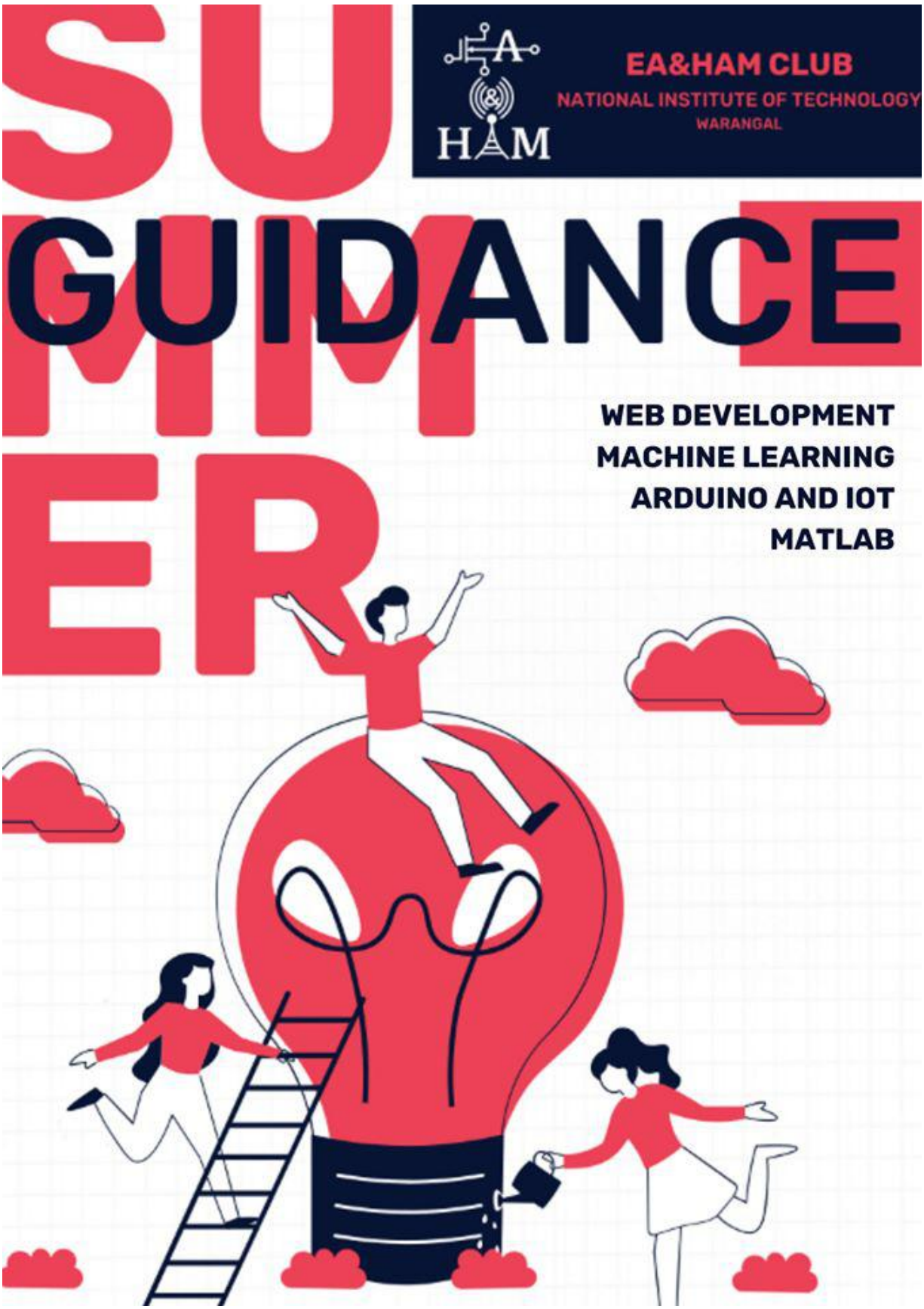
SUMMER GUIDANCE

ARDUINO AND IOT

WEEK 2



L

SUMMER GUIDANCE

ARDUINO AND IOT

WEEK 2

Hello guys I am sure you are following up on our course material this week we are planning to cover:

1) Sensors : various types and examples

2) Servo Motor

3) 7 -segment LED Display

SENSORS

Sensor as an input device which provides an output (signal) with respect to a specific physical quantity (input).The term “input device” in the definition of a Sensor means that it is part of a bigger system which provides input to a main control system (like a Processor or a Micro-controller).

Without any further due lets begin with the types of sensors used.

DR: LIGHT DEPENDENT RESISTOR

SUMMER GUIDANCE

ARDUINO AND IOT

WEEK 2

A photo-resistor or light-dependent resistor (LDR) or photocell is a light-controlled variable resistor.
The resistance of a photo-resistor decreases with increasing incident light intensity; in other words, it exhibits photoconductivity.
A photo-resistor can be applied in light-sensitive detector circuits, and light-and dark-activated switching circuits.

