Panipat Institute of Engineering & Technology,

Samalkha, Panipat

Department of Computer Applications

Lab Manual of Internet of Things

Subject Code – BCA-CTIS-409

Submitted To:

Mr. Dishu

Assistant Professor (DCA)

Submitted By:

Sundram Kumar Singh

BCA-CTIS-4th Sem

220187575

Affiliated to

Kurukshetra University Kurukshetra, India

PRACTICAL:- 1

AIM:- Describe the Arduino board.

Arduino Boards:-

Arduino is an easy-to-use open platform to create electronics projects. Arduino boards play a vital role in creating different projects. It makes electronics accessible to non-engineers, hobbyists, etc.

The various components present on the Arduino boards are Microcontroller, Digital Input/output pins, USB Interface and Connector, Analog Pins, Reset Button, Power button, LED's, Crystal Oscillator, and Voltage Regulator. Some components may differ depending on the type of board.The most standard and popular board used over time is Arduino UNO. The ATmega328 Microcontroller present on the UNO board makes it rather powerful than other boards. There arevarious types of Arduino boards used for different purposes and projects. The Arduino Boards are organized using the Arduino (IDE), which can run on various platforms. Here, IDE stands forIntegrated Development Environment.

Let's discuss some common and best Arduino boards

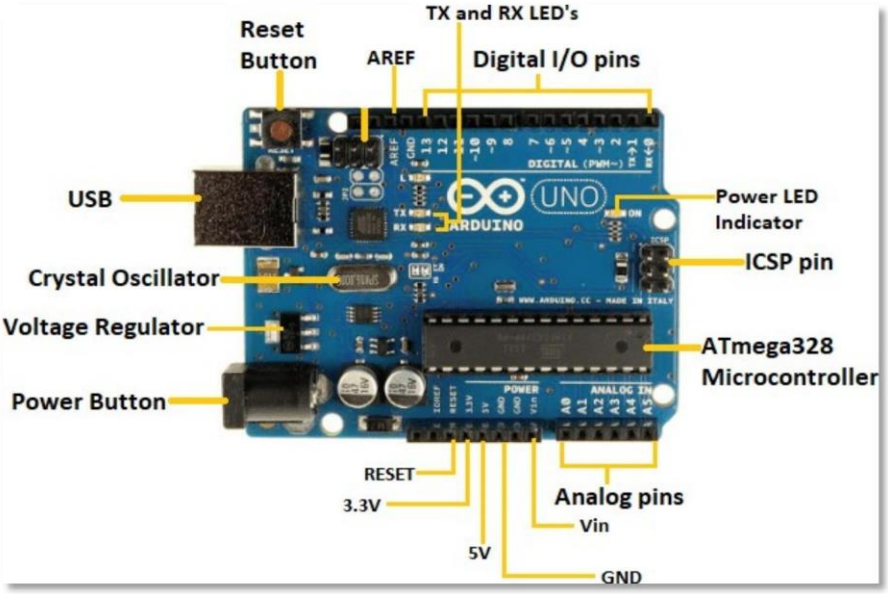

Figure 1: Arduino Board

Types of Arduino Boards

- Arduino UNO

Arduino UNO is based on an ATmega328P microcontroller. It is easy to use compared to other boards, such as the Arduino Mega board, etc. The Arduino UNO includes 6 analog pin inputs, 14 digital pins, a USB connector, a power jack, and an ICSP (In-Circuit Serial Programming) header.It is the most used and of standard form from the list of all available Arduino Boards. It is also recommended for beginners as it is easy to use.

- Arduino Nano

The Arduino Nano is a small Arduino board based on ATmega328P or ATmega628 Microcontroller. The connectivity is the same as the Arduino UNO board

The Nano board is defined as a sustainable, small, consistent, and flexible microcontroller board. It is small in size compared to the UNO board. The devices required to start our projects using the Arduino Nano board are Arduino IDE and mini USB.The Arduino Nano includes an I/O pin set of 14 digital pins and 8 analog pins. It also includes 6 Power pins and 2 Reset pins.

- Arduino Mega

The Arduino Mega is based on ATmega2560 Microcontroller. The ATmega2560 is an 8-bit microcontroller. We need a simple USB cable to connect to the computer and the AC to DC adapter or battery to get started with it. It has the advantage of working with more memory space.

The Arduino Mega includes 54 I/O digital pins and 16 Analog Input/Output (I/O), ICSP header, a reset button, 4 UART (Universal Asynchronous Reciever/Transmitter) ports, USB connection, and a power jack.

- Arduino Micro

The Arduino Micro is based on the ATmega32U4 Microcontroller. It consists of 20 sets of pins. The 7 pins from the set are PWM (Pulse Width Modulation) pins, while 12 pins are analog input pins. The other components on board are reset button, 16MHz crystal oscillator, ICSP header, and a micro USB connection.

The USB is inbuilt in the Arduino Micro board.

- Arduino Leonardo

The basic specification of the Arduino Leonardo is the same as the Arduino Micro. It is also based on ATmega32U4 Microcontroller. The components present on the board are 20 analog and digital pins, reset button, 16MHz crystal oscillator, ICSP header, and a micro USB connection.

PRACTICAL:- 2

Aim:- Describe the Analog and Digital pins of Arduino board.

Analog Pin:-

The analog pins in the Arduino Uno serve as analog-to-digital converters. They can read the voltage on a pin and convert it to a digital value that can be used in your sketches. The analog pins the Arduino Uno are labeled A0 through A5, and each one can read a voltage between 0 and 5 volts with a resolution of 10 bits, which means it can distinguish between 1024 (2^10) different levels of voltage.

