ROBOTICS
THE NEXT GENERATION

ROBOTROID
by Olatus Systems Private Limited

Layout & Schematics

Online Code Editor

Getting started guides

Language Reference

Single Sign On

CREATE

Forum and Support

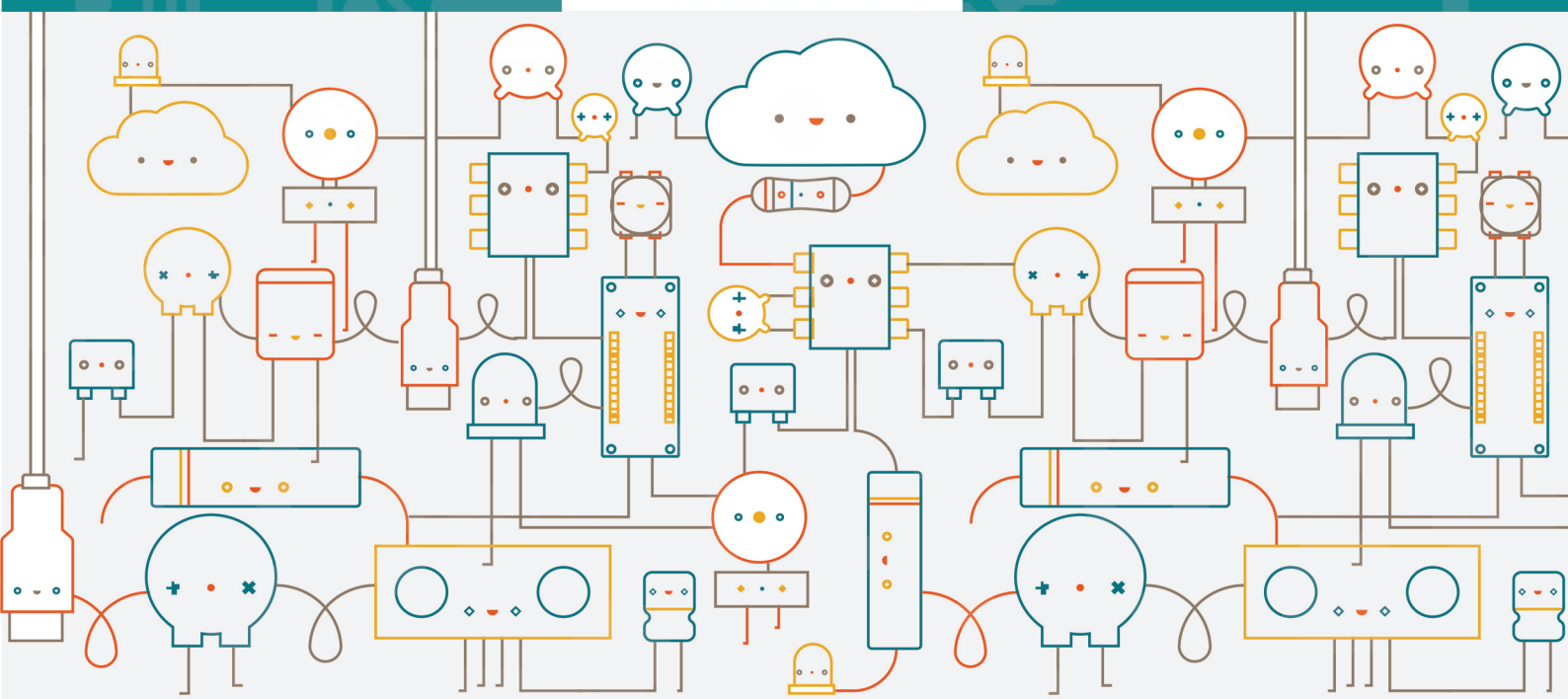
Sketchbook on Cloud

Project & Tutorials

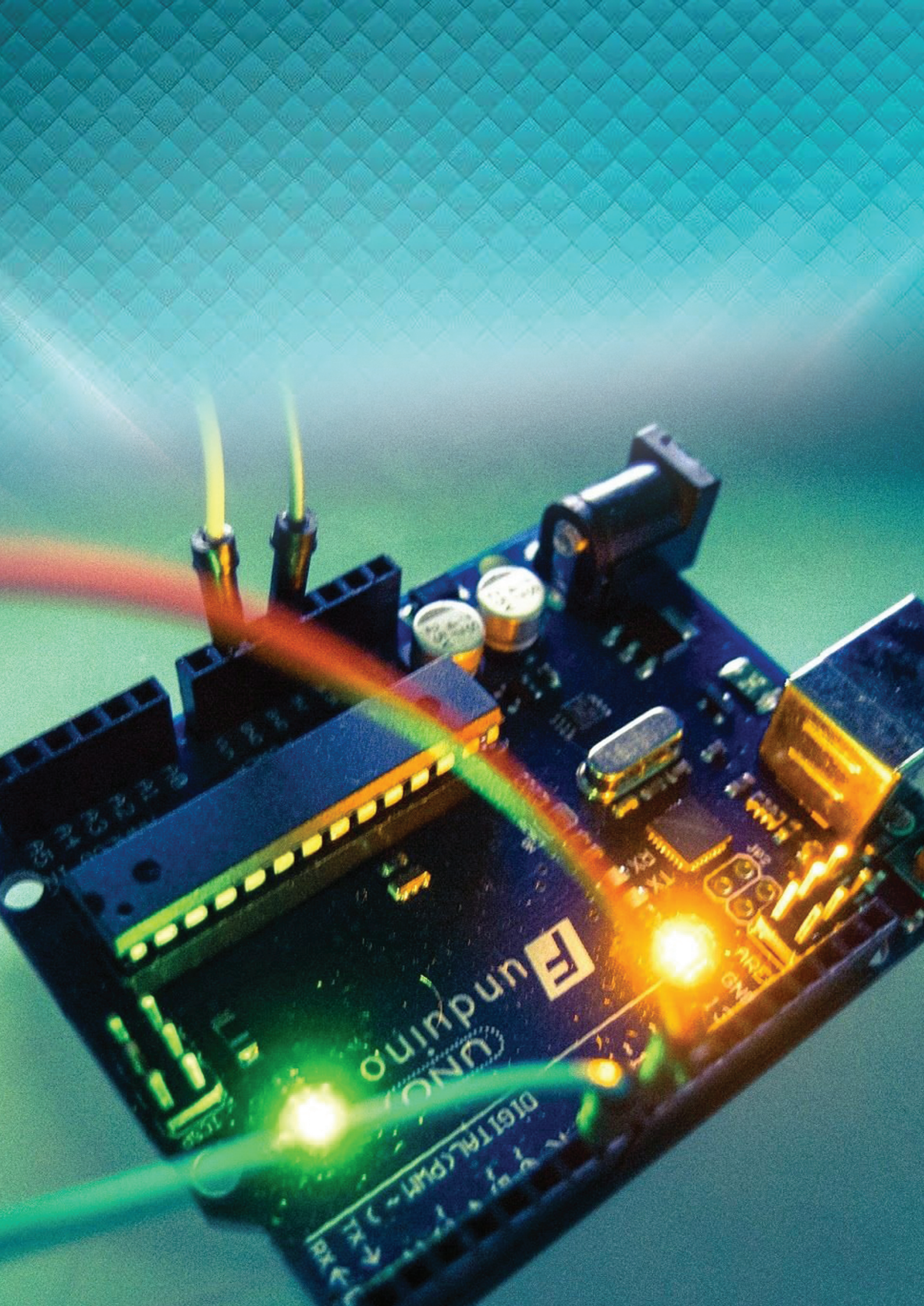
Arduino Store

Sharing features

Volume 1



ADD ON ROBOTICS ACTIVITIES

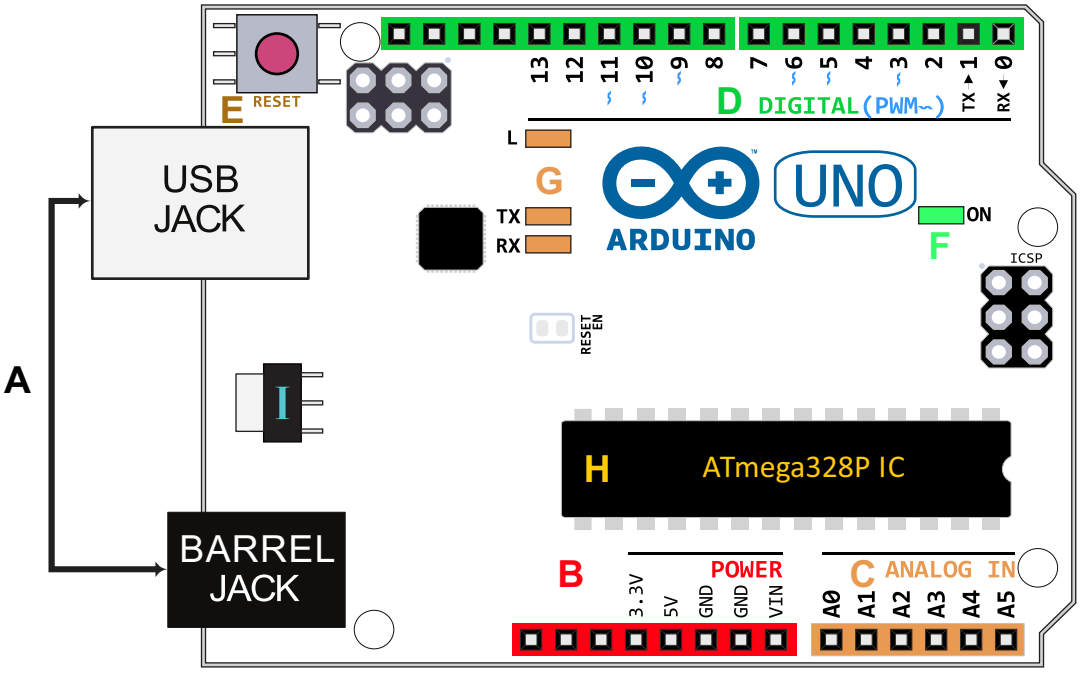

UNO
Arduino

Concept

Arduino is a development board based on the microcontroller ATmega328P. It is an open source platform which is used for building digital devices and interactive objects that can sense and control objects in the physical and digital world.

It runs on Mac, Windows, and Linux. For example, teachers and students have used it to build low cost scientific instruments, to prove chemistry and physics principles, or to get started with programming and robotics.

Arduino can interact with buttons, LEDs, motors, speakers, GPS units, cameras, the internet, and even your smartphone or your TV! For everything from robots and a heating pad hand warming blanket to honest fortune-telling machines, and even a Dungeons and Dragons dice-throwing gauntlet, the Arduino can be used as the brains behind almost any electronics project.



Parts of an Arduino Development Board:

A. Input Power (USB / Barrel Jack)

The Arduino UNO can be powered from a USB cable coming from your computer or a wall power supply through a barrel jack. The USB Port can also be used to upload code onto your Arduino Board.

B. Power Pins:

GND: Short for 'Ground'. There are three GND pins on the Arduino, any of which can be used to ground your circuit.

5V & 3.3V: As you might guess, the 5V pin supplies 5 volts of power, and the 3.3V pin supplies 3.3 volts of power. Most of the simple components used with the Arduino runs happily off on 5 or 3.3 volts.

C. Analog Inputs Pins:

Analog: The area of pins under the 'Analog In' label (A0 through A5 on the UNO) are Analog Input pins. These pins can read the signal from an analog sensor (like a temperature sensor).

D. Digital Input / Output (I/O) Pins:

Digital: We have digital pins (0 through 13 on the UNO). These pins can be used for both digital input (like telling if a button is pressed) and digital output (like glowing an LED).

PWM ~ (Analog Output Pins):

PWM: You may have noticed the tilde (~) next to some of the digital pins (3, 5, 6, 9, 10, and 11 on the UNO). These pins act as normal digital pins, but can also be used for generating analog output using a technique called Pulse Width Modulation. Think of these pins as being able to simulate analog output (like fading an LED in and out).