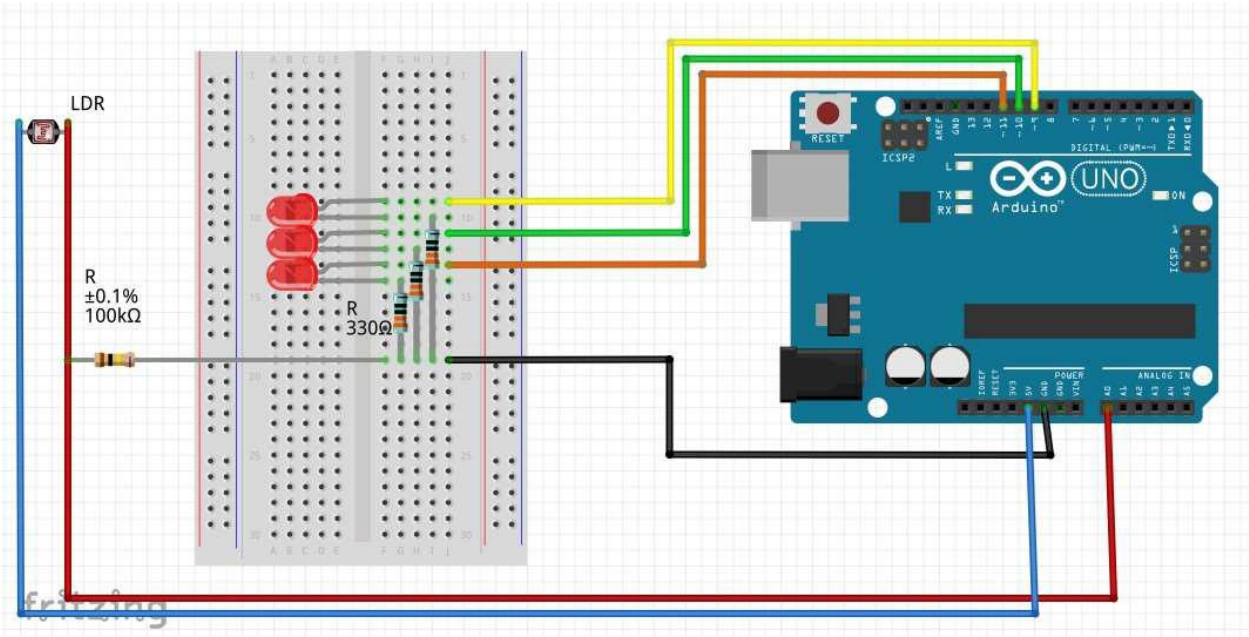
SUMMER GUIDANCE

ARDUINO AND IOT

WEEK 2

LET'S ADD LDR TO CIRCUIT

3 LED system with LDR



Read the LDR VALUE and DISPLAY it on SERIAL MONITOR

CONTROL the circuit using LDR

UPLOAD THE CODE

A simple program to control a 3 LED system using LDR

```
int pin1 = 9;  //declarations and initializations of led and ldr pins
int pin2 = 10;
int pin3 = 11;
int ldr_pin = A0;
int pin;
void setup() {
// put your setup code here, to run once:
  Serial.begin(9600);   //to start serial monitor with value 9600
  pinMode(pin1, OUTPUT); //output at pin1
  pinMode(pin2, OUTPUT);
  pinMode(pin3, OUTPUT);
  pinMode(ldr_pin, INPUT); // input from ldr
}

void loop() {
// put your main code here, to run repeatedly:
  pin = analogRead(ldr_pin); //reading input from ldr
  Serial.println(pin); //to print value on serial monitor

  if(pin >= 500)
  {
    digitalWrite(pin1, HIGH); //function to set pin1 as high
    digitalWrite(pin2, LOW);
    digitalWrite(pin3, LOW);
    delay(100);
    digitalWrite(pin1, LOW);
```

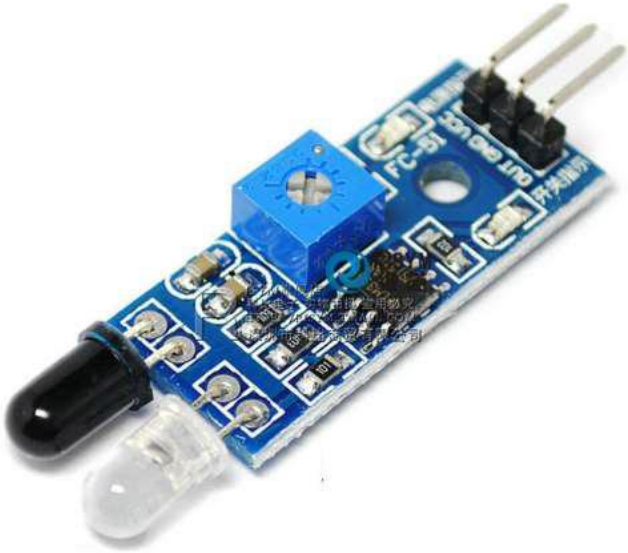
```
    digitalWrite(pin2, HIGH);
    digitalWrite(pin3, LOW);
    delay(100);
    digitalWrite(pin1, LOW);
    digitalWrite(pin2, LOW);
    digitalWrite(pin3, HIGH);
    delay(100);
  }
  else
  {
    digitalWrite(pin1, LOW);
    digitalWrite(pin2, LOW);
    digitalWrite(pin3, LOW);
  }
}
```

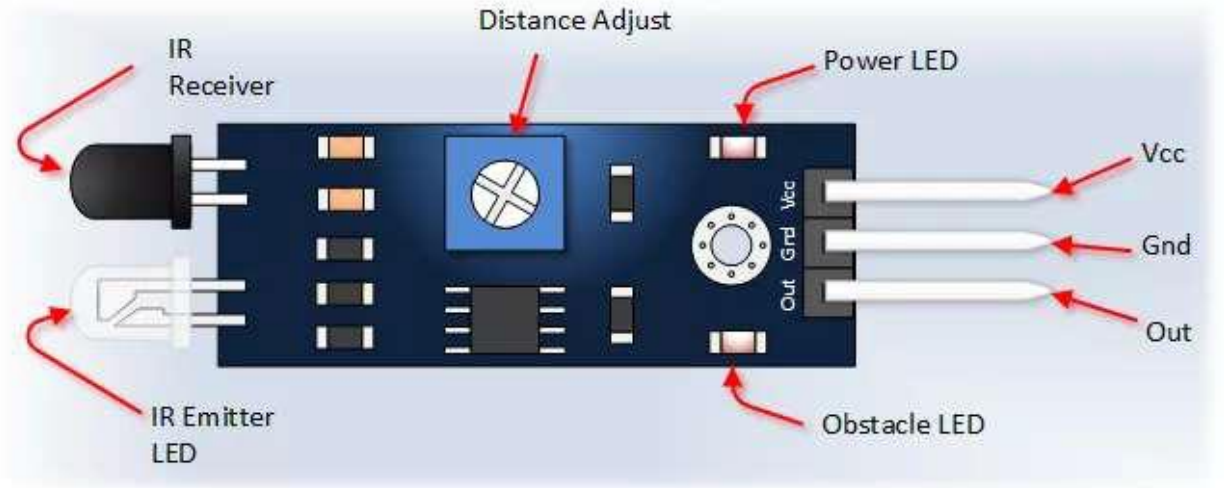
SUMMER GUIDANCE

ARDUINO AND IOT

WEEK 2

INFRARED (IR) SENSOR





SUMMER GUIDANCE

ARDUINO AND IOT

WEEK 2

WHAT IS IR SENSOR?