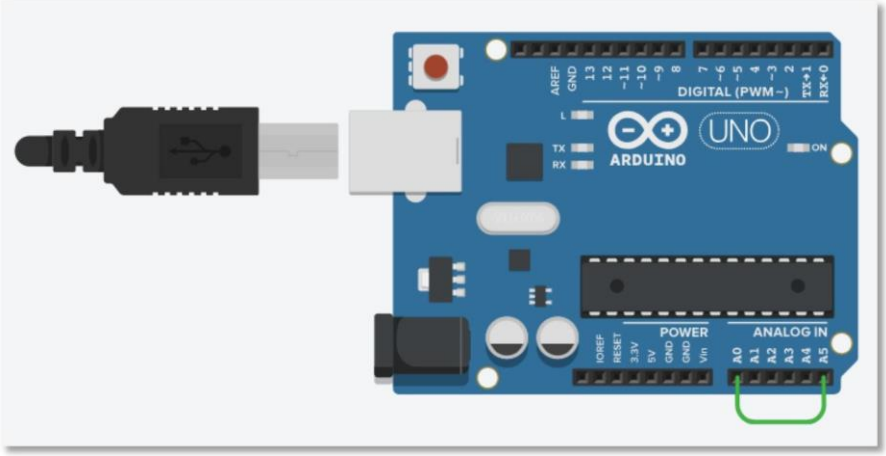

Figure 2 : Pins A0-A5

This capability makes the analog pins very useful for reading sensors that provide an analog voltage output, such as light sensors, temperature sensors, and potentiometers. You can use the analogRead() function in your sketches to read the voltage on an analog pin and convert it to a digital value that you can then use in your program.

Digital Pin:-

Arduino UNO is a microcontroller board based on the ATmega328P (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator (CSTCE16M0V53-R0), a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.. You can tinker with your UNO without worring too much about doing something wrong, worst case scenario you can replace the chip for a few dollars and start over again.

The UNO has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function.

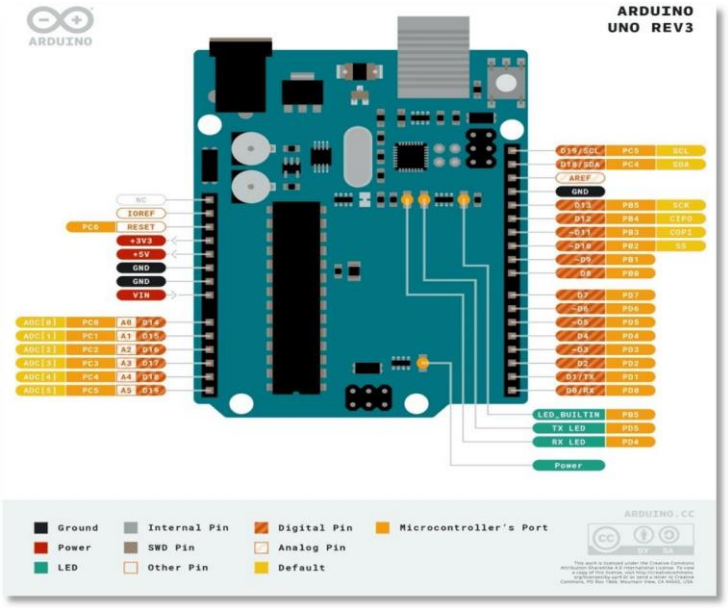

Figure 3: Digital Pin

Each of the 14 digital pins on the UNO can be used as an input or output, using pinMode(),digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50k ohm. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller.

PRACTICAL:-3

AIM:- How to blink an LED light using an Arduino board.

Blinking an LED is an introductory Arduino project in which we control an LED using Arduino. LED blinking refers to the process of continuously turning an LED (Light Emitting Diode) and off in a repetitive pattern. It is a simple and common demonstration in electronics and microcontroller-based projects.

Using Arduino, Light emitting diodes (LED's) are handy. For that, connect a wire to digital pin 13 on the Arduino board, GND wire and VIN pin a voltage of 5v-9v. We get a constant or blinking LED flushing as we need.

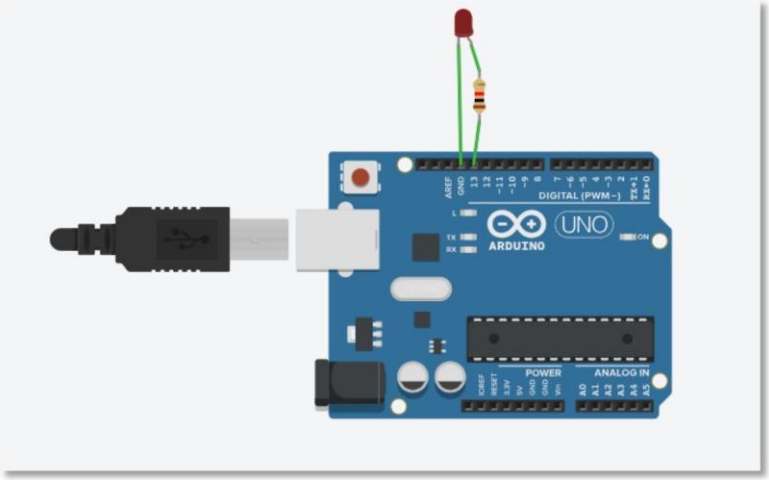

Figure 4: Arduino Board with LED

Code:-

```
/*
  This program blinks pin 13 of the Arduino (the
  built-in LED)
*/
```

This first section is title block comment, describing what the program does. Block comments are bookended by an opening /* and closing */.

```
void setup()
{
  pinMode(13, OUTPUT);
```

```
}
```

Next is the code's setup, which helps set up things your program will need later. It runs once when the program starts up, and contains everything within its curly braces { }. Our blink sketch's setup configures pin 13 as an output, which prepares the board to send signals to it, rather than listen.

```
void loop()
{
  // turn the LED on (HIGH is the voltage level)
  digitalWrite(13, HIGH);
  delay(1000); // Wait for 1000 millisecond(s)
  // turn the LED off by making the voltage LOW
  digitalWrite(13, LOW);
  delay(1000); // Wait for 1000 millisecond(s)
}
```

Output:

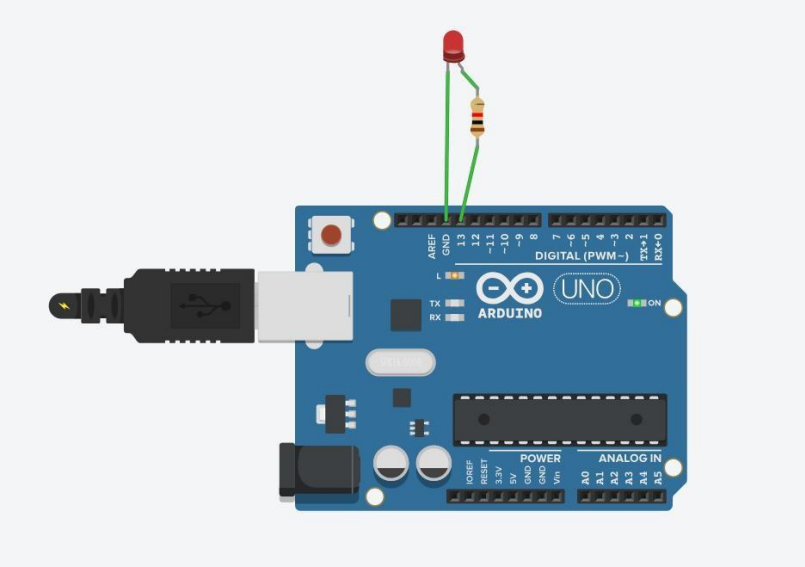

Figure 5: Arduino Board with LED

PRACTICAL:-4

AIM:- Describe the breadboard with diagram.

