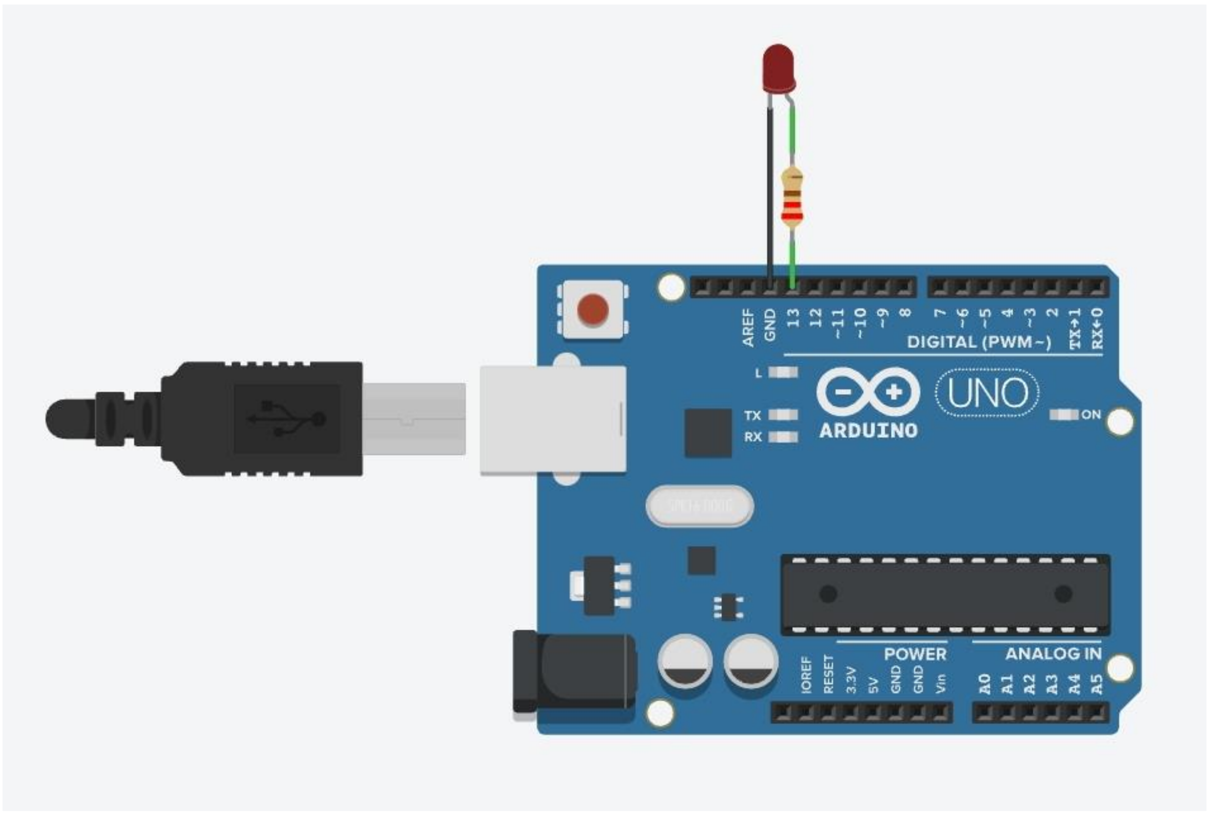
1.Blinking led

Program:

```
int LED = 13;
void setup()
{
  pinMode(LED, OUTPUT);
}
void loop()
{
  digitalWrite(LED, HIGH);
  delay(1000); // Wait for 1000 millisecond(s)
  digitalWrite(LED, LOW);
  delay(1000); // Wait for 1000 millisecond(s)
}
```

Circuit:

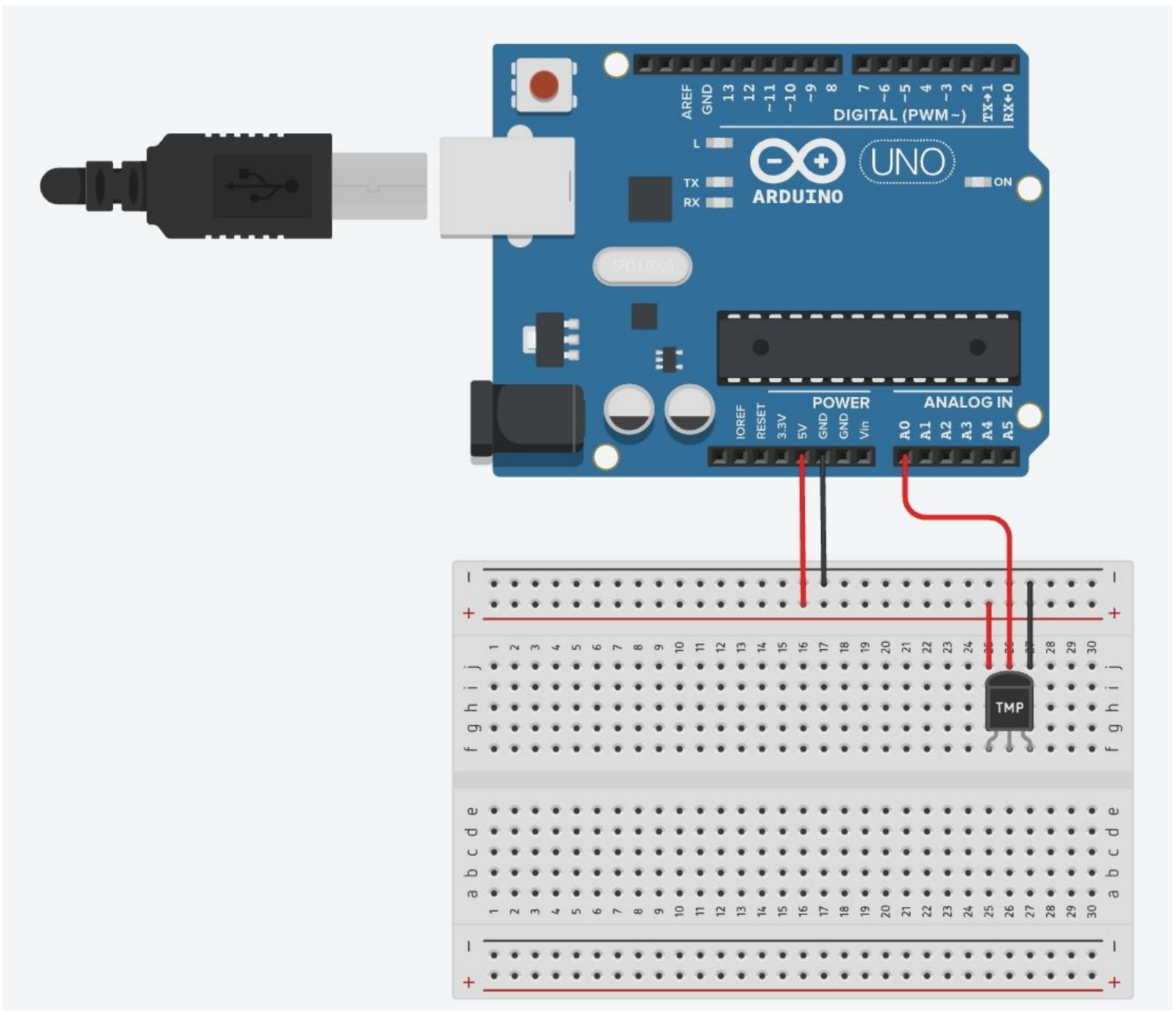


2.Measuring temperature

Program:

```
int outputpin=A0;
void setup() {
Serial.begin(9600);
}
void loop() //main loop
{
int analogValue = analogRead(outputpin);
float millivolts = (analogValue/1024.0) * 3300; //3300 is the voltage provided by NodeMCU
float celsius = millivolts/10;
Serial.print("in DegreeC= ");
Serial.println(celsius);
float fahrenheit = ((celsius*9)/5+32);
delay(1000);
}
```