E. Reset Button

The Arduino has a reset button. Pressing it will temporarily connect the reset pin to ground and restart any code that is loaded on the Arduino

F. Power LED Indicator

Just beneath and to the right of the word “UNO” on your circuit board, there's a tiny LED next to the word 'ON. This LED should light up whenever you plug your Arduino into a power source. If this light doesn't turn on, there's a good chance something is wrong. Time to re-check your circuit!

G. TX RX LEDs

TX is short for transmit, RX is short for receive. These markings appear quite a bit in electronics to indicate the pins responsible for serial communication. In our case, there are two places on the Arduino UNO where TX and RX appear -- once by digital pins 0 and 1, and a second time next to the TX and RX indicator LEDs. These LEDs will give us some nice visual indications when ever our Arduino is receiving or transmitting data (like when we're loading a new program onto the board).

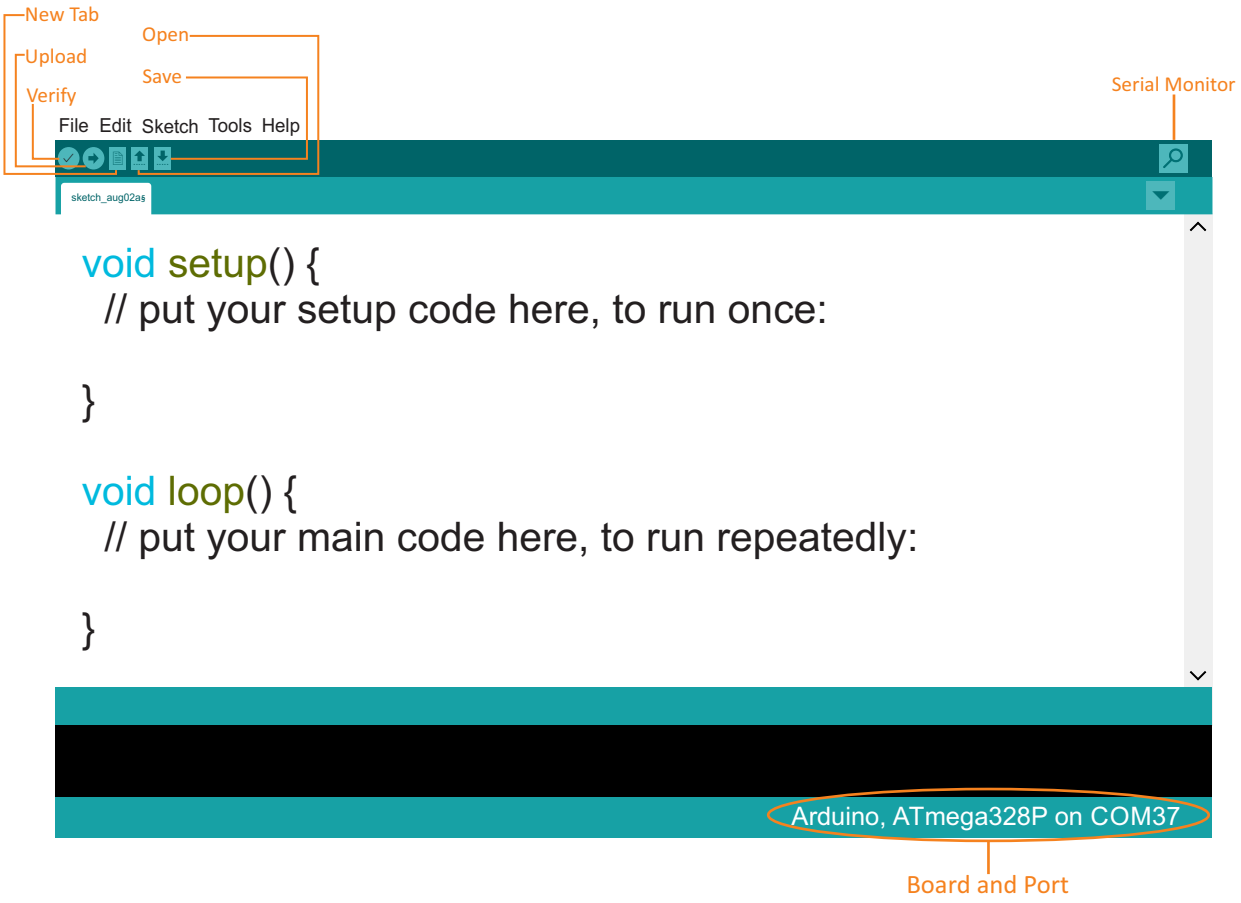
H. ATmega328P IC

The black thing with all the metal legs is an IC, or Integrated Circuit. Think of it as the brains of our Arduino. The main IC on the Arduino is slightly different from board type to board type, but is usually from the ATmega line of IC's from the ATMEL Company.

I. Voltage Regulator

The voltage regulator is not actually something you can (or should) interact with on the Arduino. But it is potentially useful to know that it is there and what it's for. The voltage regulator does exactly what it says -- it controls the amount of voltage that is let into the Arduino board. Think of it as a kind of gatekeeper; it will turn away an extra voltage that might harm the circuit. Of course, it has its limits, so don't hook up your Arduino to anything greater than 20 volts.

ARDUINO SOFTWARE IDE



Assessment

Tick The Correct One:

Q1. Which microcontroller is used in Arduino Board?