The breadboard is a white rectangular board with small embedded holes to insert electronic components. It is commonly used in electronics projects. We can also say that breadboard is a prototype that acts as a construction base of electronics.

The breadboard is shown in the below image:

Breadboard:-

A breadboard is derived from two words bread and board. The word breadboard was initially used to slice the bread pieces. But, it was further named as a breadboard for its use in electronics around the 1970s. Hence, the term breadboard refers to these boards only and provides a quick electrical connection.

A breadboard is also categorized as a Solderless board. It means that the component does not require any soldering to fit into the board. Thus, we can say that breadboard can be reused. We can easily fit the components by plugging their end terminal into the board. Hence, a breadboard is often called a plugboard.

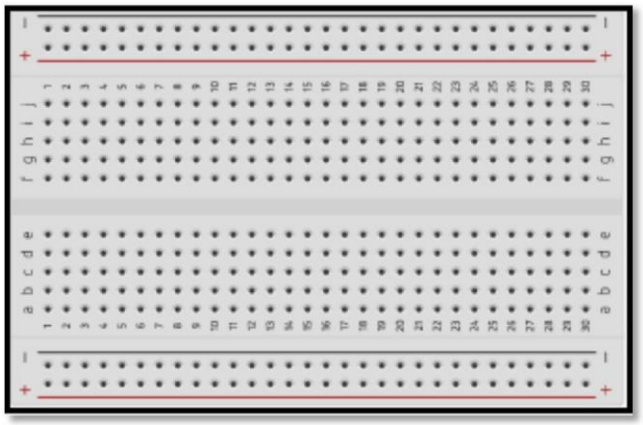

Figure 6: Breadboard

Materials used:

Let's discuss the materials used for creating the breadboard.

White plastic is the material that is used to create breadboards. Today, most of the breadboards are Solderless breadboards. We can directly plug in the electronic components and connect it with the external power supply.

The breadboards are available as per the specified point holes. For example, 400 point

breadboard, 830 point breadboard, etc.

Ques: How breadboard enables the connection when the leads of different components are plugged in?

There are three parts in a breadboard, as shown below

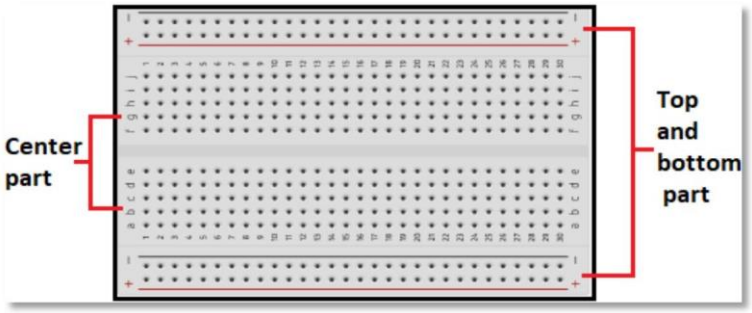

Figure 7: Breadboard Parts

Breadboard

The top and bottom holes of a row in a breadboard are connected horizontally, and the center part is connected vertically, as shown below:

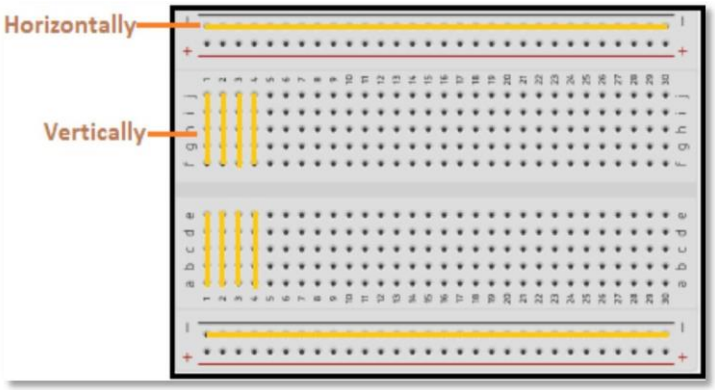

Figure 8: Vertical and Horizontal

Breadboard:-

It means a single horizontal line of a breadboard has the same connection. It is because the metal strip underneath the breadboard at the top and bottom are connected horizontally. Hence, it provides the same connection in a row. The two top and bottom parts of a breadboard are generally used for power connections.

Advantages of Breadboard:-

The advantages of using breadboard are listed below:

- Temporary prototype

We can build a temporary prototype for the projects with the help of a breadboard.

- Reusable

Today, Solderless boards are mostly used in various applications. It does not require any soldering to fix the components. Hence, it can be reused.

- Lightweight

The breadboard is made of white plastic, which is light in weight.

- Easy experimentation

We can quickly insert the leads of the components into the tiny holes of the breadboard. The circuit can be created using various components and circuit design.

- Inexpensive

The breadboards are easily available. It also cost less.

- Easy to use

It does not involve any complex parts. We can easily insert the required number of components.

The hoes are already embedded in the board. Hence, we do not require any drilling to insert the electronic components.

Disadvantages of Breadboard:-

The disadvantages or limitations of the breadboard are listed below:

- Not suited for high current applications
- Low-frequency Solderless boards are limited to low-frequency applications.
- Requires more physical space for simple circuits.
- A high number of connections in the Solderless board make the circuit messy due to a greater number of wires.
- The circuit design does not work well for high-speed design.
- The plugging and unplugging can disturb the other connections.
- Less reliable connections.
- Limited signaling.