This is a multipurpose infrared sensor which can be used for color detection. The sensor provides a digital as well as analog output. An on board LED is used to indicate the presence of an object. This digital output can be directly connected to an Arduino, Raspberry Pi or any other microcontroller to read the sensor output.

IR sensors are highly susceptible to ambient light and the IR sensor on this sensor is suitably covered to reduce effect of ambient light on the sensor. The on board potentiometer should be used to calibrate the sensor. Typical range varies from 20 cm to 150 cm.

An infrared light emitting diode (IR LED) emits light of Infrared range 700 nanometers (nm) to 1 mm. This light is not visible by naked eyes but can be seen by a camera (that is why these are also used in night vision cameras).

Working of IR Sensor

A photo diode gives response in term of change in resistance when light falls on it. That change is measured in terms of voltage.

An IR LED and a Photo diode are used in a combination for proximity and color detection. An IR LED (transmitter) emits IR light, that light gets reflected by the object, the reflected light is received by an IR receiver (Photo Diode). Amount of reflection and reception varies with the distance. This difference causes to change in input voltage through IR input. This variation in input voltage is used for proximity detection.

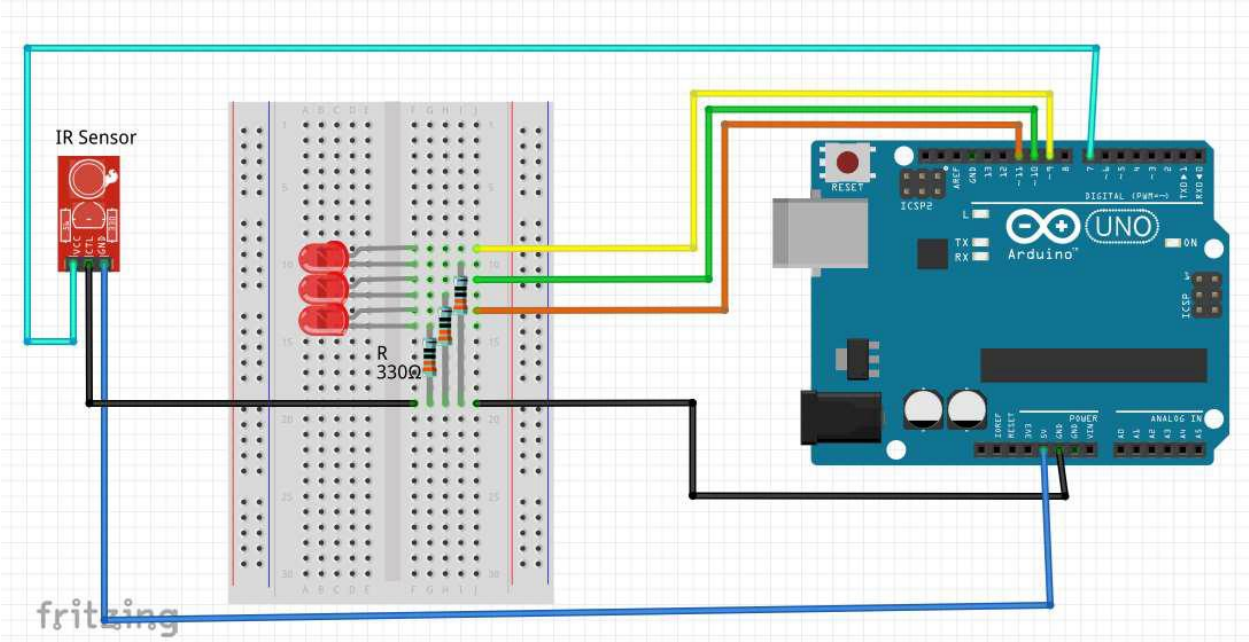
SUMMER GUIDANCE

ARDUINO AND IOT

WEEK 2

LET'S MAKE THE CONNECTIONS

3 LED system with IR



SUMMER GUIDANCE

ARDUINO AND IOT

WEEK 2

UPLOAD THE CODE

A simple program to control a 3 LED system using IR

```
int pin1 = 9;
int pin2 = 10;
int pin3 = 11;

int switch_pin = 7;
int pin = HIGH;

void setup() {
  // put your setup code here, to run once:
  pinMode(pin1, OUTPUT);
  pinMode(pin2, OUTPUT);
  pinMode(pin3, OUTPUT);
  pinMode(switch_pin, INPUT); //input from IR
}

void loop() {
  // put your main code here, to run repeatedly:
  pin = digitalRead(switch_pin); //reading the input

  if(pin == HIGH)
  {
    digitalWrite(pin1, HIGH);
    digitalWrite(pin2, LOW);
    digitalWrite(pin3, LOW);
    delay(100);
```

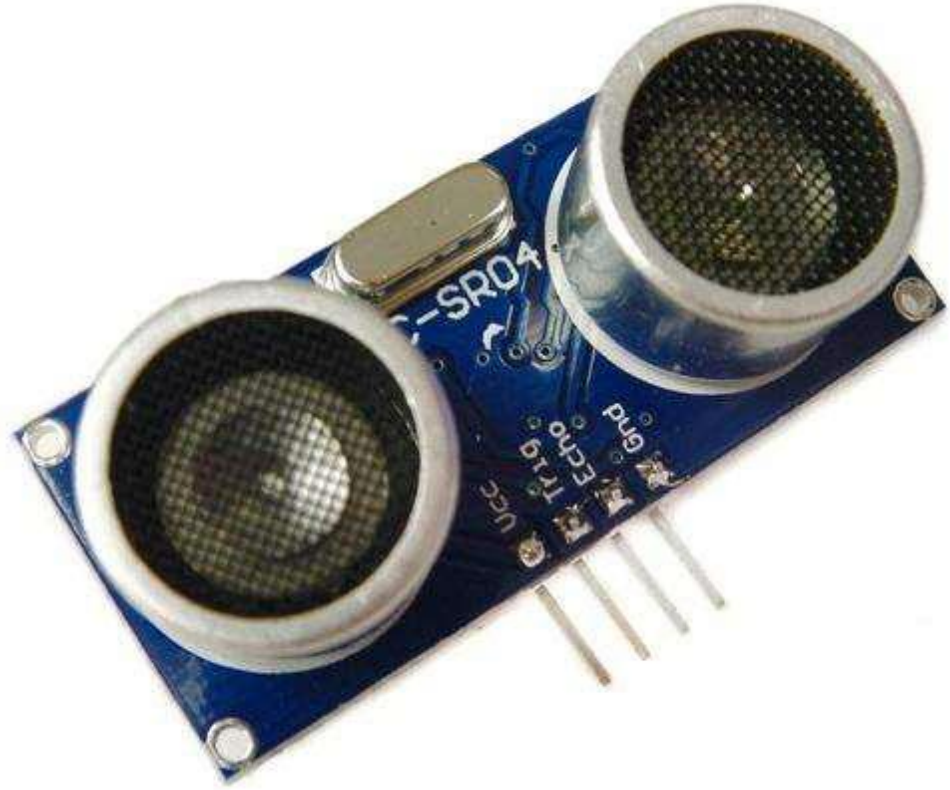
```
  digitalWrite(pin1, LOW);
  digitalWrite(pin2, HIGH);
  digitalWrite(pin3, LOW);
  delay(100);
  digitalWrite(pin1, LOW);
  digitalWrite(pin2, LOW);
  digitalWrite(pin3, HIGH);
  delay(100);
}
else
{
  digitalWrite(pin1, LOW);
  digitalWrite(pin2, LOW);
  digitalWrite(pin3, LOW);
}
}
```