Circuit:

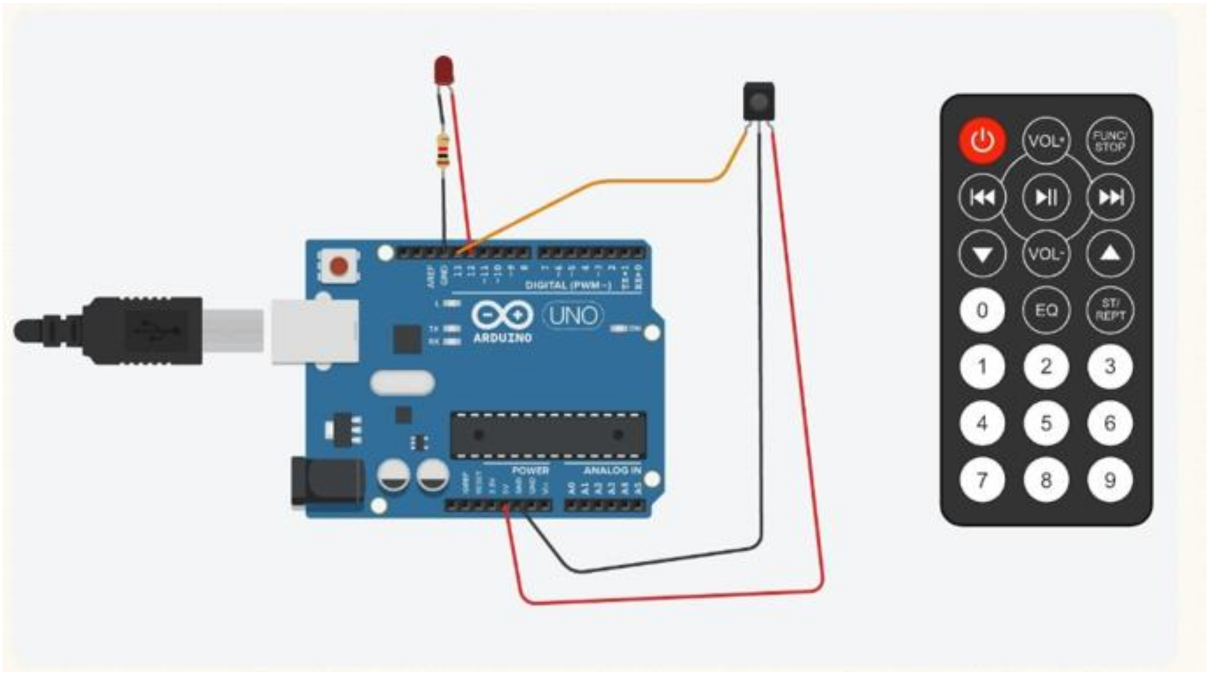


3.IR SENSOR

Program:

```
int ledPin = 12;
int inputPin = 13;
int val = 0;
void setup()
{
pinMode(ledPin, OUTPUT);
pinMode(inputPin, INPUT);
}
void loop()
{
val = digitalRead(inputPin);
if (val == HIGH)
{
digitalWrite(ledPin, LOW);
}
else
{
digitalWrite(ledPin, HIGH);
}
}
```

Circuit:



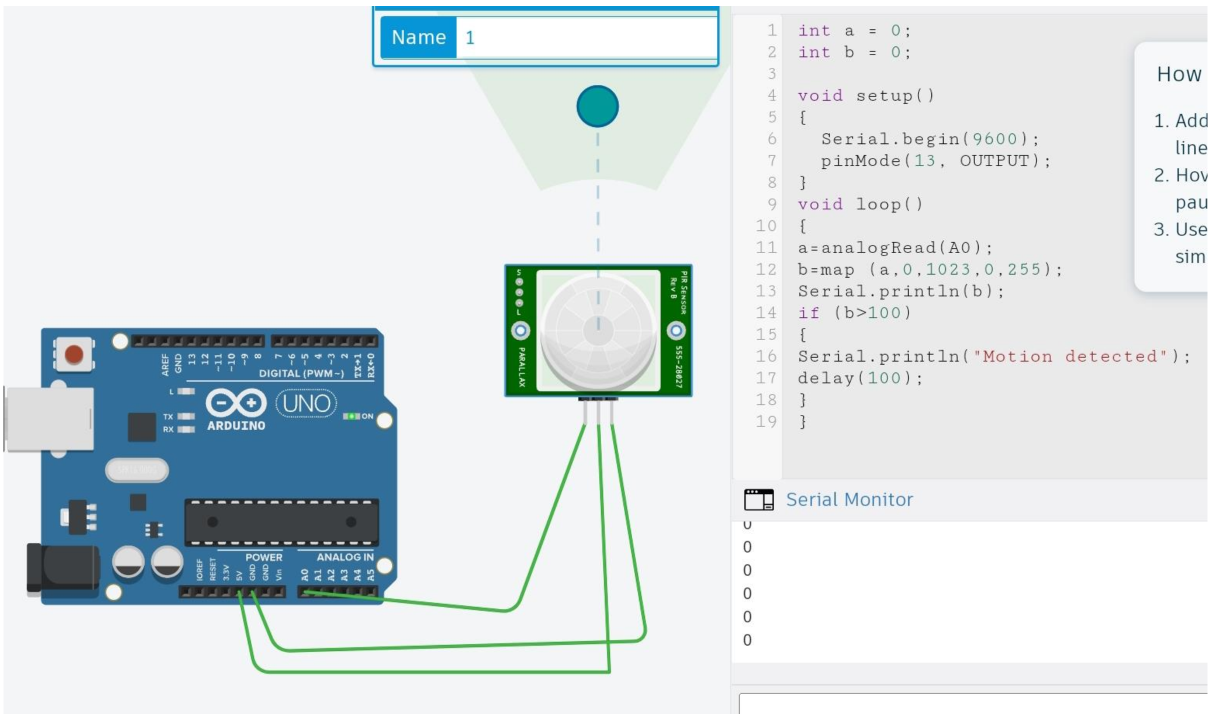
4.PIR SENSOR

Program:

```
int a = 0;
int b = 0;

void setup()
{
  Serial.begin(9600);
  pinMode(13, OUTPUT);
}
void loop()
{
a=analogRead(A0);
b=map (a,0,1023,0,255);
Serial.println(b);
if (b>100)
{
Serial.println("Motion detected");
delay(100);
}
}
```

Circuit:



```
int a = 0;
int b = 0;

void setup()
{
  Serial.begin(9600);
  pinMode(13, OUTPUT);
}
void loop()
{
a=analogRead(A0);
b=map (a,0,1023,0,255);
Serial.println(b);
if (b>100)
{
Serial.println("Motion detected");
delay(100);
}
}
```

```
0
0
0
0
0
```