- ☐ Atmega328
- ☐ Atmega168
- ☐ AT89S51
- ☐ AT89C51

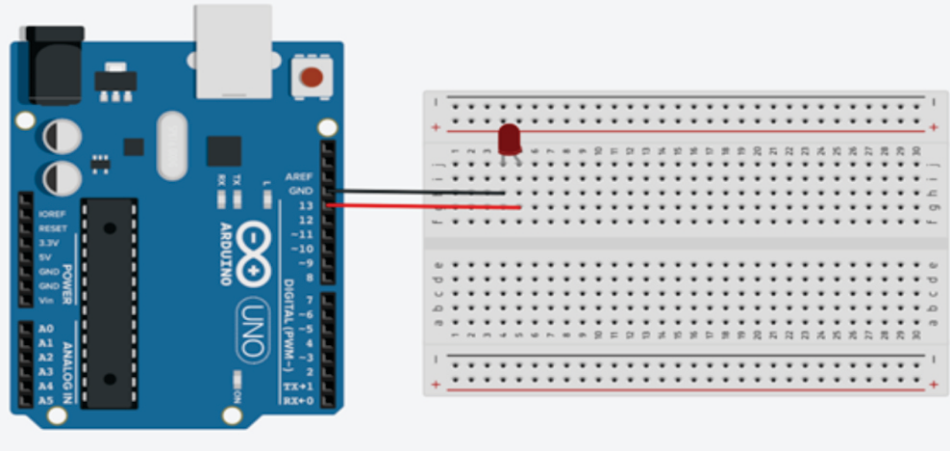
Input Output Arduino Microcontroller

Concept

Digital Output: A Digital Output allows you to control a voltage with a computer. If the computer instructs the output to be high, the output will produce a voltage (generally about 5 or 3.3 volts). If the computer instructs the output to be low, it is connected to ground and produces no voltage. E.g. Turning the Air Conditioner ON and OFF using a Remote.

Activity:- 1 - Blinking an LED

In this class, you will be introduced to Arduino programming and learn about Digital Logic. We will use, digital pins on the Arduino Board to control a digital device (LED)



Definitions & Syntax:

pinMode: Configures the specified pin to behave either as an input or an output.

Syntax: pinMode(pin,mode)
pin: the number of the pin whose mode you wish to set.
mode: INPUT, OUTPUT, or INPUT_PULLUP

digitalWrite: If the pin has been configured as an OUTPUT with pinMode you can command it to be HIGH (output 5 volts), or LOW (output 0 volts).

Syntax: **digitalWrite(pin, value)**
pin: the pin number.
value: HIGH or LOW.

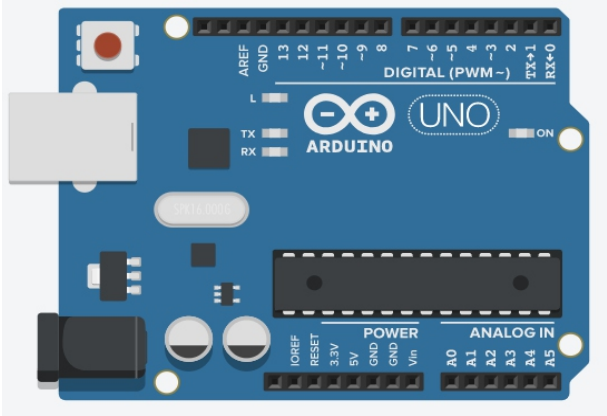
Delay: Pauses the program for the amount of time (in milliseconds) specified as parameter. (There are 1000 milliseconds in a second.)

Syntax: **delay(ms)**
'ms': the number of milliseconds to pause

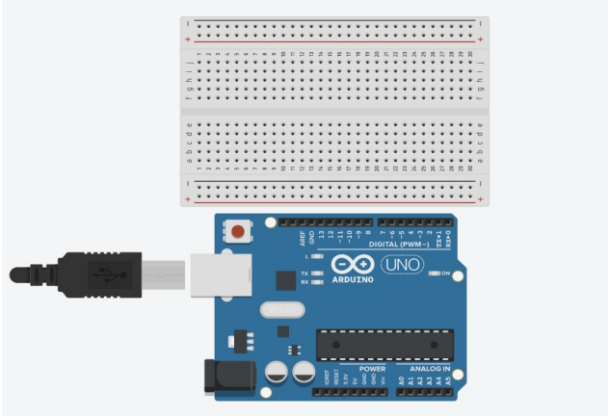
COMPONENTS REQUIRED

- Arduino Uno - 1
- Breadboard Mini - 1
- LEDs – 4 of Different Colors
- Jumper Cable(M-M) - 10

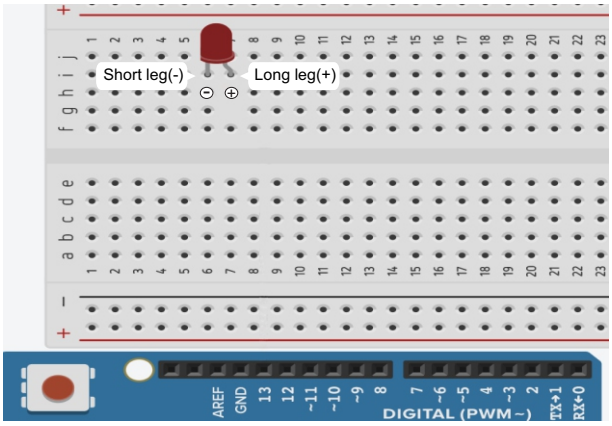
1 Take an UNO board



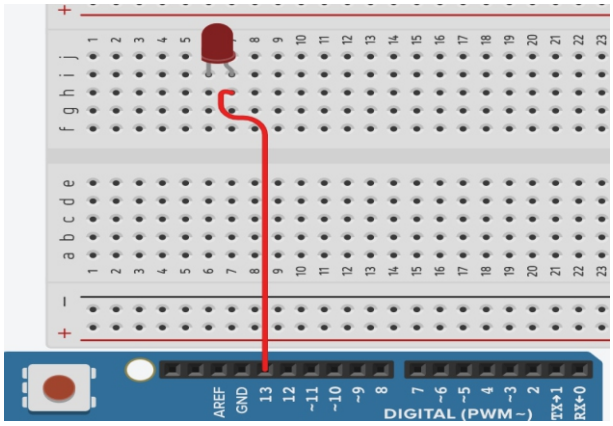
2 Place a breadboard as shown.