PRACTICAL:- 5

Aim:- How to setup an LED light show on tinkercad.

We have created a simulation using a circuit that is joined with multiple LEDs, Breadboard and Arduino. This project will show that how can we join multiple LEDs and glow them by connecting them to Arduino and breadboard. So you get the question of how it is working? Let us discuss basic components to help this simulation to complete.

Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards can read inputs — light on a sensor, a finger on a button, or a Twitter message — and turn it into an output — activating a motor, turning on an LED, publishing something online.

A breadboard is a rectangular board with many mounting holes. They are used for creating electrical connections between electronic components and single-board computers or microcontrollers such as Arduino and Raspberry Pi. The connections aren't permanent and they can be removed and placed again.

The circuit consists of numerous LEDs, resistors, wires, an Arduino, and a breadboard. We've simply connected wires from the Arduino's 5V and GND pins to the anode(-) and cathode(+). The positive and negative terminal ends of the breadboard are linked to the positive and negative terminals, respectively. After that, we simply linked the LEDs in a circuit.Finally, we connected wires from the breadboard to the Arduino's 13V, 12V, and -11V points. Now that the circuit is functional, we can begin the simulation.

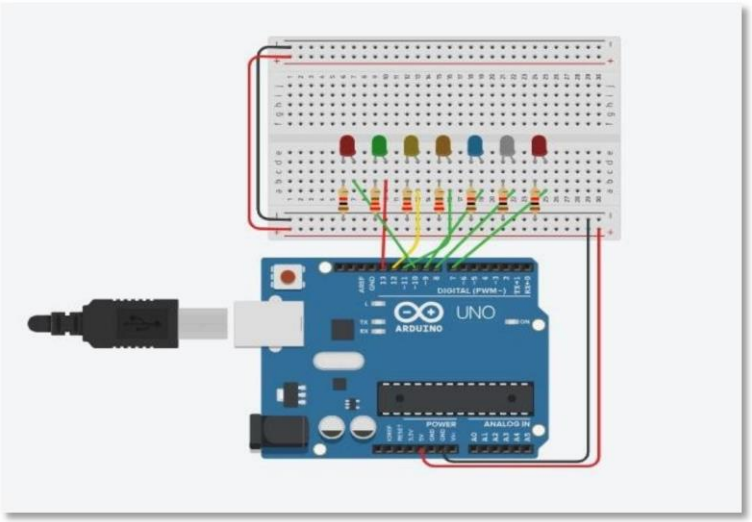

Figure 9: Glowing LED using Breadboard

Code:-

```
int animationSpeed = 0;
void setup()
{
  pinMode(LED_BUILTIN, OUTPUT);
  pinMode(12,
  OUTPUT);
  pinMode(11,
  OUTPUT);
  pinMode(10,
  OUTPUT);
  pinMode(9,
  OUTPUT);
  pinMode(8,
  OUTPUT);
  pinMode(7,
  OUTPUT);
}
void loop()
{
  animationSpeed=400;
  digitalWrite(LED_BUILTINHIGH;
  delay(animationSpeed);
  digitalWrite(12,HIGH);
  delay(animationSpeed);
  digitalWrite(11,HIGH);
  delay(animationSpeed);
  digitalWrite(10,HIGH);
  delay(animationSpeed);
```

```
digitalWrite(9,HIGH);
delay(animationSpeed);
digitalWrite(8,HIGH);
delay(animationSpeed);
digitalWrite(7, HIGH);
delay(animationSpeed);
animationSpeed=0;
digitalWrite(LED_BUILTIN,LOW;
delay(animationSpeed);
digitalWrite(12,LOW);
delay(animationSpeed);
digitalWrite(11,LOW);
delay(animationSpeed);
digitalWrite(10,LOW);
delay(animationSpeed);
digitalWrite(9,LOW);
delay(animationSpeed);
digitalWrite(8,LOW);
delay(animationSpeed);
digitalWrite(7,LOW);
delay(animationSpeed);
```

Output:-

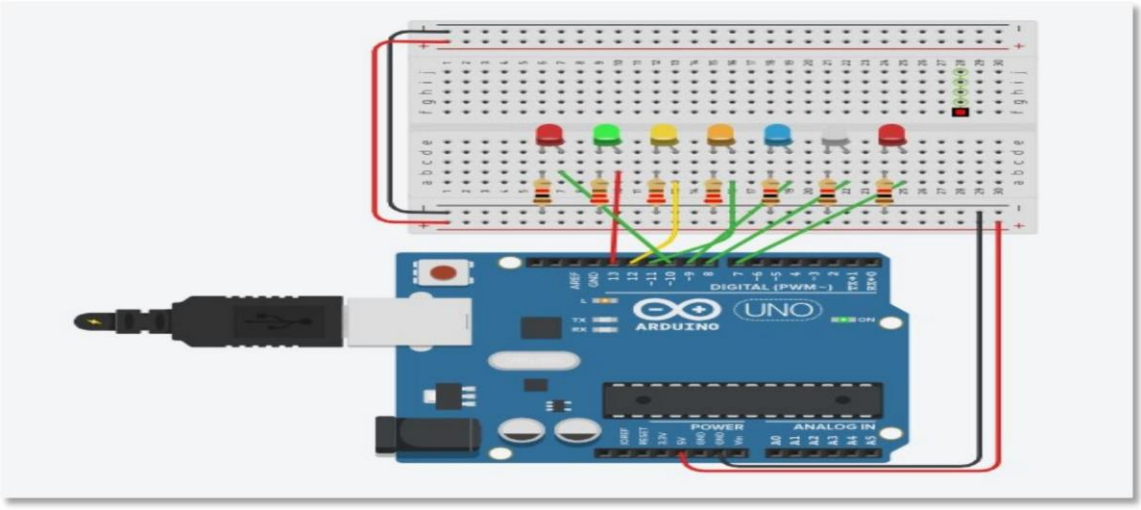

Figure 10: Glowing LED using Breadboard

PRACTICAL :- 6

AIM:- Create a RGB LED color mixing circuit with Arduino.

We'll connect an RGB LED to the Arduino Uno and compose a simple program to change its color. Here in Tinkercad Circuits, you can explore the sample circuit (click Start Simulation towatch the LED change color) and build your own right next to it. Optionally grab your electronics supplies and build along with a physical Arduino Uno, USB cable, breadboard, RGB LED, resistors (any value from 100-1K ohms will do), and some breadboard wires.