5.Ultrasonic sensor

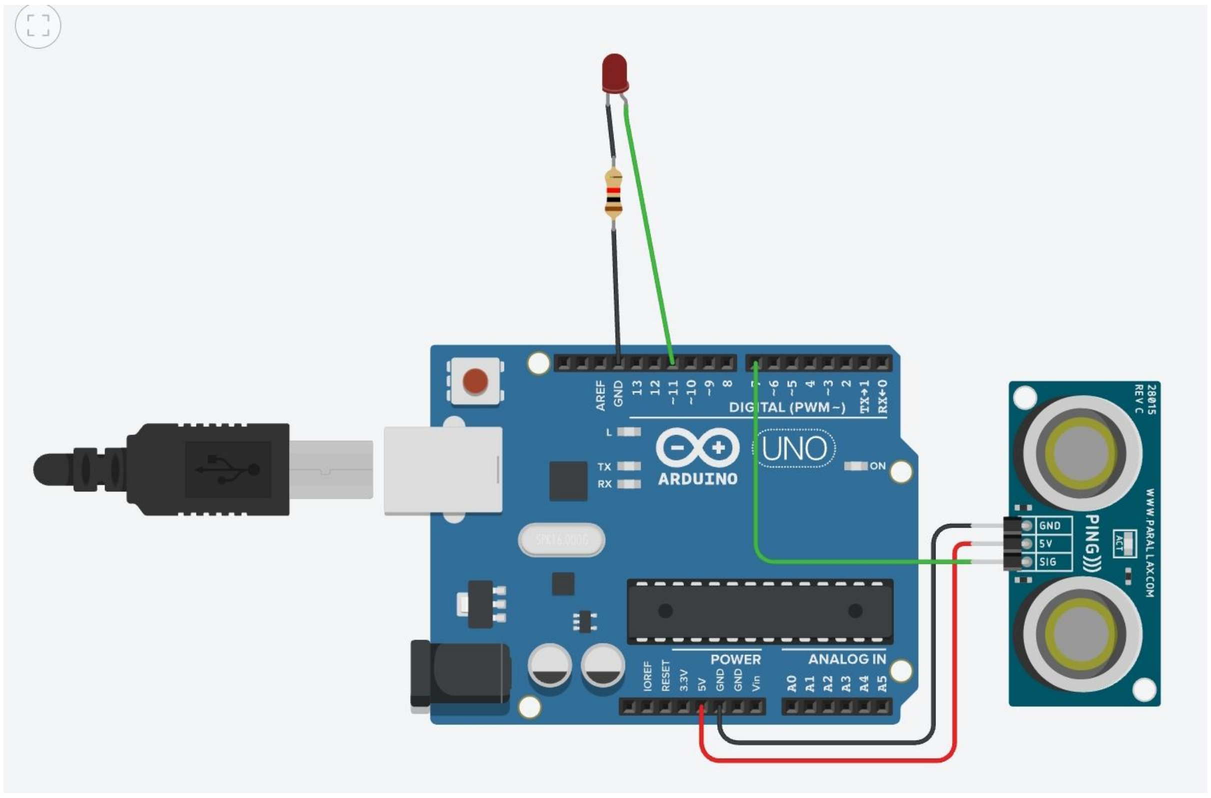
Program:

```
int cm = 0;
long readUltrasonicDistance(int triggerPin, int echoPin)
{
  pinMode(triggerPin, OUTPUT);  // Clear the trigger
  digitalWrite(triggerPin, LOW);
  delayMicroseconds(2);
  digitalWrite(triggerPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(triggerPin, LOW);
  pinMode(echoPin, INPUT);
  return pulseIn(echoPin, HIGH);
}

void setup()
{
  pinMode(11, OUTPUT);
}

void loop()
{
  cm = 0.01723 * readUltrasonicDistance(7, 7);
  if (cm < 50) {
    digitalWrite(11, HIGH);
  } else {
    digitalWrite(11, LOW);
  }
  delay(100);
}
```

Circuit:


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
IOREF
RESET
3.3V
5V
GND
GND
Vin
POWER
ANALOG IN
A0
A1
A2
A3
A4
A5
GND
5V
SIG
PING)))
ACT
WWW.PARALLAX.COM
2015
REV C