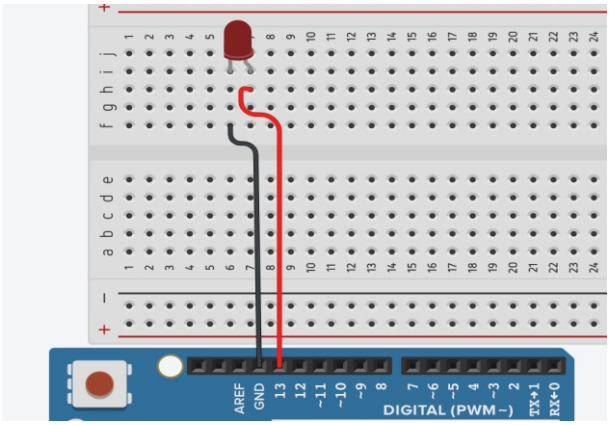
3 Take an LED and place it on the breadboard as shown



4 Take a red wire and connect it to the pin no. 13 on the Arduino board. Take the other end of the wire and connect it to the positive leg (long) of the LED as shown

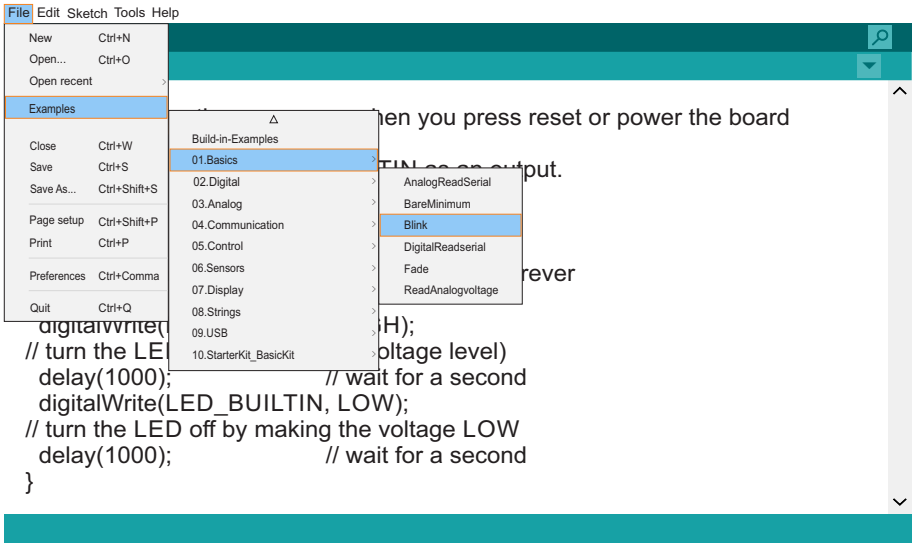


5 Now take a black wire and connect it to the pin named GND on the UNO board. Take the other end of the wire and connect it to the negative leg (short) of the LED.

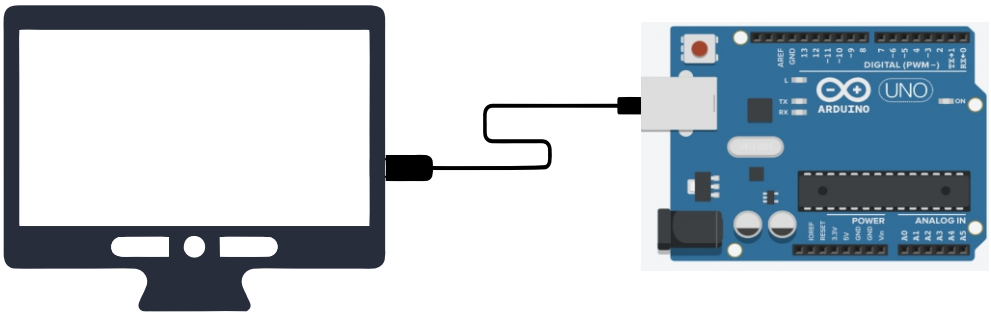


6 Open up the Arduino IDE Software.

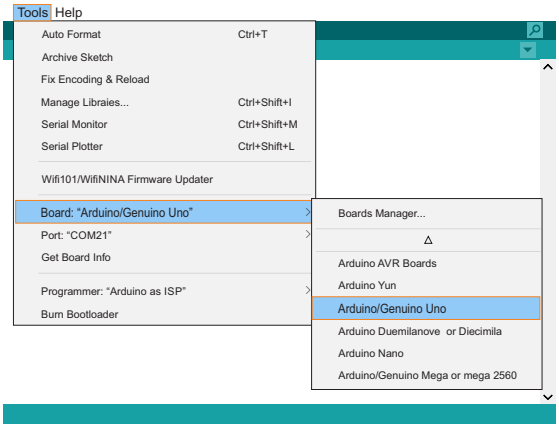
7 Open up the Blink example code by clicking on: File >> Examples >> 0.1 Basics >> Blink



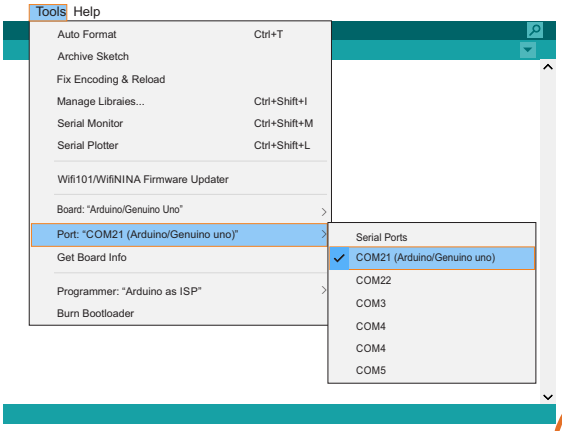
8 Connect your Arduino Board with Computer using USB cable.