1. Try it for yourself! Add a new Arduino and breadboard along side the sample, and prep your breadboard by connecting Arduino 5V to the power rail and Arduino GND to the ground rail.
2. Add an RGB LED from the components panel and place it across four different rows of the breadboard.
3. The RGB LED in the simulator has a common cathode (negative, ground) at its second leg, so wire this row/pin to ground.
4. Add three resistors (drag all three or create one and then copy/paste) and move them to the breadboard rows for the remaining three LED pins, bridging across the breadboard's center gap to three separate rows on the other side.

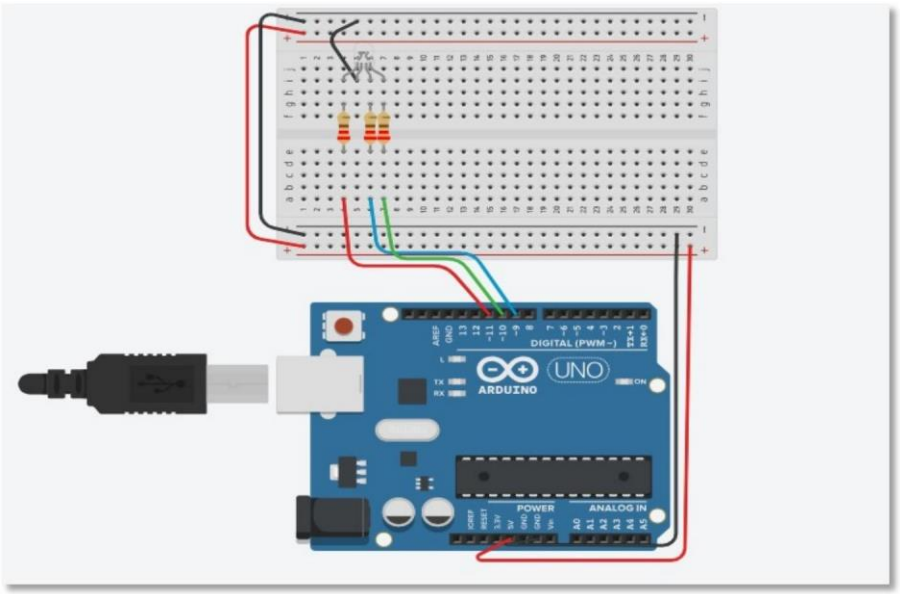

Figure 11: RGB LED using Breadboard

Code:-

```
void setup()
{

pinMode(11,   OUTPUT);
pinMode(10,   OUTPUT);
pinMode(9, OUTPUT);
}

void loop()

{
analogWrite(11, 255);

analogWrite(10, 255);

analogWrite(9, 0);
delay(1000);
analogWrite(11, 255);

analogWrite(10, 255);

analogWrite(9, 102);
delay(1000);
analogWrite(11, 192);
analogWrite(10, 192);

analogWrite(9, 192);
delay(1000);
  analogWrite(11, 0);

analogWrite(10, 128);
analogWrite(9, 128);
delay(1000);
  analogWrite(11, 128);
analogWrite(10, 0);

analogWrite(9, 128);
```

```
delay(1000);

  analogWrite(11, 128);
analogWrite(10, 0);

analogWrite(9, 0);
delay(1000);
}
```

Output:-

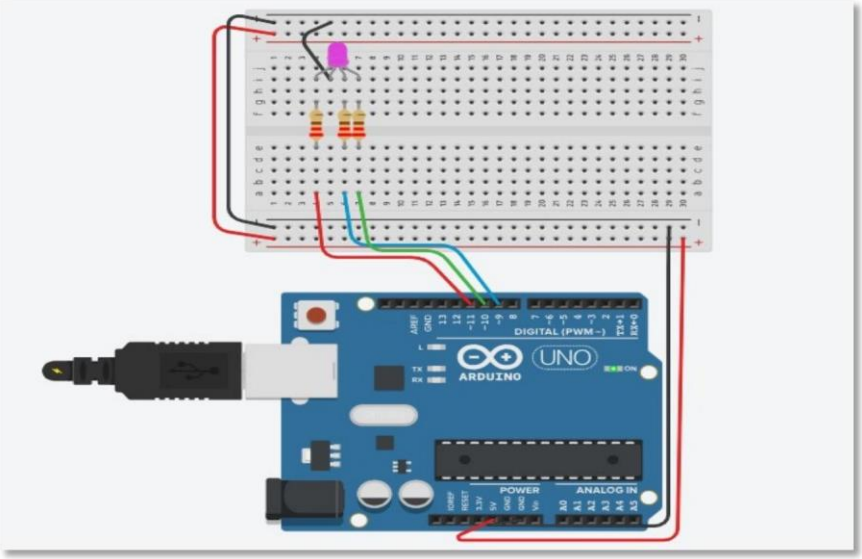

Figure 12: RGB LED using Breadboard

PRACTICAL:- 7

AIM:- Describe LCD. Print your name on the LCD with Arduino.

LCD is a flat display technology, stands for "Liquid Crystal Display," which is generally used in computer monitors, instrument panels, cell phones, digital cameras, TVs, laptops, tablets, and calculators. It is a thin display device that offers support for large resolutions and better picture quality. The older CRT display technology has replaced by LCDs, and new display technologies like OLEDs have started to replace LCDs. An LCD display is most commonly found with Dell laptop computers and is available as an active-matrix, passive-matrix, or dual-scan display. The picture is an example of an LCD computer monitor.

As compared to CRT technology, LCD consumed much less power and allowed displays to be much thinner that also made them very less heavy. Instead of emitting light, LCDs work on the principle of blocking light. In an LCD, where an LED ejects light, the liquid crystals produce a picture with the help of using a backlight. Also, while comparing with gas-display and LED displays, LCDs consume less energy.

The CRT monitors and TVs have a refresh rate, but LCD screens do not have a refresh rate.

