EXPERIMENT NO : 01

AIM : To design Blinking-LED using Arduino.

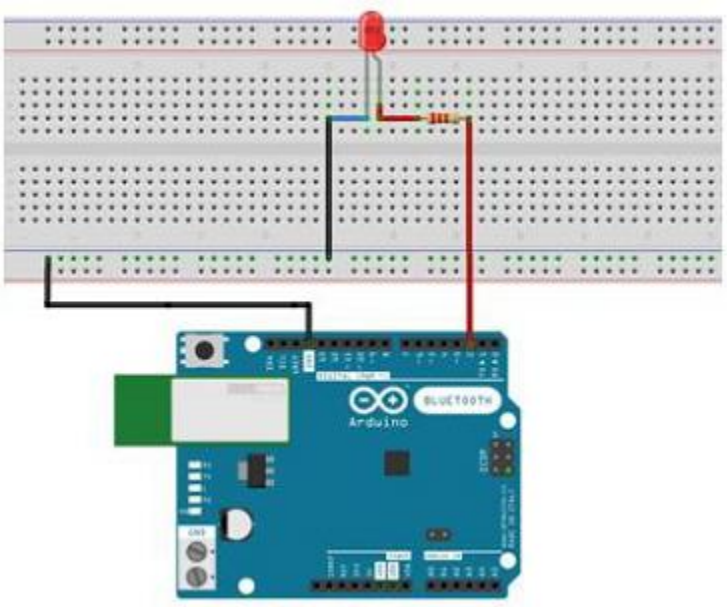
Components Required

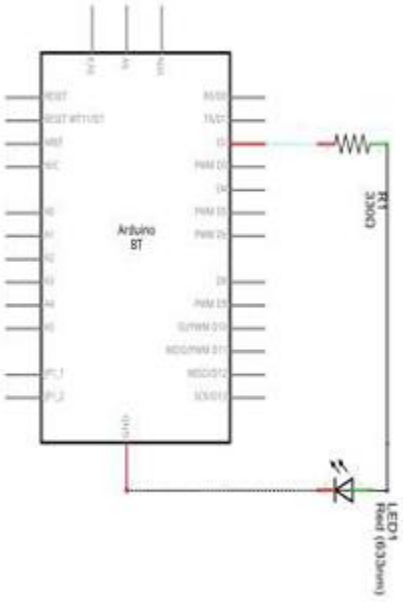
You will need the following components −

- 1 × Breadboard
- 1 × Arduino Uno R3
- 1 × LED
- 1 × 330Ω Resistor
- 2 × Jumper

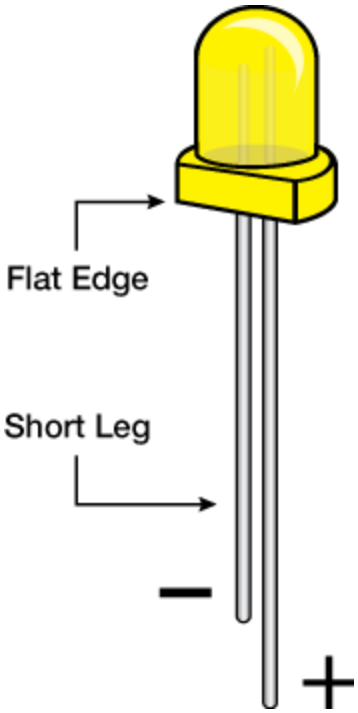
Procedure

Follow the circuit diagram and hook up the components on the breadboard as shown in the image given below.

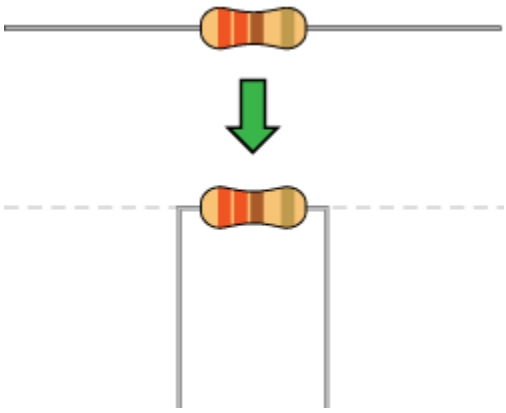




Note − To find out the polarity of an LED, look at it closely. The shorter of the two legs, towards the flat edge of the bulb indicates the negative terminal.

