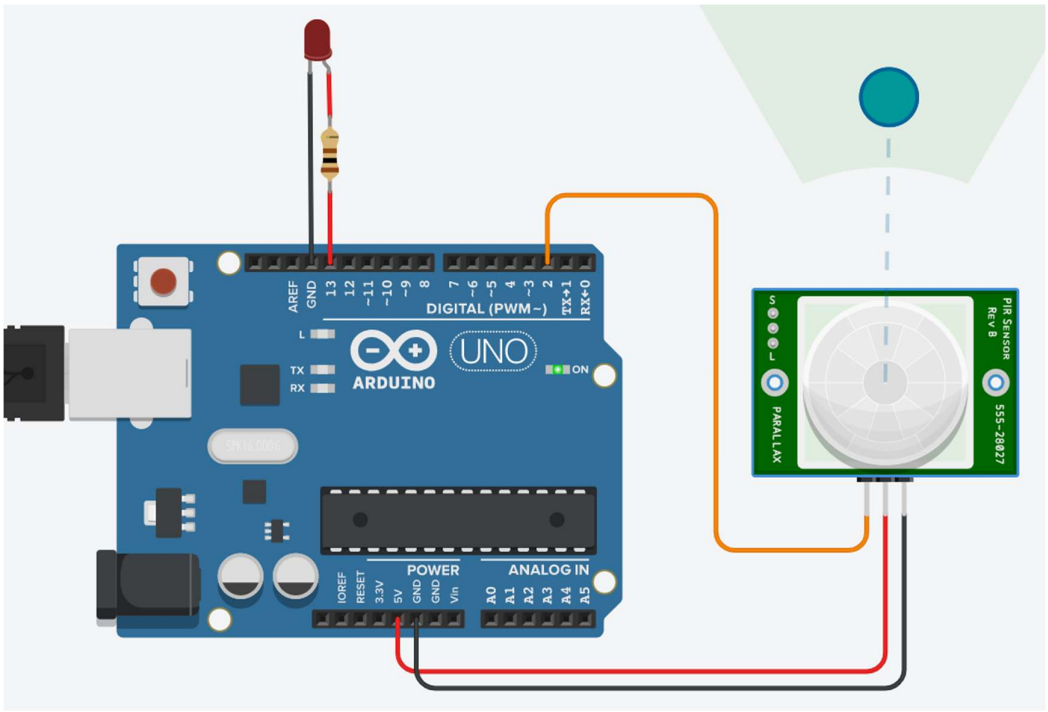


Program:

```
void setup()
{
  pinMode(13, OUTPUT);
  pinMode(2, INPUT);
}
void loop()
{
  int sensorMotion = digitalRead(2);
  if (sensorMotion == HIGH)
    digitalWrite(13, HIGH);
  else
    digitalWrite(13, LOW);
}
```

Output:


DIGITAL (PWM~)
UNO
ARDUINO
POWER
ANALOG IN
PIR SENSOR
PARALLAX


DIGITAL (PWM~)
UNO
ARDUINO
POWER
ANALOG IN
PIR SENSOR
PARALLAX

Practical No. 18

Aim: To interface Ultrasonic sensor with Arduino and write a program to display object distance(in inch/cm) in serial monitor depending on sound detection.

Component: Arduino Uno, Ultrasonic Distance Sensor

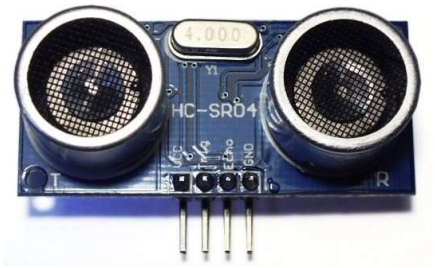
Theory:

An ultrasonic sensor is an instrument that measures the distance to an object using ultrasonic sound waves.