SUMMER GUIDANCE

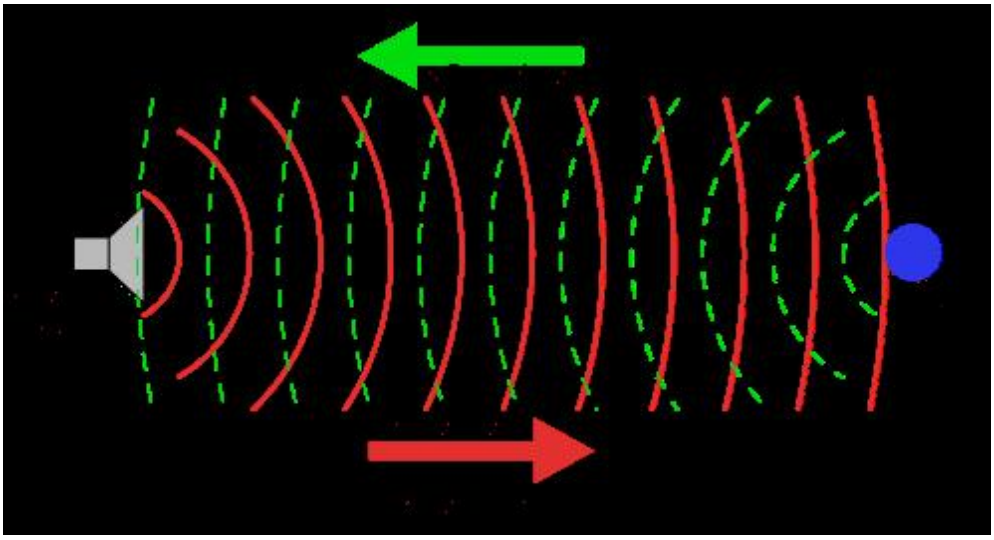
ARDUINO AND IOT

WEEK 2

ULTRASONIC SENSOR



HOW IT WORKS?



Working Of US Sensor

It emits an ultrasound at 40 000 Hz which travels through the air and if there is an object or obstacle on its path It will bounce back to the module. Considering the travel time and the speed of the sound you can calculate the distance. Typical range is between 2 cm to 450 cm.

It emits an ultrasound at 40 000 Hz which travels through the air and if there is an object or obstacle on its path It will bounce back to the module. Considering the travel time and the speed of the sound you can calculate the distance.

In order to generate the ultrasound you need to set the Trig on a High State for 10 µs. That will send out an 8 cycle sonic burst which will travel at the speed sound and it will be received in the Echo pin. The Echo pin will output the time in microseconds the sound wave traveled.

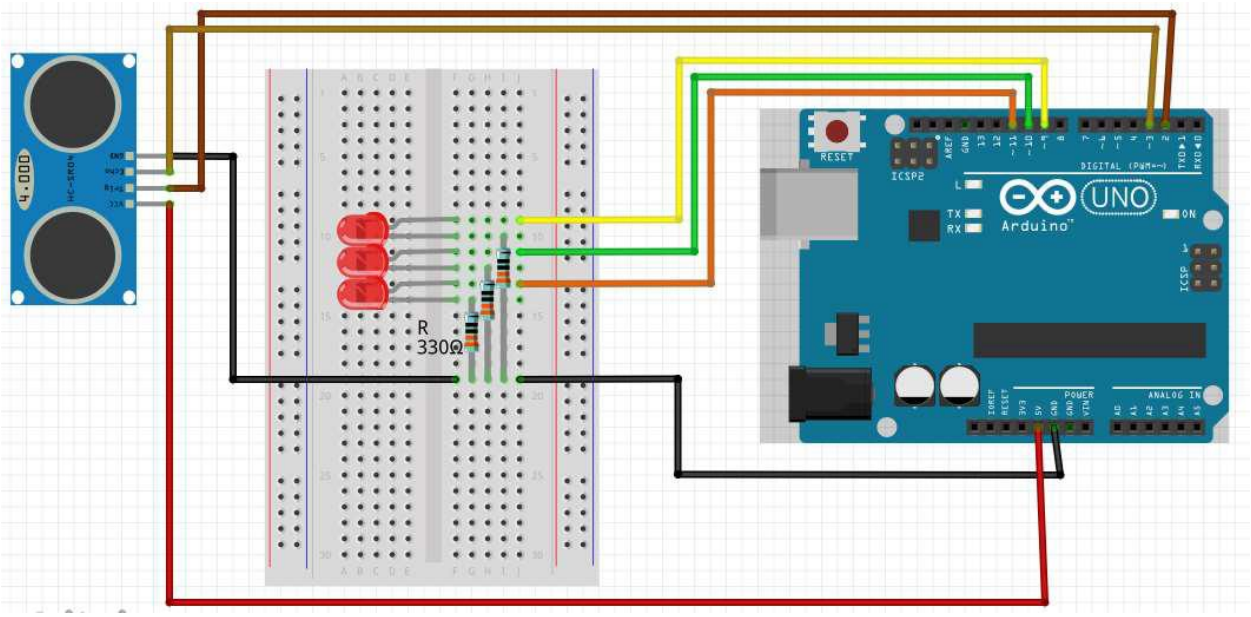
For example, if the object is 10 cm away from the sensor, and the speed of the sound is 340 m/s or 0.034 cm/µs the sound wave will need to travel about 294 u seconds. But what you will get from the Echo pin will be double that number because the sound wave needs to travel forward and bounce backward. So in order to get the distance in cm we need to multiply the received travel time value from the echo pin by 0.034 and divide it by 2.