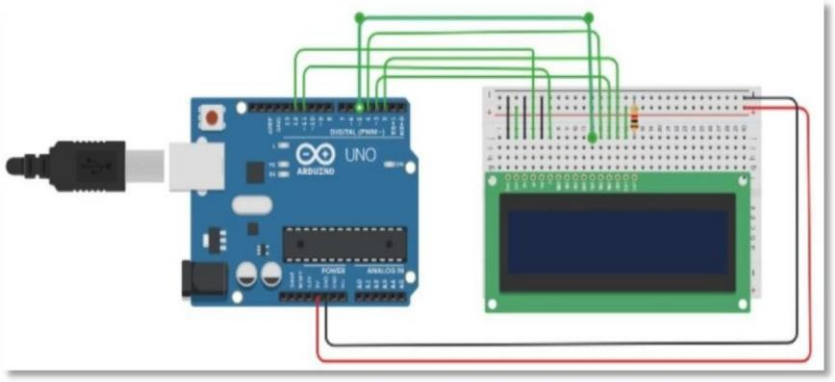

Figure 13:LCD with Arduino

Code:-

```
//Display name in LCD
#include<LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5,4,3,2);
void setup()
{
 lcd.begin(16, 2);
```

```
}
void loop()
{
  lcd.setCursor(0,1);
  lcd.print("Kartik");
   }
```

Output:-

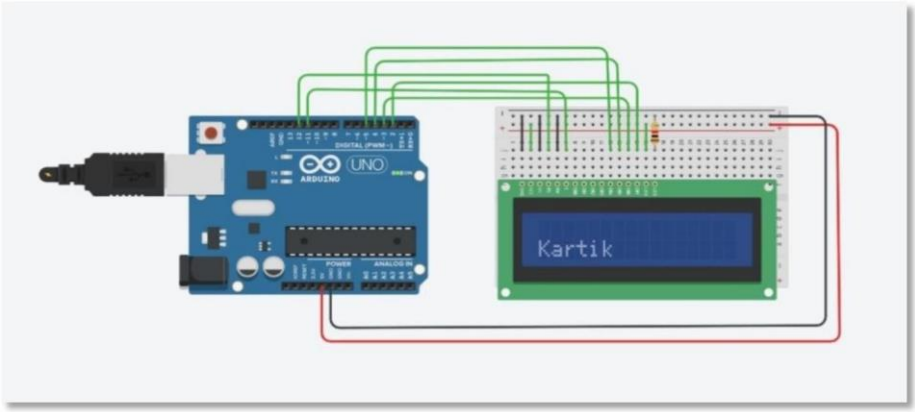

Figure 14: Dispalying name on LCD with Arduino

PRACTICAL:- 8

AIM:- Measure temperature using Arduiono.

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.

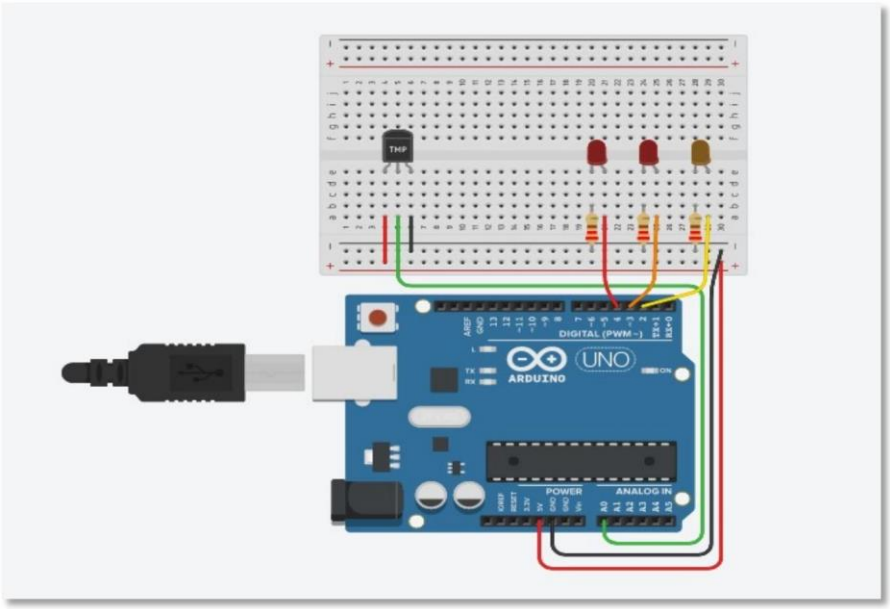

Figure 15: Measuring Temperature

CODE:-

```
int baselineTemp =
0;int celsius = 0;
int  fahrenheit  =
0;void setup()
{


 pinMode(A0,
 INPUT);
 Serial.begin(9600);
 pinMode(2,
```

```
 OUTPUT);
 pinMode(3,
 OUTPUT);
 pinMode(4,
 OUTPUT);
}


void loop()
{
 baselineTemp = 10;

 celsius = map(((analogRead(A0) - 20) * 3.04), 0, 1023, -40, 125);

 fahrenheit = ((celsius * 9) / 5 +
 32);Serial.print(celsius);
 Serial.print(" C, ");
 Serial.print(fahrenheit)
 ;Serial.println(" F");
 if (celsius < baselineTemp)
  {  digitalWrite(2,  LOW);
  digitalWrite(3,     LOW);
  digitalWrite(4, LOW);
 }

 if (celsius >= baselineTemp && celsius < baselineTemp +
  10) {digitalWrite(2, HIGH);
  digitalWrite(3,
  LOW);
  digitalWrite(4,
  LOW);
 }

 if (celsius >= baselineTemp + 10 && celsius < baselineTemp +
  20) {digitalWrite(2, HIGH);
```

```
    digitalWrite(3,
    HIGH);
    digitalWrite(4,
    LOW);
   }

   if (celsius >= baselineTemp + 20 && celsius < baselineTemp +
    30) {digitalWrite(2, HIGH);
    digitalWrite(3,
    HIGH);
    digitalWrite(4,
    HIGH);
   }

   if (celsius >= baselineTemp + 30)
{digitalWrite(2, HIGH);
digitalWrite(3,   HIGH);
digitalWrite(4, HIGH);
   }
   delay(1000); // Wait for 1000 millisecond(s)}
```