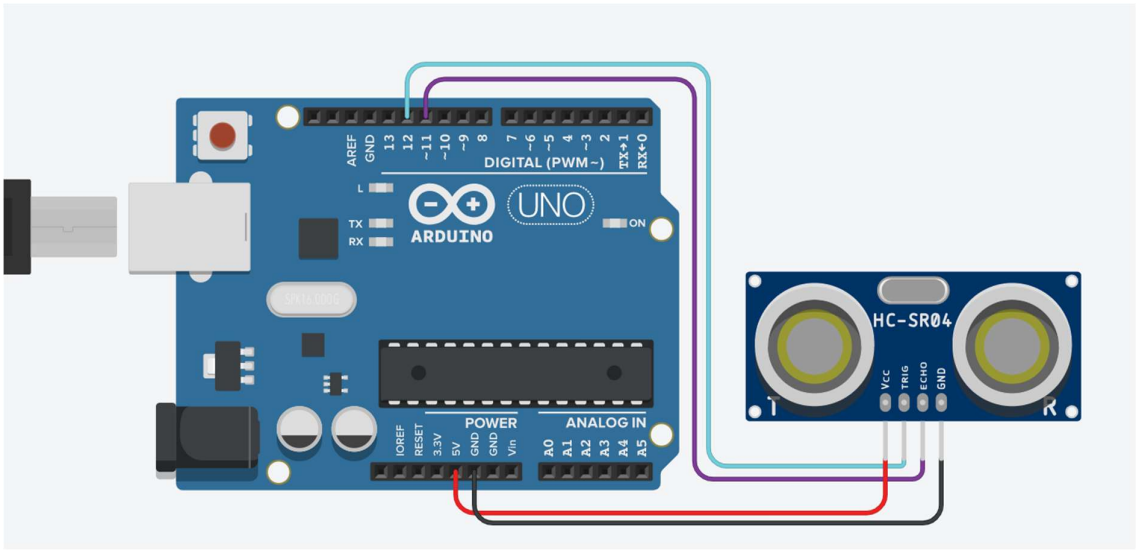
An ultrasonic sensor uses a transducer to send and receive ultrasonic pulses that relay back information about an object's proximity.

High-frequency sound waves reflect from boundaries to produce distinct echo patterns

Ultrasonic sensors work by sending out a sound wave at a frequency above the range of human hearing. The transducer of the sensor acts as a microphone to receive and send the ultrasonic sound. Our ultrasonic sensors, like many others, use a single transducer to send a pulse and to receive the echo. The sensor determines the distance to a target by measuring time lapses between the sending and receiving of the ultrasonic pulse.



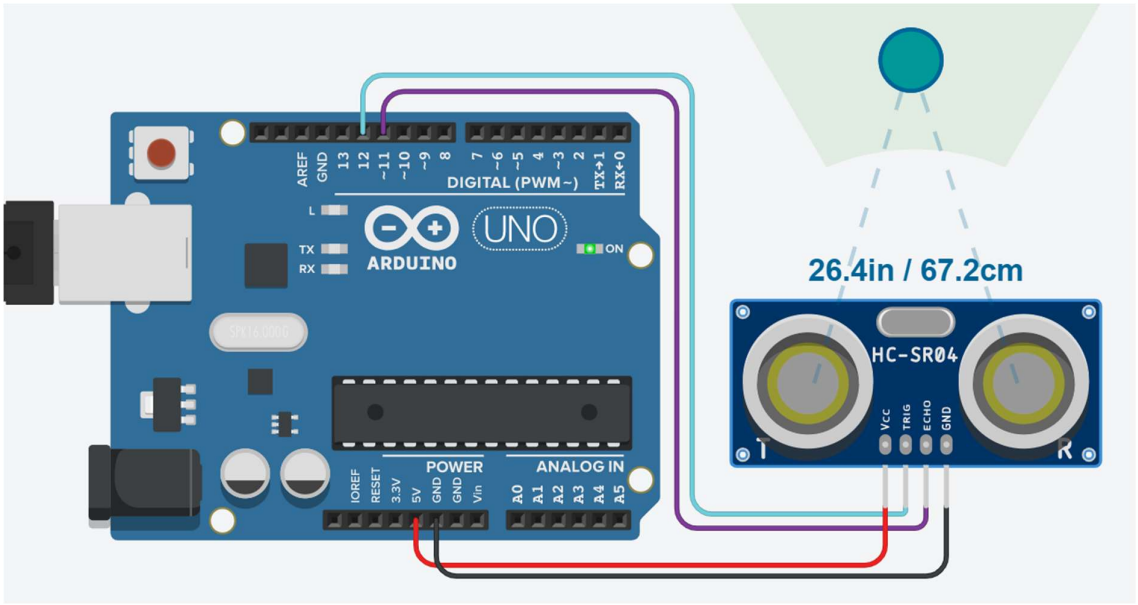
Circuit Diagram:



Program:

```
int trig = 12;
int echo = 11;
int travelTime;
int distance;
int distanceInch;

void setup()
{
  pinMode(trig, OUTPUT);
  pinMode(echo, INPUT);
  Serial.begin(9600);
}
void loop()
{
  digitalWrite(trig , LOW);
  delayMicroseconds(10);
  digitalWrite(trig , HIGH);
  delay(10);
  digitalWrite(trig , LOW);
  travelTime = pulseIn(echo,HIGH);
  delay(20);
  distance = (travelTime / 2) * 0.0343;
  distanceInch = ( travelTime / 2)* 0.013464 ;
  Serial.print("Distance in CM ");
  Serial.println(distance);
  Serial.print("Distance in INCHES ");
  Serial.println(distanceInch);
}
```

Output:



Serial Monitor

```
Distance in INCHES 25
Distance in CM 65
Distance in INCHES 25
Distance in CM 65
Distance in INCHES 25
Distance in CM 65
Distance in INCHES 25
Distance in CM 65
Distance in INCHES 25
Distance in CM 65
Distance in INCHES 25
Distance in CM 65
Distance in INCHES 25
Distance in CM 65
Distance in INCHES 25
Distance in CM 65
Distance in INCHES 25
Distance in CM 65
Distance in INCHES 25
Distance in CM 65
Distance in INCHES 25
Distance in CM 65
```

Practical No. 19

Aim: To interface servo motor with Arduino and write a program to sweep a servo back and forth through its full range of motion.

Component: Arduino Uno, Positional Micro Servo

Theory:

A servomotor is a rotary actuator or linear actuator that allows for precise control of angular or linear position, velocity and acceleration.[1] It consists of a suitable motor coupled to a sensor for position feedback. It also requires a relatively sophisticated controller, often a dedicated module designed specifically for use with servomotors.