SUMMER GUIDANCE

ARDUINO AND IOT

WEEK 2

LET'S MAKE THE CONNECTIONS



3 LED system with US sensor

SUMMER GUIDANCE

ARDUINO AND IOT

WEEK 2

UPLOAD THE CODE

A simple program to control a 3 LED system using US

```
#define echoPin 3   //Macro to declare 3 as echo

#define trigPin 2  // Macro to declare 2 as trig

int pin1 = 9;

int pin2 = 10;

int pin3 = 11;

long duration, distance;

void setup() {

  // put your setup code here, to run once:

  Serial.begin (9600);

  pinMode(trigPin, OUTPUT);

  pinMode(echoPin, INPUT);

  pinMode(pin1, OUTPUT);

  pinMode(pin2, OUTPUT);

  pinMode(pin3, OUTPUT);

}

void loop() {
```

```
 // put your main code here, to run repeatedly:
 digitalWrite(trigPin, LOW);
 delayMicroseconds(100);
digitalWrite(trigPin, HIGH);
 delayMicroseconds(100);
 digitalWrite(trigPin, LOW);
 duration = pulseIn(echoPin, HIGH);
 distance = duration/52.8;
 Serial.println(distance);
  if(distance < 15)
  {
    digitalWrite(pin1, HIGH);
    digitalWrite(pin2, LOW);
    digitalWrite(pin3, LOW);
    delay(100);
    digitalWrite(pin1, LOW);
    digitalWrite(pin2, HIGH);
    digitalWrite(pin3, LOW);
```

```
  delay(100);
  digitalWrite(pin1, LOW);
  digitalWrite(pin2, LOW);
  digitalWrite(pin3, HIGH);
  delay(100);
 }
 else
 {
  digitalWrite(pin1, LOW);
  digitalWrite(pin2, LOW);
  digitalWrite(pin3, LOW); }                }
```

For more explanation and clear understanding I would request everyone to watch the following tutorial series from no.17 to no.20

https://www.youtube.com/watch?v=eaHRHQ5Ca_U&list=PLGsOVKk2DiYx6CMdOQR_hmJ2NbB4mZQn-&index=18

A few mini projects are too covered under this and everyone try to implement it on your simulator tools.

SUMMER GUIDANCE

ARDUINO AND IOT

WEEK 2

Servo Motor

A **servo motor** is an electrical device which can push or rotate an object with great precision. If you want to rotate and object at some specific angles or distance, then you use servo motor. It is just made up of simple motor which run through **servo mechanism**. If motor is used is DC powered then it is called DC servo motor, and if it is AC powered motor then it is called AC servo motor. We can get a very high torque servo motor in a small and light weight packages

Applications of Servo Motor

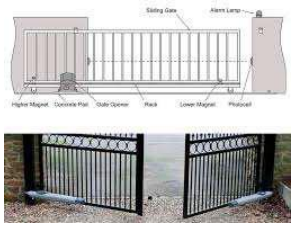
1) Humanoid Robot 2) Robotic arm

3) Crane 4) Automatic Gates

SUMMER GUIDANCE

ARDUINO AND IOT

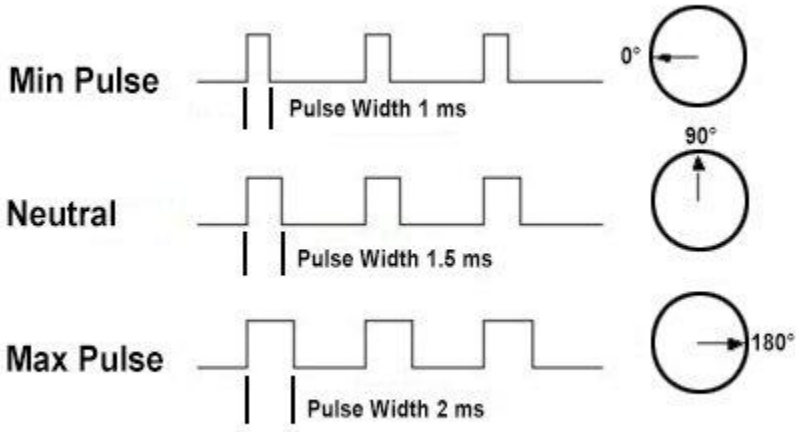
WEEK 2



Before we jump into the code and the connections it is necessary to understand how a servo motor works.

Servos are controlled by sending an electrical pulse of variable width, or pulse width modulation (PWM), through the control wire. There is a minimum pulse, a maximum pulse, and a repetition rate. A servo motor can usually only turn 90° in either direction for a total of 180° movement. The motor's neutral position is defined as the position where the servo has the same amount of potential rotation in the both the clockwise or counter-clockwise direction. The PWM sent to the motor determines position of the shaft, and based on the duration of the pulse sent via the control wire; the rotor will turn to the desired position. The servo motor expects to see a pulse every 20 milliseconds (ms) and the length of the pulse will determine how far the motor turns. For example, a 1.5ms pulse will make the motor turn to the 90° position. Shorter than 1.5ms moves it in the counter

clockwise direction toward the 0° position, and any longer than 1.5ms will turn the servo in a clockwise direction toward the 180° position.



When these servos are commanded to move, they will move to the position and hold that position. If an external force pushes against the servo while the servo is holding a position, the servo will resist from moving out of that position. The maximum amount of force the servo can exert is called the torque rating of the servo. Servos will not hold their position forever though; the position pulse must be repeated to instruct the servo to stay in position.