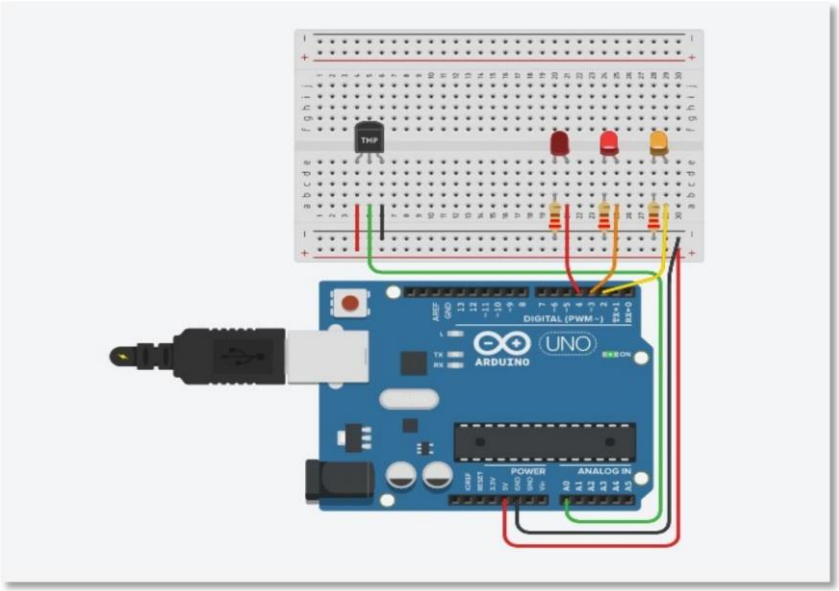

Figure 16: Measuring Temperature

PRACTICAL:-9

AIM:- How to fade LEDs with Analog output.

The LED is connected in series with a resistor between Arduino pin 9 and ground. Remember that the solderless breadboard rows are connected inside, so you can plug in components and wires to make quick temporary connections.

You'll see the following connections:

Breadboard power (+) and ground (-) rails to Arduino 5V and ground (GND), respectively

LED cathode (negative, shorter leg) to one leg of a resistor (anywhere from 100-1K ohms is fine)

Other resistor leg to ground

LED anode (positive, longer leg) to Arduino pin 9.

Inside the counting loop, add an output block to set one of the special pins, and adjust it to pin 9. Navigate to Variables and drag the brightness block to the output block to set pin 9 to the current value of brightness, which changes over the course of the counting loop.

Add a wait block, and set it to 30 milliseconds. The duration of this block can bechanged to slow down or speed up the fading effect.

The counting loop you created fades the LED from off to all the way on. To fade the LED back off again, we have to create another counting loop. Either drag a new counting loop into the editor, or duplicate this one, and this time change it to count down, start with 255, and go down to zero.

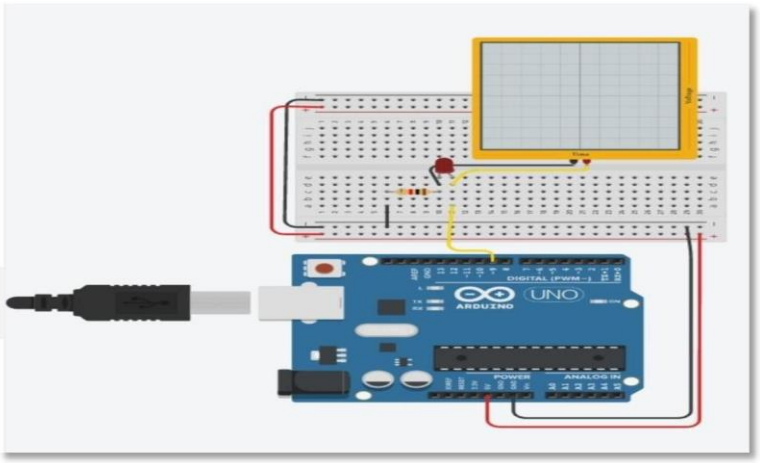

Figure 17: Fade LEDs

Code:-

```
int brightness = 0;
void setup()
{
  pinMode(9, OUTPUT);
}
```

The main body of the program starts out by creating a variable called brightness and sets it equal to zero, then inside the setup() pin 9 is initialized as an output.

```
void loop()
{
  for (brightness = 0; brightness <= 255; brightness += 5)
    {analogWrite(9,  brightness);
    delay(30); // Wait for 30 millisecond(s)
  }
  for (brightness = 255; brightness >= 0; brightness -= 5)
    {analogWrite(9,  brightness);
    delay(30); // Wait for 30 millisecond(s)
  }}
```

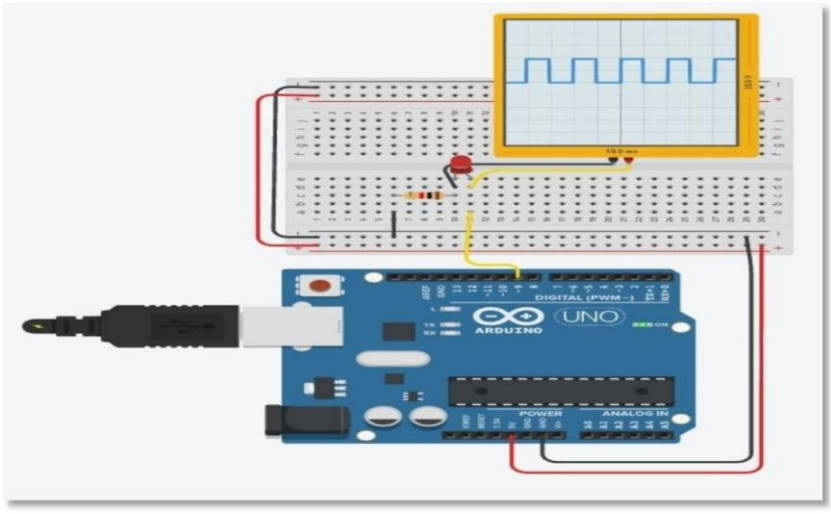

Figure 18: Fade LEDs

PRACTICAL:- 10

AIM:- Measure the distance of an object using an ultrasonic sensor and Aurdino.

An ultrasonic Sensor is a device used to measure the distance between the sensor and an object without physical contact. This device works based on time-to-distance conversion.Ultrasonic sensors measure distance by sending and receiving the ultrasonic wave. The ultrasonic sensor has a sender to emit the ultrasonic waves and a receiver to receive the ultrasonic waves. The transmitted ultrasonic wave travels through the air and is reflected by hitting the Object. Arduino calculates the time taken by the ultrasonic pulse wave to reach the receiver from the sender. We know that the speed of sound in air is nearly 344 m/s,So, the known parameters are time and speed (constant). Using these parameters, we can calculate the distance traveled by the sound wave.