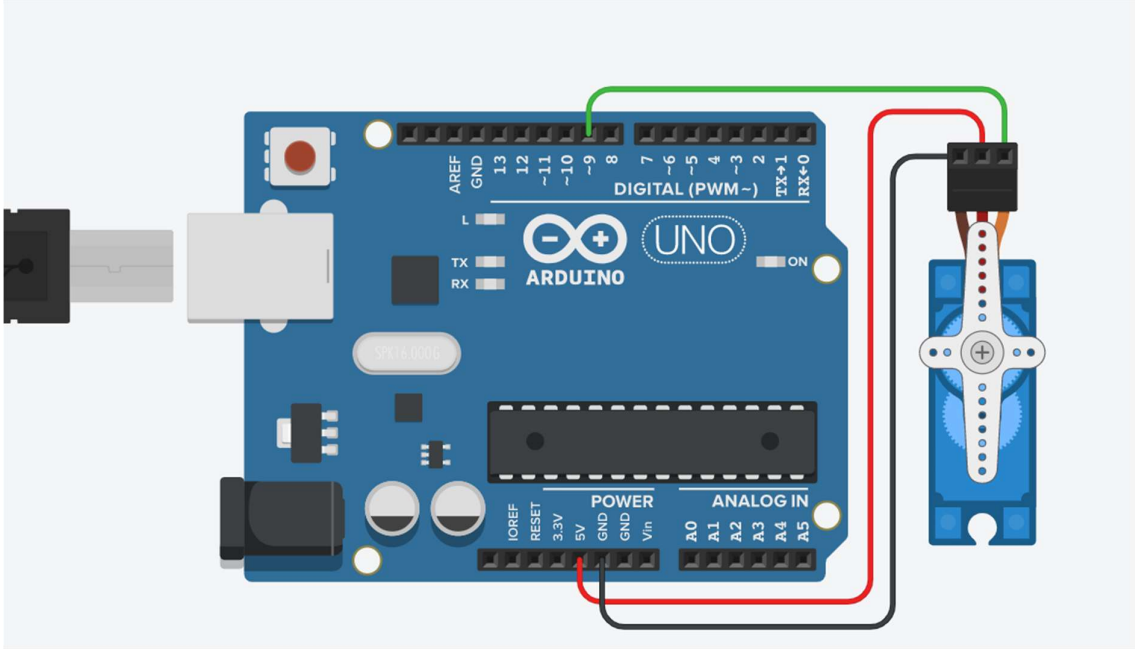
Servomotors are not a specific class of motor, although the term servomotor is often used to refer to a motor suitable for use in a closed-loop control system.

Servomotors are used in applications such as robotics, CNC machinery or automated manufacturing.

A servomotor is a closed-loop servomechanism that uses position feedback to control its motion and final position. The input to its control is a signal (either analogue or digital) representing the position commanded for the output shaft.

The motor is paired with some type of position encoder to provide position and speed feedback. In the simplest case, only the position is measured. The measured position of the output is compared to the command position, the external input to the controller. If the output position differs from that required, an error signal is generated which then causes the motor to rotate in either direction, as needed to bring the output shaft to the appropriate position. As the positions approach, the error signal reduces to zero and the motor stops.

Circuit Diagram:

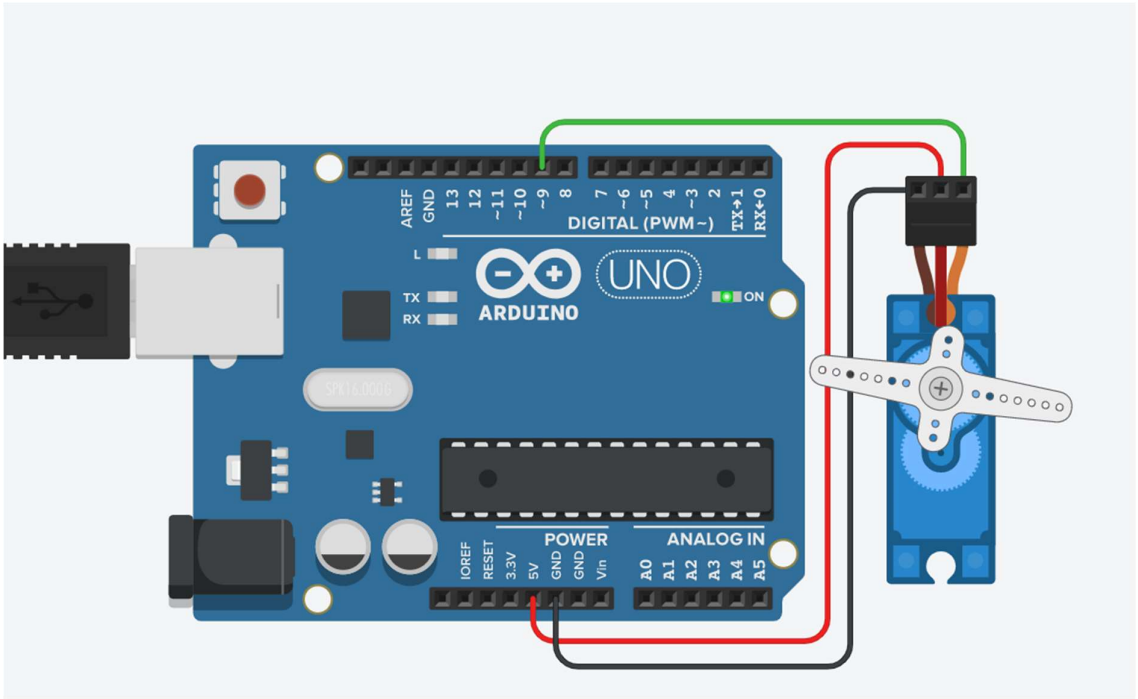


Program:

```
#include <Servo.h>

int servoPin = 9;
int servoPos = 45;
int servoInitalPos = 180;
Servo myServo;

void setup()
{
  myServo.attach(servoPin);
}
void loop()
{
 int pos = 0;
 int dtwait=15;
```

```
for(pos = 0; pos < 180; pos += 1) {
myServo.write(pos);
delay(dtwait);
}
for(pos = 180; pos>=1; pos -= 1) {
myServo.write(pos);
delay(dtwait);
}
}
```

Output:



Practical No. 20

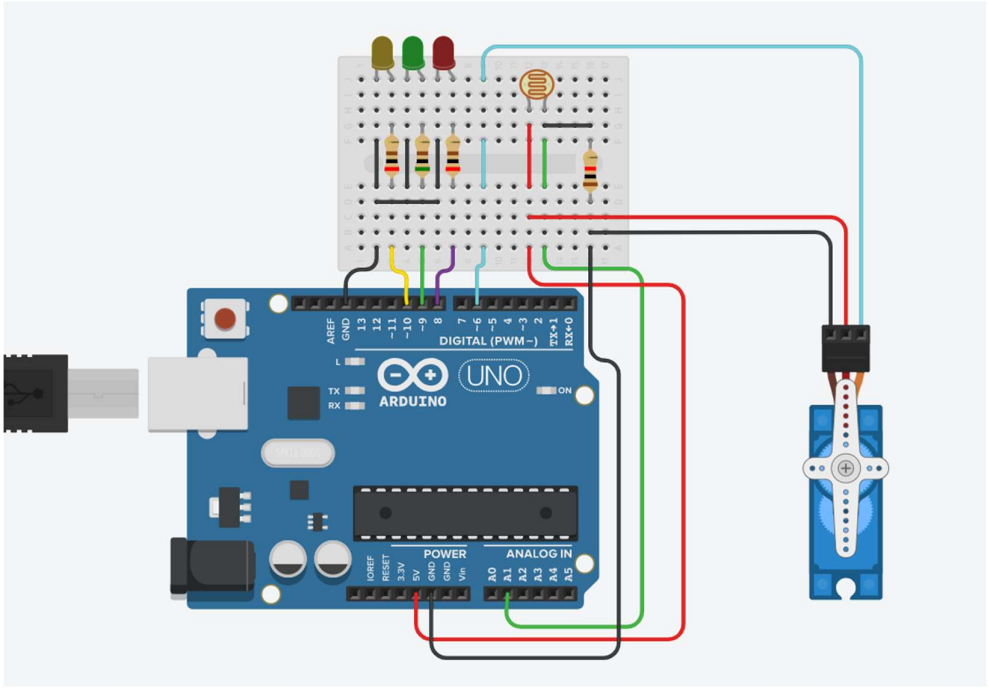
Aim: To interface servo motor, Photo resistor (LDR) with Arduino and write a program to sweep a servo back and forth through its full range of motion depending on light.