This video perfectly demonstrates our case:

https://www.youtube.com/watch?v=LXURLvga8bQ

Now you are good to go and start the implementation part

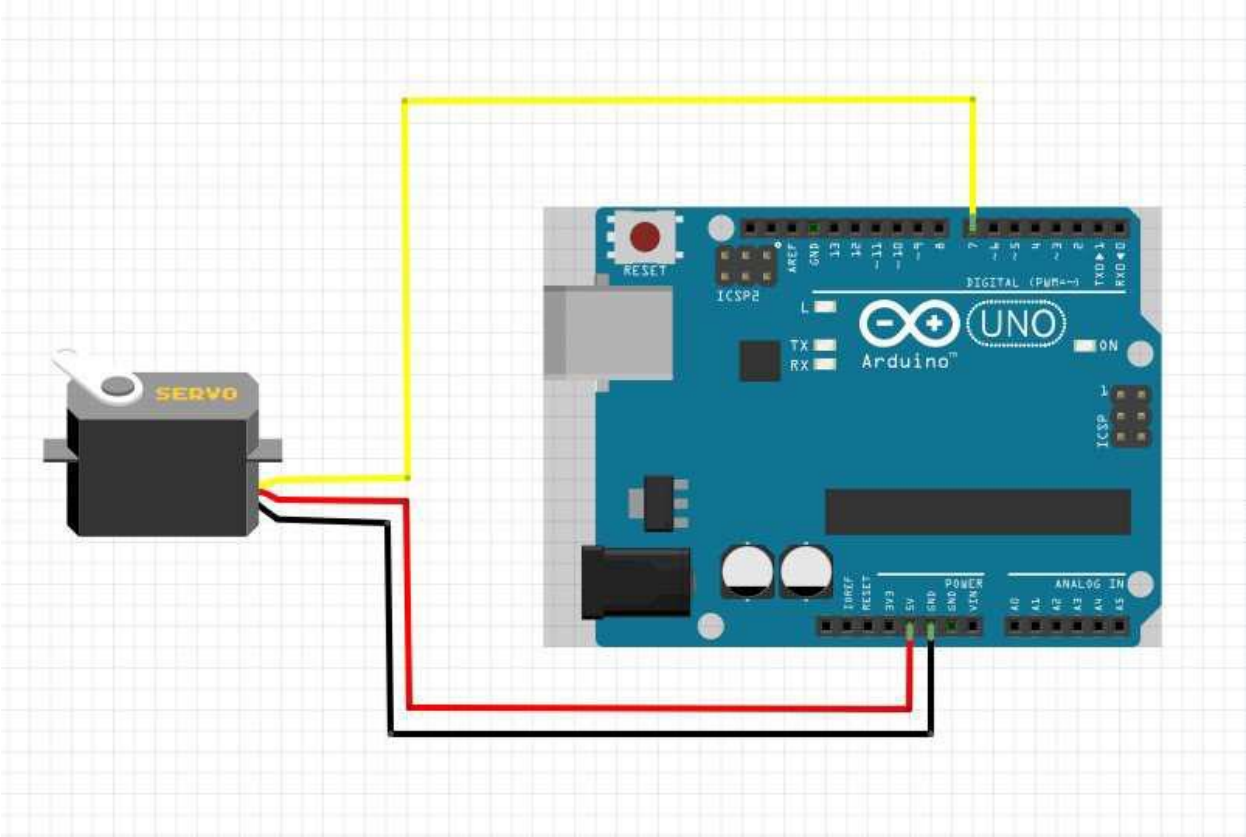
SUMMER GUIDANCE

ARDUINO AND IOT

WEEK 2

MAKE THE CONNECTIONS

Connection of servo motor to Arduino



// Servo red wire – 5V pin Arduino

//Servo brown wire – Ground pin Arduino

//Servo yellow wire – PWM(9) pin Arduino

UPLOAD THE CODE

SUMMER GUIDANCE

ARDUINO AND IOT

WEEK 2

A simple program to run servo motor

```
#include<Servo.h>  //loading the servo library
Servo myservo; //creating object as myservo
int pos=0;
void setup() {          // put your setup code here, to run once:
 myservo.attach(7); } // attach servo at pin 7
void loop() {    // put your main code here, to run repeatedly:
for(pos=0; pos<=180;pos+=1){
  myservo.write(pos); //writing the angle of movement in deg
  delay(15); //delay of 15 ms
 }
 for(pos=180;pos>=0;pos-=1){
  myservo.write(pos);
  delay(15);       }       }
```

Do watch this video which summarizes what we have learned uptil now :

https://www.youtube.com/watch?v=LXURLvga8bQ

This is hear is a very simple program for controlling how servos work. Servos are a huge asset for us and are frequently used in our projects so play around the code and try to experiment as much as you can until you get a firm grip on how servos works.

Here are a few more examples of Controlling Servos :

https://www.instructables.com/id/Arduino-Multiple-Servo-Control-With-Arduino/

https://maker.pro/arduino/tutorial/how-to-set-up-pan-tilt-camera-stand-with-arduino-uno-and-joystick-module

Once you have implemented the following two examples try out and simulate this mini-project.The tutorial for which has been given :