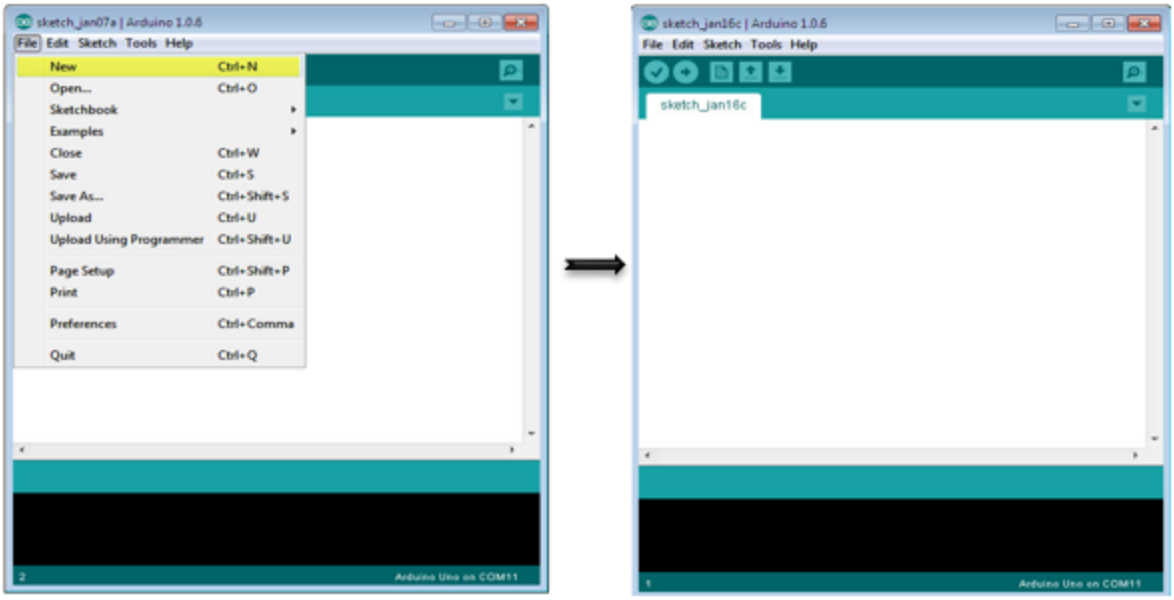


Components like resistors need to have their terminals bent into 90° angles in order to fit the breadboard sockets properly. You can also cut the terminals shorter.



Sketch

Open the Arduino IDE software on your computer. Coding in the Arduino language will control your circuit. Open the new sketch File by clicking New.



Arduino Code

```
/*
   Blink
   Turns on an LED on for one second, then off for one second,
repeatedly.
*/

// the setup function runs once when you press reset or power the
board

void setup() {  // initialize digital pin 13 as an output.
   pinMode(2, OUTPUT);
}

// the loop function runs over and over again forever

void loop() {
   digitalWrite(2, HIGH); // turn the LED on (HIGH is the voltage
level)
   delay(1000); // wait for a second
   digitalWrite(2, LOW); // turn the LED off by making the voltage
LOW
   delay(1000); // wait for a second
}
```

Code to Note

pinMode(2, OUTPUT) − Before you can use one of Arduino's pins, you need to tell Arduino Uno R3 whether it is an INPUT or OUTPUT. We use a built-in "function" called pinMode() to do this.

digitalWrite(2, HIGH) − When you are using a pin as an OUTPUT, you can command it to be HIGH (output 5 volts), or LOW (output 0 volts).

Result

You should see your LED turn on and off. If the required output is not seen, make sure you have assembled the circuit correctly, and verified and uploaded the code to your board.