Iot cloud

Program:

```
#include <ESP8266WiFi.h>
#include “ThingSpeak.h”
Char msg[50];
Const char* ssid = “boolean”; //(your network SSID (name)
Const char* password = “meow meow”; // your network password
WiFiClient client;
Unsigned long myChannelNumber = 00000000; // replace with your channel number
Const char *myWriteAPIKey = “UGGIHGJNIH 云”; // replace with your write API key
// Time variables

Unsigned long lastTime = 0;
Unsigned long timerDelay = 30000; // 30 seconds

// Variable to hold temperature readings
Float temperature;

Int outputPin = A0;

Void setup() {
Serial.begin(115200); // Initialize serial
WiFi.mode(WIFI_STA);
ThingSpeak.begin(client);
Serial.print(“Attempting to connect to SSID: “);
Serial.println(ssid);
While (WiFi.status() != WL_CONNECTED) {
WiFi.begin(ssid, password);
Delay(10000); // Delay for 10 seconds
Serial.print(“.”);
}
Serial.println(“Connected to WiFi”);
}

Void loop() {
If ((millis() – lastTime) > timerDelay) {
Int analogValue = analogRead(outputPin);
Float millivolts = (analogValue / 1024.0) * 3300;
Temperature = millivolts / 10; // Adjust the conversion factor as needed
Serial.print(“Temperature (°C): “);
Serial.println(temperature);
Int x = ThingSpeak.writeField(myChannelNumber, 1, temperature, myWriteAPIKey);
If (x == 200) {
Serial.println(“Channel update successful.”);
} else {
Serial.println(“Problem updating channel. HTTP error code “ + String(x));
}
lastTime = millis();
}
}
```

Single led

Program:

```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ThingSpeak.h>

Const char* ssid = “YOUR WIFI SSID”;
Const char* password = “YOUR WIFI PASSWORD”;

Unsigned long channelID = YOUR_CHANNEL_ID;
Const char* writeAPIKey = “YOUR_WRITE_API_KEY”;

Const int ledPin = 13;

WiFiClient client;

Void setup() {
Serial.begin(115200);

WiFi.begin(ssid, password);

While (WiFi.status() != WL_CONNECTED) {
Delay(500);
Serial.print(“.”);
}

Serial.println(“”);
Serial.println(“WiFi Connected”);

ThingSpeak.begin(client);

pinMode(ledPin, OUTPUT);
}

Void loop() {
Int status = ThingSpeak.writeField(channelID, 1, random(0, 100), writeAPIKey);

If (status == 200) {
Serial.println(“Channel update successful”);
digitalWrite(ledPin, HIGH);
delay(1000);
digitalWrite(ledPin, LOW);
delay(1000);
} else {
Serial.print(“Problem updating channel. HTTP error code: “);
Serial.println(status);
}

Delay(20000); // Wait 20 seconds before sending the next update
}
```

Multiple led

Program:

```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ThingSpeak.h>

Const char* ssid = "your-wifi-ssid";
Const char* password = "your-wifi-password";
Unsigned long channelID = your_channel_ID; // Replace with your channel ID
Const char* writeAPIKey = "your-write-API-key"; // Replace with your write API key
Const int ledPin = 0; // GPIO0
Const int ledPin2 = D3; // GPIO0 (D3 on NodeMCU)

WiFiClient client;

Void setup() {
Serial.begin(115200);

WiFi.begin(ssid, password);

While (WiFi.status() != WL_CONNECTED) {
Delay(500);
Serial.print(".");
}

Serial.println("");
Serial.println("WiFi connected");

ThingSpeak.begin(client);

pinMode(ledPin, OUTPUT);
pinMode(ledPin2, OUTPUT);
}

Void loop() {
Int status = ThingSpeak.writeField(channelID, 1, random(0, 100), writeAPIKey);
If (status == 200) {
Serial.println("Channel update successful!");
digitalWrite(ledPin, HIGH);
digitalWrite(ledPin2, HIGH);
delay(1000);
digitalWrite(ledPin, LOW);
digitalWrite(ledPin2, LOW);
delay(1000);
} else {
Serial.print("Problem updating channel. HTTP error code: ");
Serial.println(status);
}
Delay(20000); // Wait 20 seconds before sending the next update
}
```