https://www.youtube.com/watch?v=4cdFHZ_Z-_Q

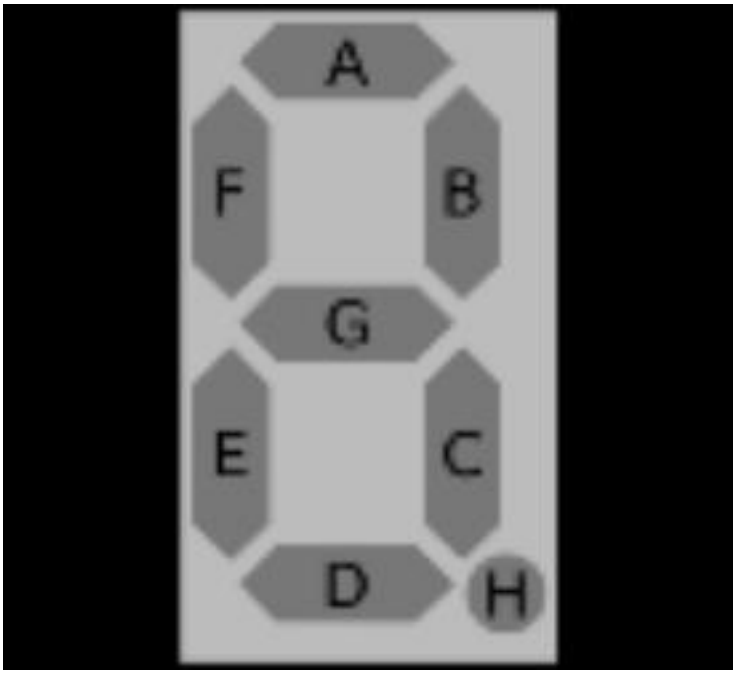
SUMMER GUIDANCE

ARDUINO AND IOT

WEEK 2

SEVEN SEGMENT DISPLAY

REFERENCE GUIDE

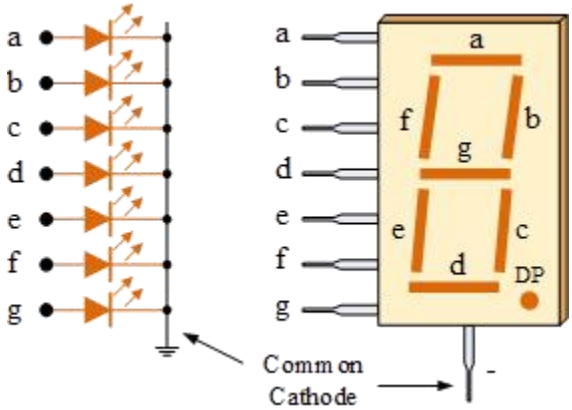


The displays common pin is generally used to identify which type of 7-segment display it is. As each LED has two connecting pins, one called the “Anode” and the other called the “Cathode”, there are therefore two types of LED 7-segment display called: **Common Cathode** (CC) and **Common Anode** (CA).

The difference between the two displays, as their name suggests, is that the common cathode has all the cathodes of the 7-segments connected directly together and the common anode has all the anodes of the 7-segments connected together and is illuminated as follows.

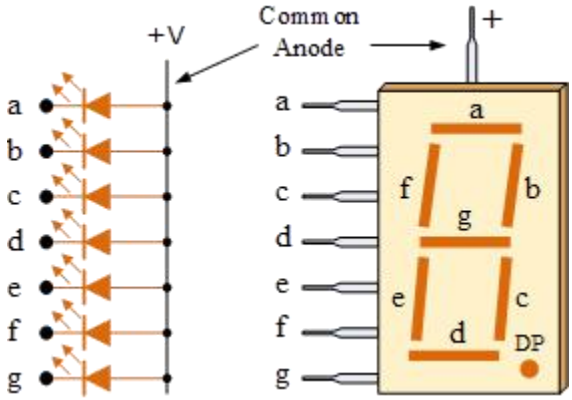
1. The Common Cathode (CC) – In the common cathode display, all the cathode connections of the LED segments are joined together to logic “0” or ground. The individual segments are illuminated by application of a “HIGH”, or logic “1” signal via a current limiting resistor to forward bias the individual Anode terminals (a-g).

Common Cathode 7-segment Display



2. The Common Anode (CA) – In the common anode display, all the anode connections of the LED segments are joined together to logic “1”. The individual segments are illuminated by applying a ground, logic “0” or “LOW” signal via a suitable current limiting resistor to the Cathode of the particular segment (a-g).

Common Anode 7-segment Display



In general, common anode displays are more popular as many logic circuits can sink more current than they can source. Also note that a common cathode display is not a direct replacement in a circuit for a common anode display and vice versa, as it is the same as connecting the LEDs in reverse, and hence light emission will not take place.

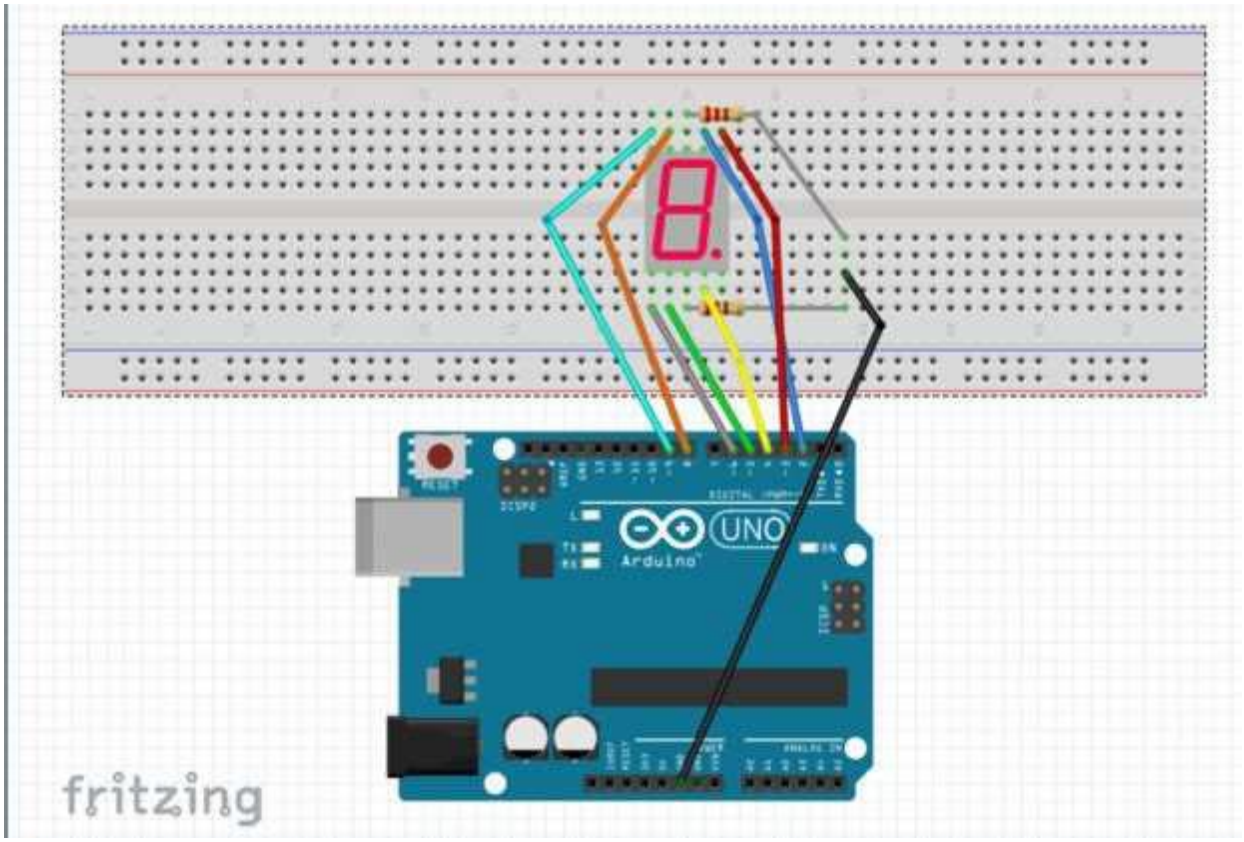
Depending upon the decimal digit to be displayed, the particular set of LEDs is forward biased. For instance, to display the numerical digit 0, we will need to light up six of the LED segments corresponding to a, b, c, d, e and f. Thus the various digits from 0 through 9 can be displayed using a 7-segment display as shown.