EXPERIMENT NO : 02

AIM : Humidity-sensor using Arduino.

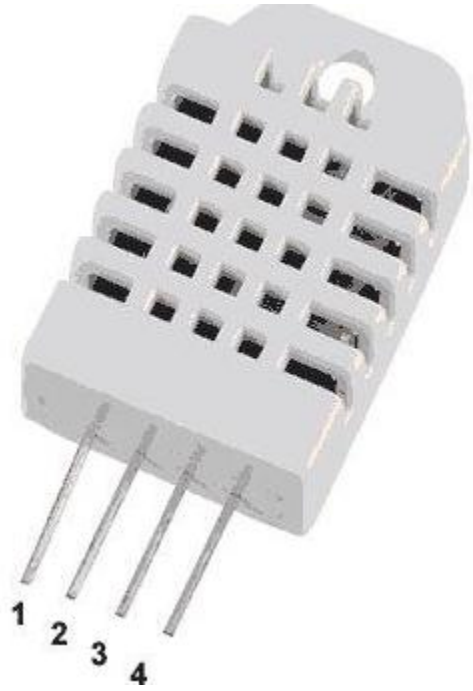
Humidity Sensor (DHT22)

The DHT-22 (also named as AM2302) is a digital-output, relative humidity, and temperature sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air, and sends a digital signal on the data pin.

In this example, you will learn how to use this sensor with Arduino UNO. The room temperature and humidity will be printed to the serial monitor.

The DHT-22 Sensor

DHT22 pins	
1	VCC
2	DATA
3	NC
4	GND



The connections are simple. The first pin on the left to 3-5V power, the second pin to the data input pin and the right-most pin to the ground.

Technical Details

- **Power** − 3-5V
- **Max Current** − 2.5mA
- **Humidity** − 0-100%, 2-5% accuracy
- **Temperature** − 40 to 80°C, ±0.5°C accuracy