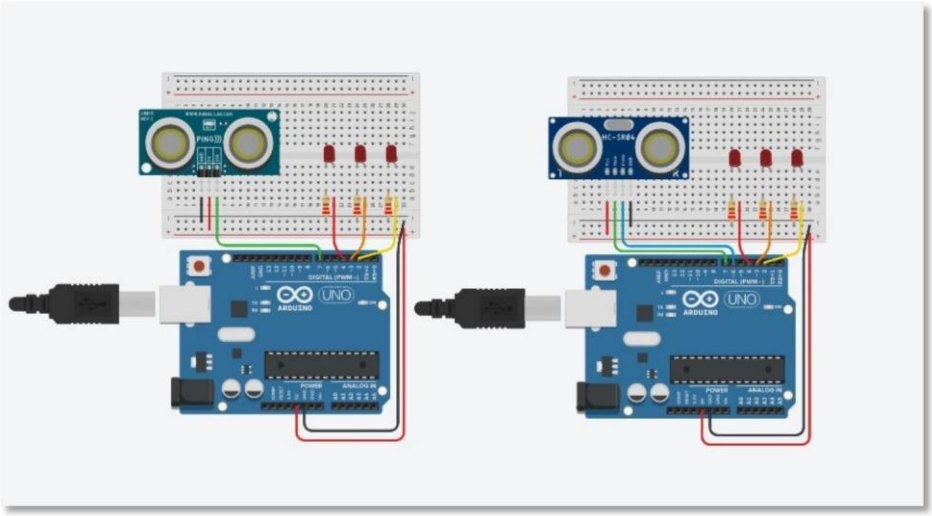

Figure 19: Distance Finder

Code:-

```
int distanceThreshold =
0;int cm = 0;
int inches = 0;
```

```
long readUltrasonicDistance(int triggerPin, int echoPin)

{
 pinMode(triggerPin, OUTPUT); // Clear the
 trigger digitalWrite(triggerPin, LOW);
 delayMicroseconds(2);
 // Sets the trigger pin to HIGH state for 10
 microseconds digitalWrite(triggerPin, HIGH);
 delayMicroseconds(10);
 digitalWrite(triggerPin,
 LOW); pinMode(echoPin,
 INPUT);
 // Reads the echo pin, and returns the sound wave travel time in
 microsecondsreturn pulseIn(echoPin, HIGH);
}

void setup()

{

 Serial.begin(9600);
 pinMode(2,
 OUTPUT);
 pinMode(3,
 OUTPUT);
 pinMode(4,
 OUTPUT);
}

void loop()
```

```
{

 // set threshold distance to activate LEDs
 distanceThreshold = 350;
 // measure the ping time in cm

 cm = 0.01723 * readUltrasonicDistance(7, 6);

 // convert to inches by dividing by
 2.54inches = (cm / 2.54);

 Serial.print("cm,
 ");

 Serial.print(inches)
 ;

 Serial.println("in")
 ;

 if (cm > distanceThreshold)
   {  digitalWrite(2,  LOW);
  digitalWrite(3,      LOW);
  digitalWrite(4, LOW);
 }

 if (cm <= distanceThreshold && cm > distanceThreshold -
  100) {digitalWrite(2, HIGH);
  digitalWrite(3,
  LOW);
  digitalWrite(4,
  LOW);
```

```
 }

 if (cm <= distanceThreshold - 100 && cm > distanceThreshold - 250)
  {digitalWrite(2, HIGH);
  digitalWrite(3,
  HIGH);
  digitalWrite(4,
  LOW);
 }

 if (cm <= distanceThreshold - 250 && cm > distanceThreshold - 350)
  {digitalWrite(2, HIGH);
  digitalWrite(3,
  HIGH);
  digitalWrite(4,
  HIGH);
 }

 if (cm <= distanceThreshold -
  350) {digitalWrite(2, HIGH);

  digitalWrite(3,
  HIGH);
  digitalWrite(4,
  HIGH);
 }

 delay(100); // Wait for 100 millisecond(s)

}
```

Output:-

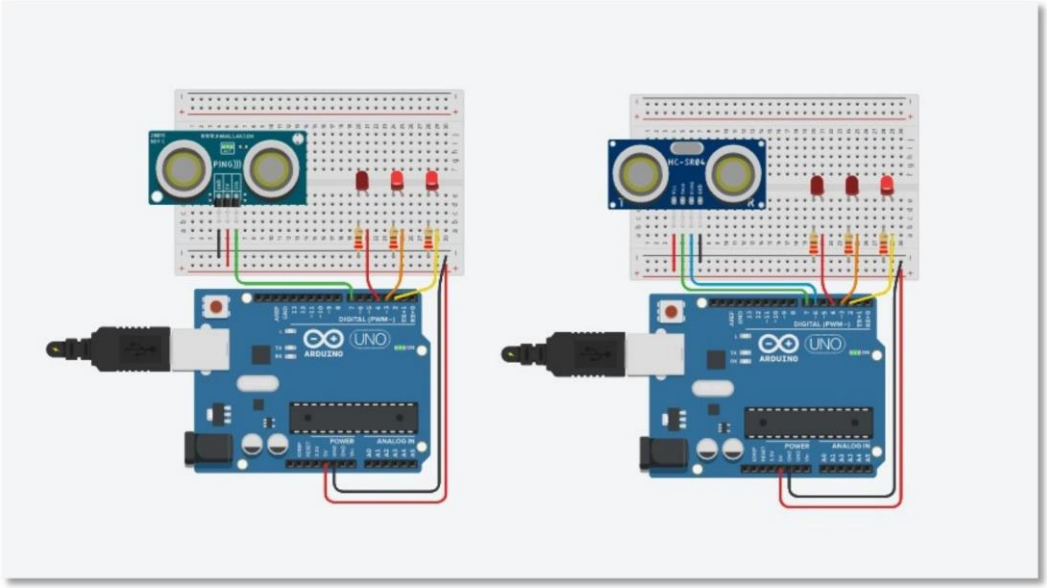

Figure 20: Distance Measuring