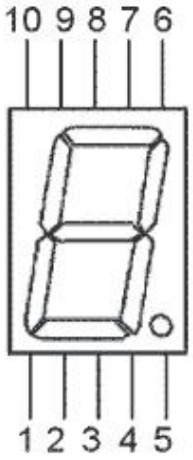
SUMMER GUIDANCE

ARDUINO AND IOT

WEEK 2

CIRCUIT DIAGRAM





SUMMER GUIDANCE

ARDUINO AND IOT

WEEK 2

A simple program:
To display numbers from 1-9

UPLOAD THE CODE

```
int pin1=2,pin2=3,pin3=4,pin4=5,pin5=6,pin6=7,pin7=8;

void setup()

{

pinMode(pin1,OUTPUT);          pinMode(pin2,OUTPUT);

pinMode(pin3,OUTPUT);          pinMode(pin4,OUTPUT);

pinMode(pin5,OUTPUT);          pinMode(pin6,OUTPUT);

pinMode(pin7,OUTPUT);

}

void loop() //display numbers from 1-9 with a delay of 500 ms

{

display(9);          delay(500);          display(8);          delay(500);

display(7);          delay(500);          display(6);          delay(500);

display(5);          delay(500);          display(4);          delay(500);

display(3);          delay(500);          display(2);          delay(500);

display(1);          delay(500);          display(0);          delay(500);

}

void display(int digit) //function to display output
```

```
{
digitalWrite(pin1, HIGH);        digitalWrite(pin2, HIGH);
digitalWrite(pin3, HIGH);        digitalWrite(pin4, HIGH);
digitalWrite(pin5, HIGH);        digitalWrite(pin6, HIGH);
digitalWrite(pin7, HIGH);
if(digit==0)
{
digitalWrite(pin1, HIGH);        digitalWrite(pin2, HIGH);
digitalWrite(pin3, HIGH);        digitalWrite(pin4, HIGH);
digitalWrite(pin5, HIGH);        digitalWrite(pin6, HIGH);
digitalWrite(pin7, LOW);
}
else if(digit==1)
{
digitalWrite(pin1, LOW);         digitalWrite(pin2, HIGH);
digitalWrite(pin3, HIGH);        digitalWrite(pin4, LOW);
digitalWrite(pin5, LOW);         digitalWrite(pin6, LOW);
digitalWrite(pin7, LOW);
}
else if(digit==2)
{
```

```
digitalWrite(pin1, HIGH);        digitalWrite(pin2, HIGH);
digitalWrite(pin3, LOW);         digitalWrite(pin4, HIGH);
digitalWrite(pin5, HIGH);        digitalWrite(pin6, LOW);
digitalWrite(pin7, HIGH);
}
else if(digit==3)
{
digitalWrite(pin1, HIGH);        digitalWrite(pin2, HIGH);
digitalWrite(pin3, HIGH);        digitalWrite(pin4, HIGH);
digitalWrite(pin5, LOW);         digitalWrite(pin6, LOW);
digitalWrite(pin7, HIGH);
}
else if(digit==4)
{
digitalWrite(pin1, LOW);         digitalWrite(pin2, HIGH);
digitalWrite(pin3, HIGH);        digitalWrite(pin4, LOW);
digitalWrite(pin5, LOW);         digitalWrite(pin6, HIGH);
digitalWrite(pin7, HIGH);
}
else if(digit==5)
{
```

```
digitalWrite(pin1, HIGH);        digitalWrite(pin2, LOW);
digitalWrite(pin3, HIGH);        digitalWrite(pin4, HIGH);
digitalWrite(pin5, LOW);         digitalWrite(pin6, HIGH);
digitalWrite(pin7, HIGH);
}
else if(digit==6)
{
digitalWrite(pin1, HIGH);
digitalWrite(pin2, LOW);         digitalWrite(pin3, HIGH);
digitalWrite(pin4, HIGH);        digitalWrite(pin5, HIGH);
digitalWrite(pin6, HIGH);        digitalWrite(pin7, HIGH);
}
else if(digit==7)
{
digitalWrite(pin1, HIGH);        digitalWrite(pin2, HIGH);
digitalWrite(pin3, HIGH);        digitalWrite(pin4, LOW);
digitalWrite(pin5, LOW);         digitalWrite(pin6, LOW);
digitalWrite(pin7, LOW);
}
else if(digit==8)
{
```

```
digitalWrite(pin1, HIGH);        digitalWrite(pin2, HIGH);
digitalWrite(pin3, HIGH);        digitalWrite(pin4, HIGH);
digitalWrite(pin5, HIGH);        digitalWrite(pin6, HIGH);
digitalWrite(pin7, HIGH);
}
else if(digit==9)
{
digitalWrite(pin1, HIGH);        digitalWrite(pin2, HIGH);
digitalWrite(pin3, HIGH);        digitalWrite(pin4, HIGH);
digitalWrite(pin5, LOW);         digitalWrite(pin6, HIGH);
digitalWrite(pin7, HIGH);        }       }
```

Here are is the example for using multiple seven segment display :

https://create.arduino.cc/projecthub/KVLakshmiSri/00-to-99-on-seven-segment-displays-1ca7e0