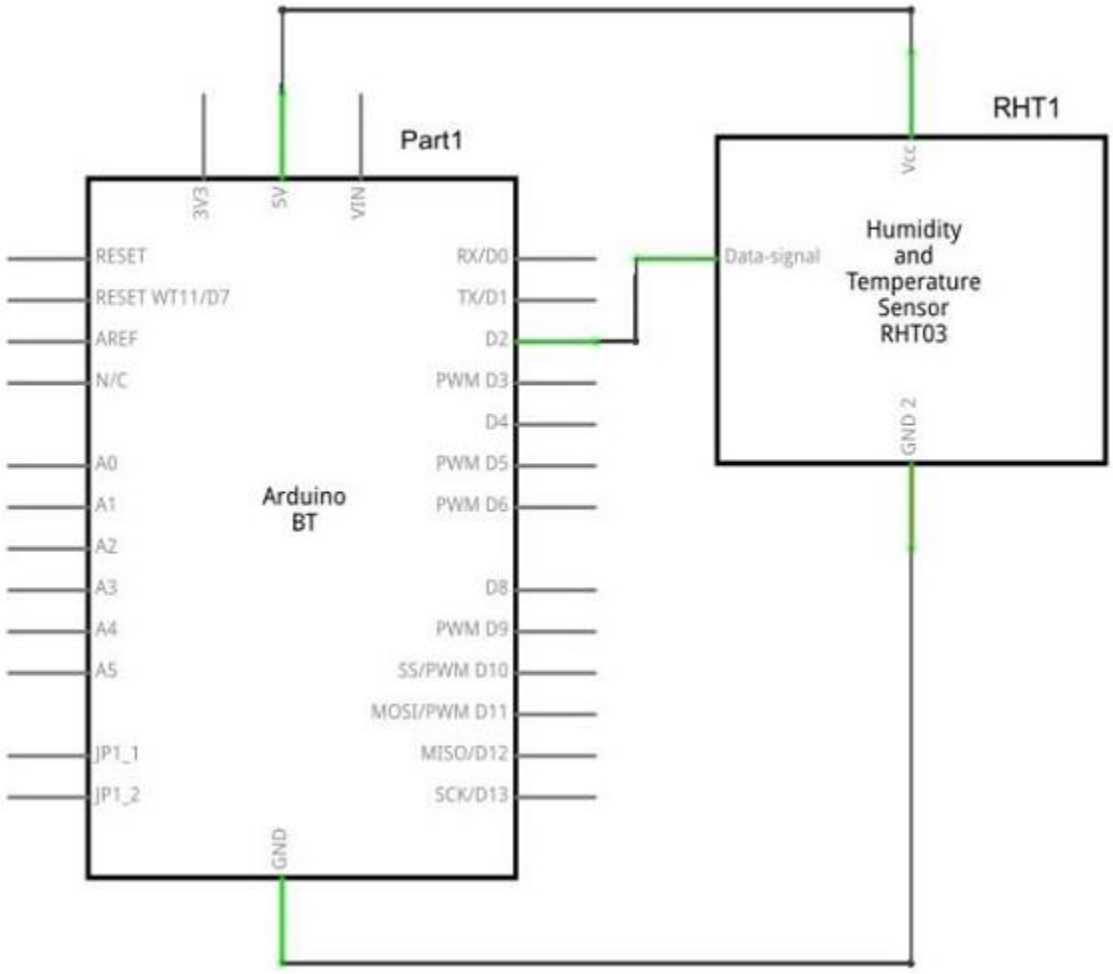
Components Required

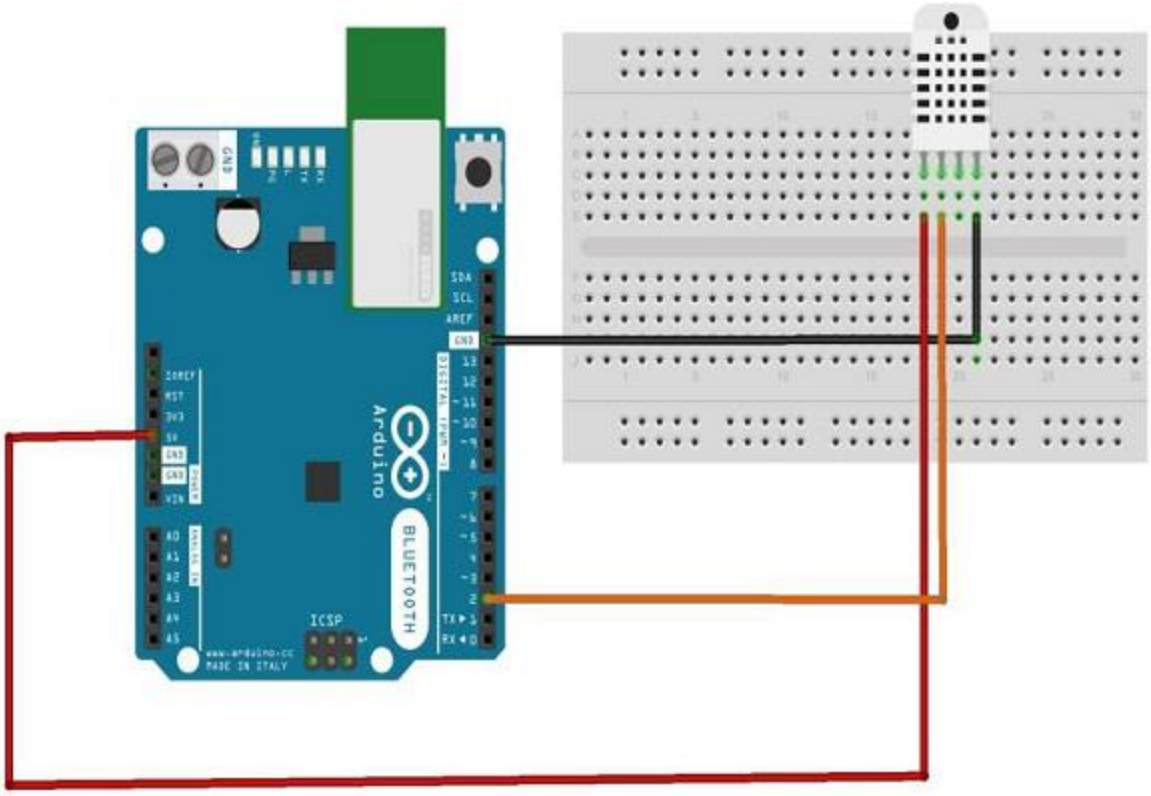
You will need the following components −