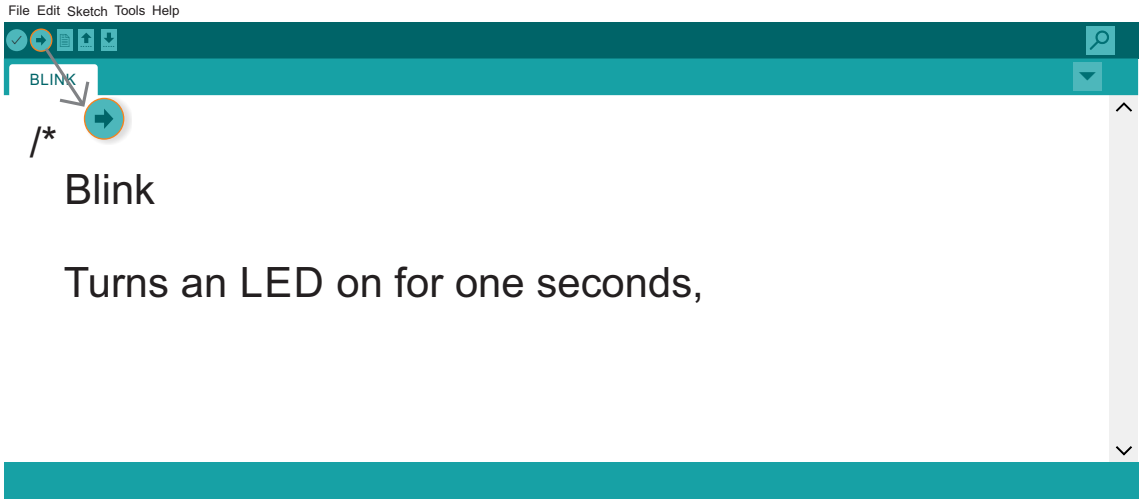
9a In the IDE, go to 'Tools' menu and set the board as shown.



9b Set the port of the board you just connected as shown.



10 Use the Upload button to burn the code in the Arduino.

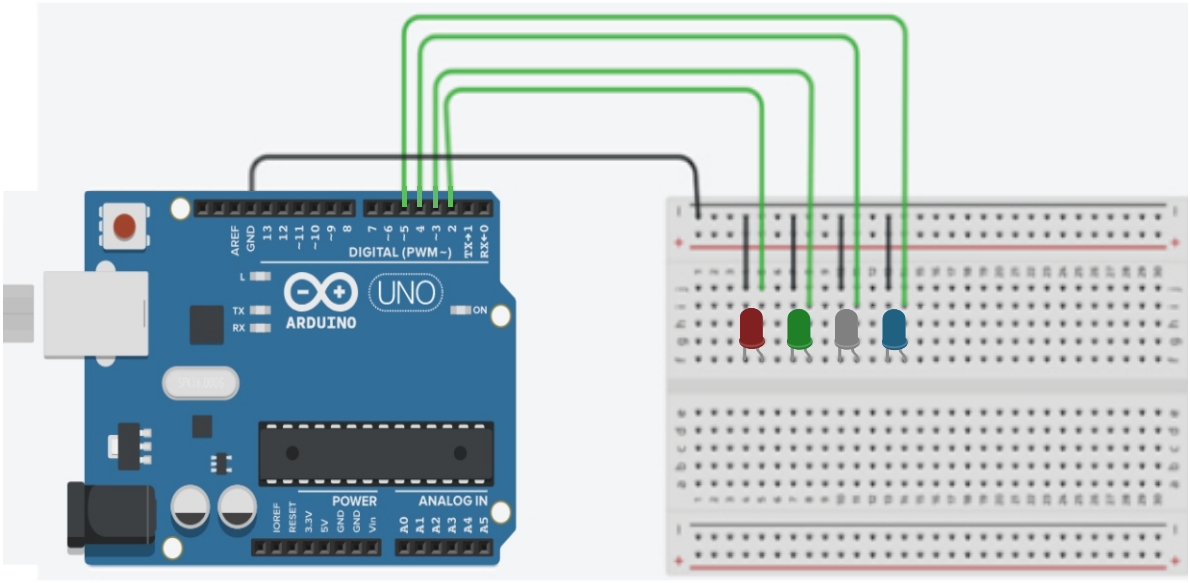


CODE

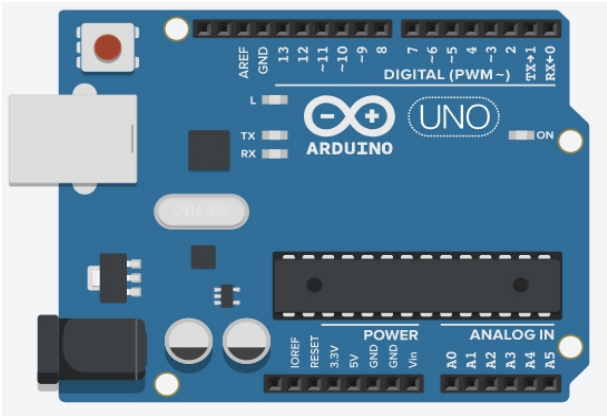
```
// the setup function runs once when you press reset or power the board
void setup()
{
  // initialize digital pin LED_BUILTIN as an output.
pinMode(LED_BUILTIN, OUTPUT);
}
// the loop function runs over and over again forever
void loop() {
digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
delay(1000);                       // wait for a second
digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
delay(1000);                       // wait for a second
}
```

Activity:- 2 - Led Chaser

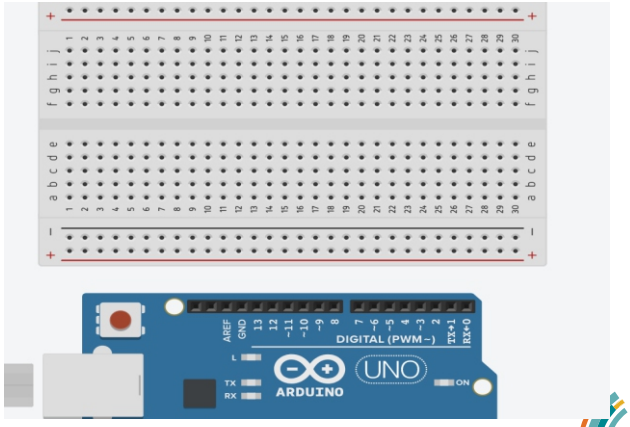
In this activity we will make an led chaser using four different colored LED's. We will program them to light in a trail in 1second intervals. The cycle will repeat, giving them running light appearance. The LED chaser is one of the most popular types of LED-driving circuit and is widely used in advertising displays and in running-light 'rope' displays in small discos, etc. You can create many more patterns by changing the 'delay function' and 'digitalWrite' function in the code, like the different patterns in diwali lights.