Component:

Quantity	Component
1	Arduino Uno R3
1	Photoresistor
1	Green LED
4	0.2 kΩ Resistor
1	1 kΩ Resistor
1	Positional Micro Servo
1	Red LED
1	Yellow LED
1	0.5 kΩ Resistor

Theory:

Circuit Diagram:

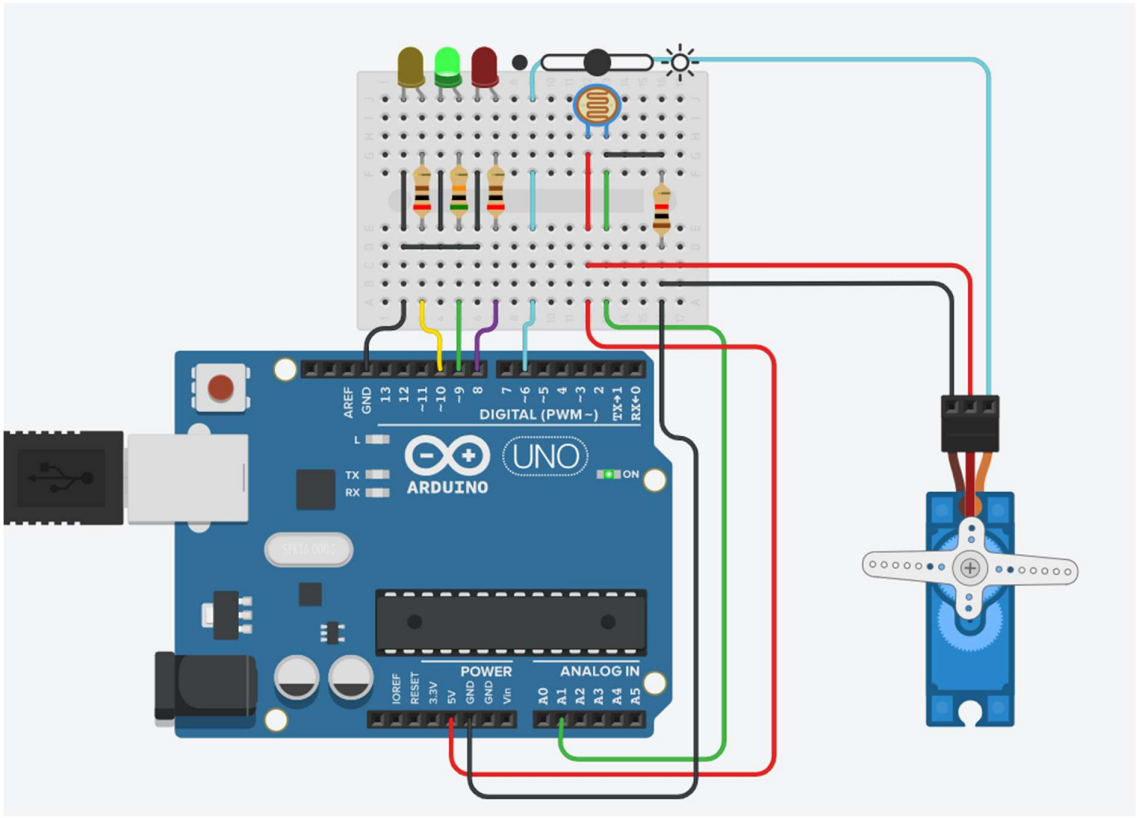


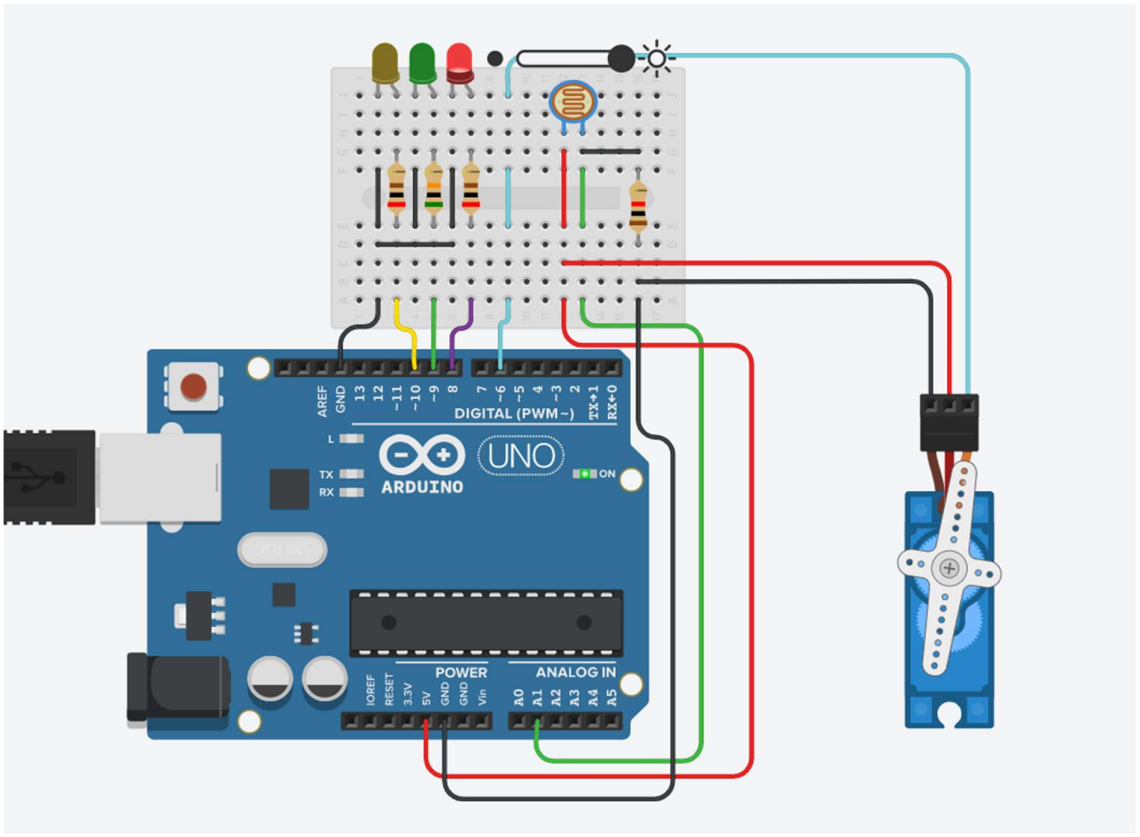
Program:

```
#include <Servo.h>

int servoPin = 6;
int servoPos = 0;
int red = 8;
int green = 9;
int yellow = 10;
Servo myServo;

void setup()
{
  pinMode(red, OUTPUT);
  pinMode(green, OUTPUT);
  pinMode(yellow, OUTPUT);
  myServo.attach(servoPin);
  Serial.begin(9600);
}

void loop()
{
  Serial.println(analogRead(A1));
  Serial.println(analogRead(A1));
  if (analogRead(A1) <= 700 && analogRead(A1) >595){
    digitalWrite(red, HIGH);
    digitalWrite(green, LOW);
    digitalWrite(yellow, LOW);
    myServo.write(180);
  }
  else if (analogRead(A1) <= 595 && analogRead(A1) > 200){
```

```
    digitalWrite(red, LOW);
    digitalWrite(green, HIGH);
    digitalWrite(yellow, LOW);
    myServo.write(90);
  }
  else{
    digitalWrite(red, LOW);
    digitalWrite(green, LOW);
    digitalWrite(yellow, HIGH);
    myServo.write(0);
  }

}
```

Output:




AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
L
TX
RX
ARDUINO
UNO
ON
POWER
ANALOG IN
IOREF
RESET
3.3V
5V
GND
GND
Vin
A0
A1
A2
A3
A4
A5