- 1 × Breadboard
- 1 × Arduino Uno R3
- 1 × DHT22
- 1 × 10K ohm resistor

Procedure

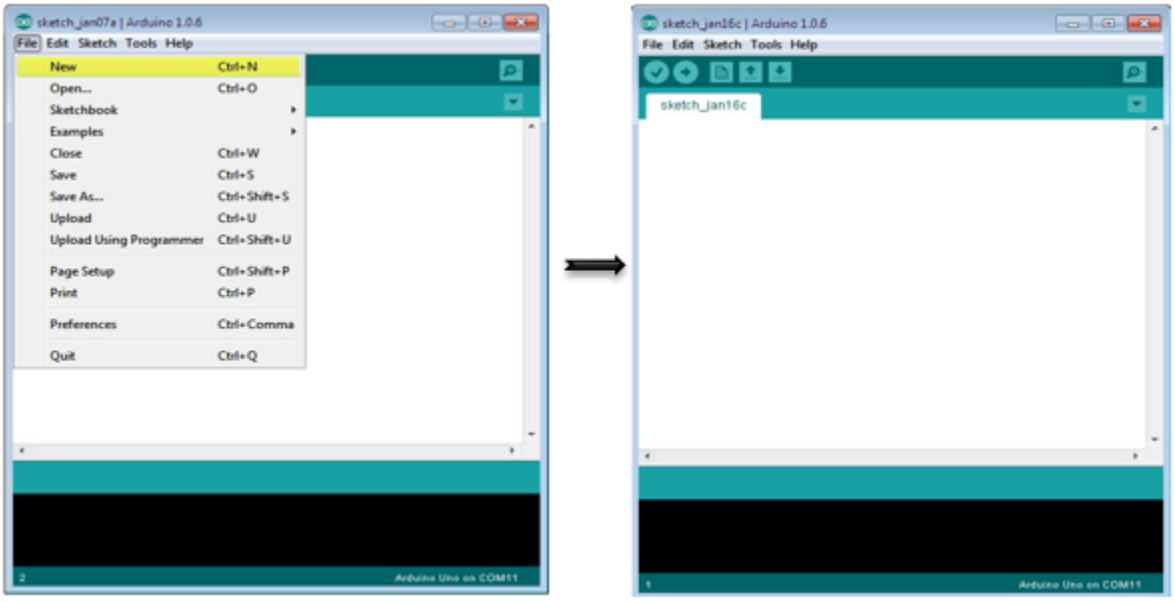
Follow the circuit diagram and hook up the components on the breadboard as shown in the image below.





Sketch

Open the Arduino IDE software on your computer. Coding in the Arduino language will control your circuit. Open a new sketch File by clicking New.