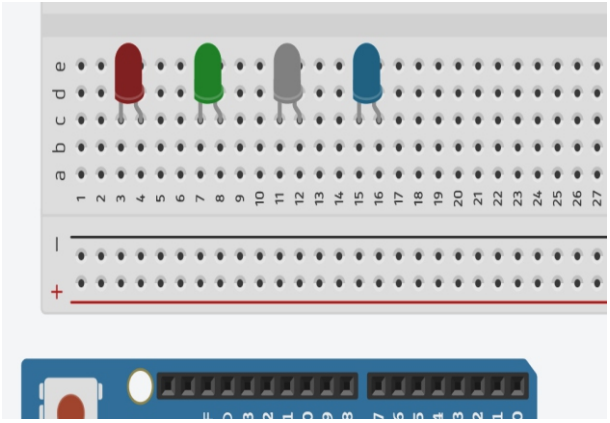
1 Take an UNO board



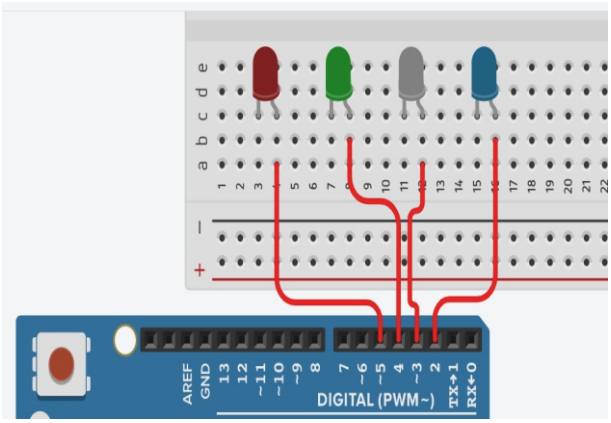
2 Place a breadboard as shown.



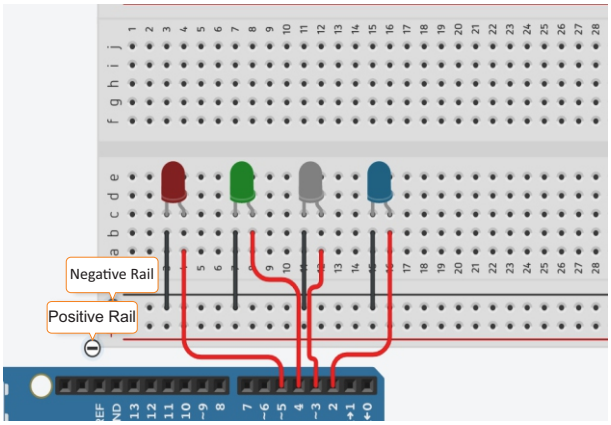
3 Place 4 LEDs on the breadboard as shown



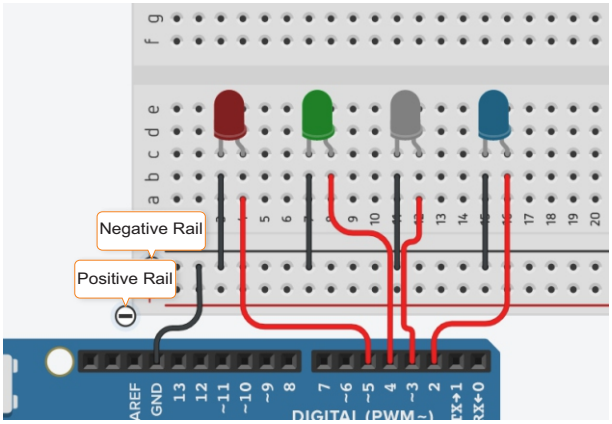
4 Connect the Long legs (+ve) of the LED's to pin nos. 2,3,4,5 on the board



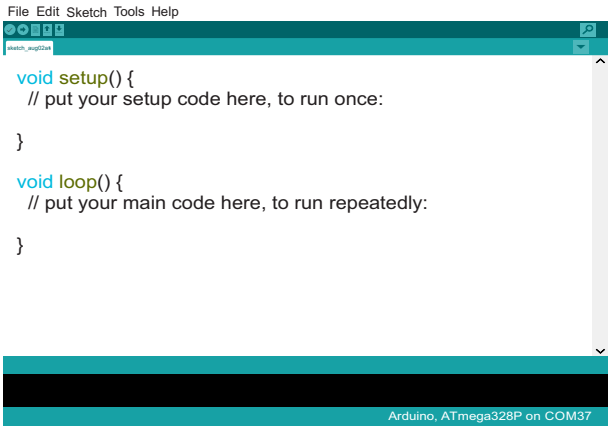
5 Connect shorter legs(-ve) of the LEDs to the negative(-) rail of the breadboard as shown



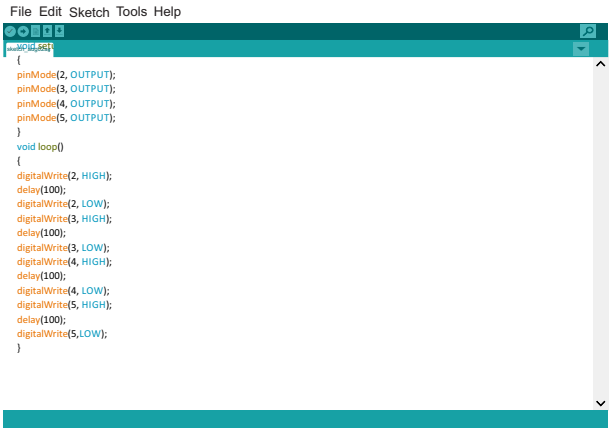
6 Connect the negative Rail of the Bread to the Ground (GND) pin on the breadboard