Arduino Code

```
// Example testing sketch for various DHT humidity/temperature
sensors

#include "DHT.h"
#define DHTPIN 2 // what digital pin we're connected to
// Uncomment whatever type you're using!
//#define DHTTYPE DHT11 // DHT 11
#define DHTTYPE DHT22 // DHT 22 (AM2302), AM2321
//#define DHTTYPE DHT21 // DHT 21 (AM2301)
// Connect pin 1 (on the left) of the sensor to +5V
// NOTE: If using a board with 3.3V logic like an Arduino Due
connect pin 1
// to 3.3V instead of 5V!
// Connect pin 2 of the sensor to whatever your DHTPIN is
// Connect pin 4 (on the right) of the sensor to GROUND
// Connect a 10K resistor from pin 2 (data) to pin 1 (power) of the
sensor
// Initialize DHT sensor.
// Note that older versions of this library took an optional third
parameter to
// tweak the timings for faster processors. This parameter is no
longer needed
// as the current DHT reading algorithm adjusts itself to work on
faster procs.
DHT dht(DHTPIN, DHTTYPE);

void setup() {
   Serial.begin(9600);
   Serial.println("DHTxx test!");
   dht.begin();
}

void loop() {
   delay(2000); // Wait a few seconds between measurements
   float h = dht.readHumidity();
   // Reading temperature or humidity takes about 250 milliseconds!
   float t = dht.readTemperature();
   // Read temperature as Celsius (the default)
   float f = dht.readTemperature(true);
   // Read temperature as Fahrenheit (isFahrenheit = true)
   // Check if any reads failed and exit early (to try again).
   if (isnan(h) || isnan(t) || isnan(f)) {
      Serial.println("Failed to read from DHT sensor!");
      return;
   }

   // Compute heat index in Fahrenheit (the default)
```

```
   float hif = dht.computeHeatIndex(f, h);
   // Compute heat index in Celsius (isFahreheit = false)
   float hic = dht.computeHeatIndex(t, h, false);
   Serial.print ("Humidity: ");
   Serial.print (h);
   Serial.print (" %\t");
   Serial.print ("Temperature: ");
   Serial.print (t);
   Serial.print (" *C ");
   Serial.print (f);
   Serial.print (" *F\t");
   Serial.print ("Heat index: ");
   Serial.print (hic);
   Serial.print (" *C ");
   Serial.print (hif);
   Serial.println (" *F");
}
```

Code to Note

DHT22 sensor has four terminals (V_{cc}, DATA, NC, GND), which are connected to the board as follows −

- DATA pin to Arduino pin number 2
- V_{cc} pin to 5 volt of Arduino board
- GND pin to the ground of Arduino board
- We need to connect 10k ohm resistor (pull up resistor) between the DATA and the V_{cc} pin

Once hardware connections are done, you need to add DHT22 library to your Arduino library file as described earlier.

Result

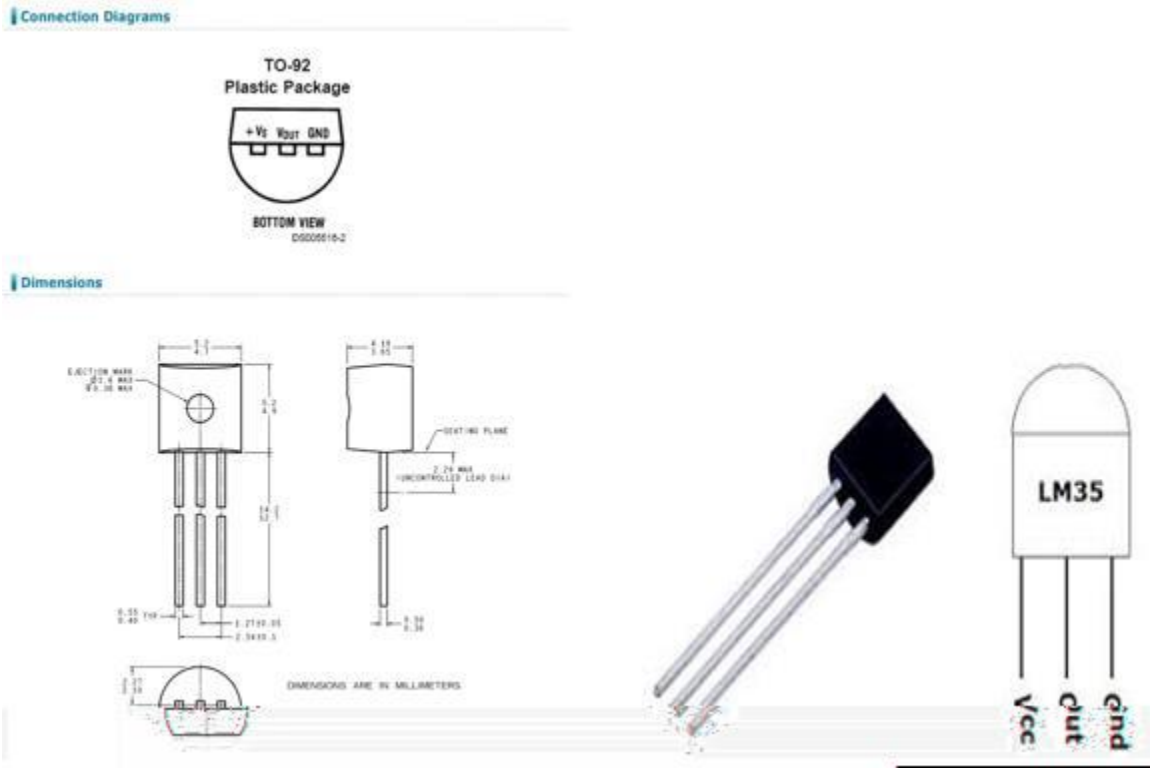
You will see the temperature and humidity display on serial port monitor which is updated every 2 seconds.

EXPERIMENT NO : 03

AIM : Temprature -sensor using Arduino.

The Temperature Sensor LM35 series are precision integrated-circuit temperature devices with an output voltage linearly proportional to the Centigrade temperature.

The LM35 device has an advantage over linear temperature sensors calibrated in Kelvin, as the user is not required to subtract a large constant voltage from the output to obtain convenient Centigrade scaling. The LM35 device does not require any external calibration or trimming to provide typical accuracies of ±¼°C at room temperature and ±¾°C over a full −55°C to 150°C temperature range.



Technical Specifications

- Calibrated directly in Celsius (Centigrade)
- Linear + 10-mV/°C scale factor
- 0.5°C ensured accuracy (at 25°C)
- Rated for full −55°C to 150°C range
- Suitable for remote applications

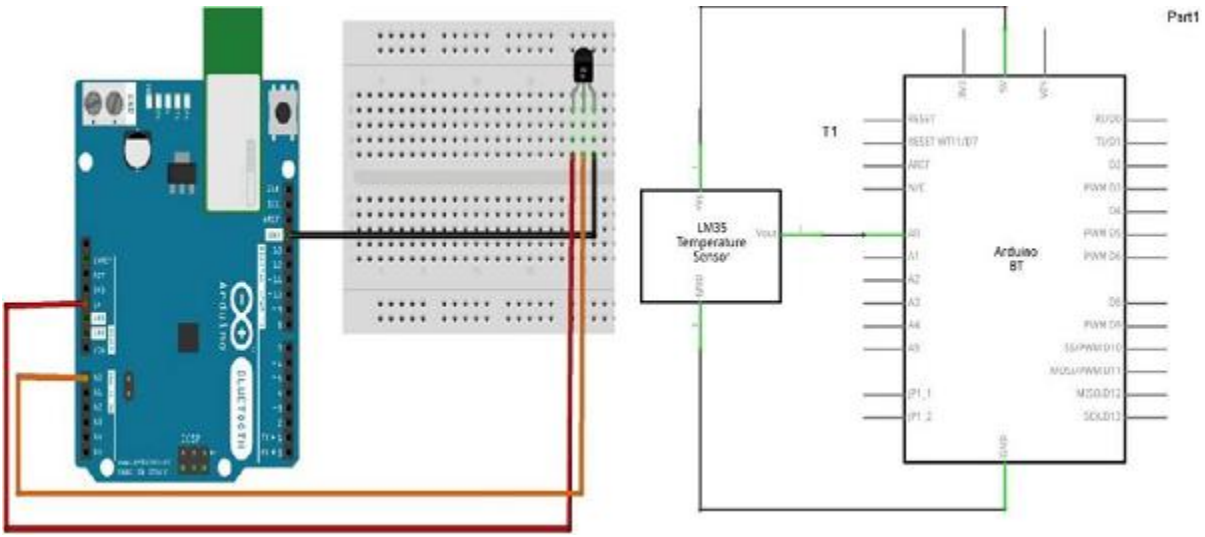
Components Required

You will need the following components −

- 1 × Breadboard
- 1 × Arduino Uno R3
- 1 × LM35 sensor

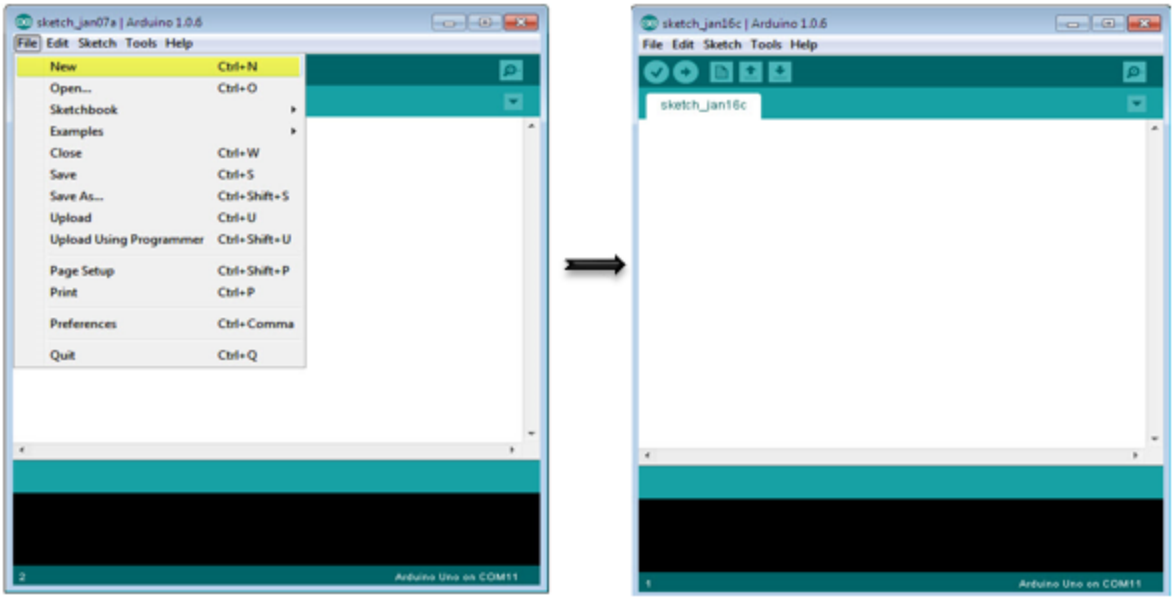
Procedure

Follow the circuit diagram and hook up the components on the breadboard as shown in the image given below.



Sketch

Open the Arduino IDE software on your computer. Coding in the Arduino language will control your circuit. Open a new sketch File by clicking New.



Arduino Code

```
float temp;
int tempPin = 0;

void setup() {
   Serial.begin(9600);
}

void loop() {
   temp = analogRead(tempPin);
   // read analog volt from sensor and save to variable temp
   temp = temp * 0.48828125;
   // convert the analog volt to its temperature equivalent
   Serial.print("TEMPERATURE = ");
   Serial.print(temp); // display temperature value
   Serial.print("*C");
   Serial.println();
   delay(1000); // update sensor reading each one second
}
```

Code to Note

LM35 sensor has three terminals - V_s, V_{out} and GND. We will connect the sensor as follows −

- Connect the $+V_s$ to +5v on your Arduino board.
- Connect V_{out} to Analog0 or A0 on Arduino board.
- Connect GND with GND on Arduino.

The Analog to Digital Converter (ADC) converts analog values into a digital approximation based on the formula ADC Value = sample * 1024 / reference voltage (+5v). So with a +5 volt reference, the digital approximation will be equal to input voltage * 205.

Result

You will see the temperature display on the serial port monitor which is updated every second.