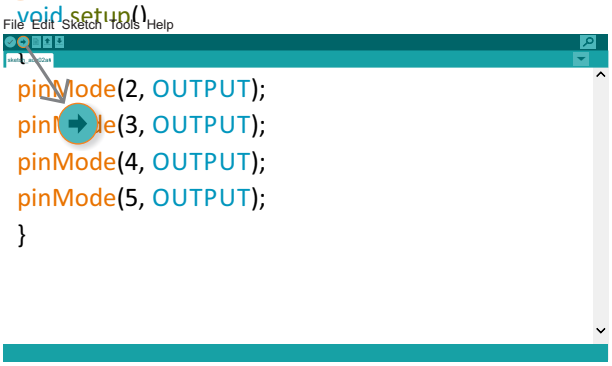
7 Open the Arduino IDE Software



8 Write the given code inside the code window



9 Upload the code onto your Arduino Board



CODE

```
void setup() {
pinMode(2, OUTPUT);
pinMode(3, OUTPUT);
pinMode(4, OUTPUT);
pinMode(5, OUTPUT);
}
void loop() {
digitalWrite(2, HIGH);
delay(100);
digitalWrite(2, LOW);
digitalWrite(3, HIGH);
delay(100);
digitalWrite(3, LOW);
digitalWrite(4, HIGH);
delay(100);
digitalWrite(4, LOW);
digitalWrite(5, HIGH);
delay(100);
digitalWrite(5,LOW);
}
```

Assessment

Tick The Correct One:

Q1. What value of time do you enter inside the delay(function)?

- ☐ Seconds
- ☐ Microseconds
- ☐ Milliseconds
- ☐ None of the above

Q2. What is the maximum number of LED's you can use in a chaser?

- ☐ 4
- ☐ 10
- ☐ 13
- ☐ None of the above

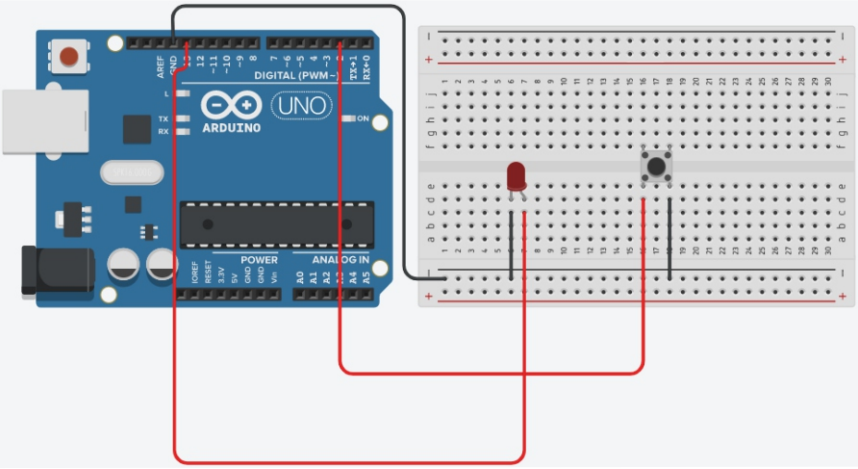
delay pinMode digitalWrite

Concept

Digital Input: A digital input detects if a voltage is above/below a specific threshold. If the voltage is higher than some value, the computer will detect the digital input as high/set/1. If the voltage is lower than some value, the computer will detect the digital input as low/clear/0. We connect three wires to the Arduino board. The first goes from one leg of the pushbutton through a resistor (here 1 KiloOhms) to the 5 volt supply. The second goes from the corresponding leg of the push button to ground. The third connects to a digital pin (here pin 6) which reads the button's state.When the pushbutton is open (unpressed) there is no connection between the two legs of the pushbutton, so the pin is connected to 5 volts and we read a HIGH. When the button is closed (pressed), it makes a connection between its two legs, connecting the pin to ground, so that we read a LOW. (The pin is still connected to 5 volts, but the resistor in-between those means that the pin is "closer" to ground.)

Activity:- 1 - Push Button

In this activity, we will learn how to use an Arduino board to obtain digital inputs from a push button (switch)



Definitions & Syntax:

Pushbutton: The pushbutton is a component that connects two points in a circuit when you press it. The example turns on an LED when you press the button.

digitalRead: Reads the value from a specified digital pin, either HIGH or LOW.

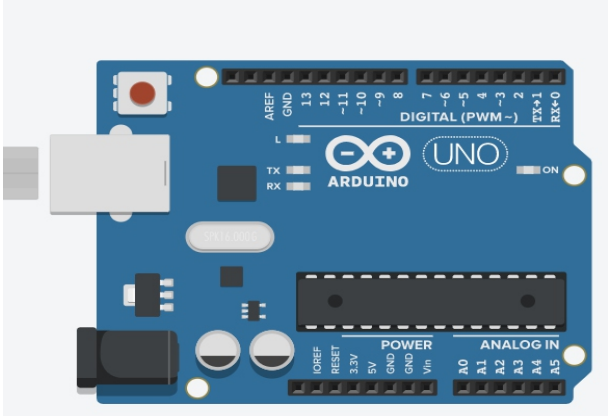
Syntax:

digitalRead(pin)
pin: the number of the digital pin you want to read
Returns-HIGH or LOW

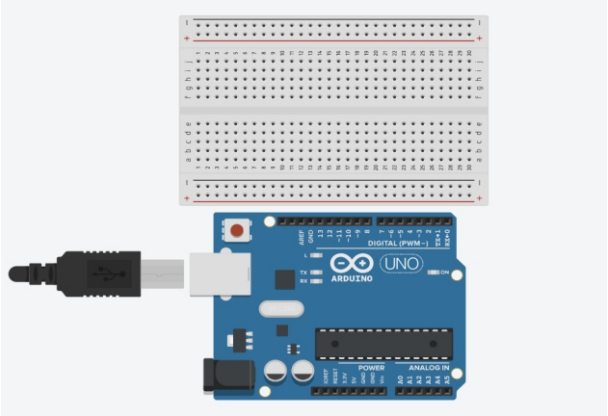
COMPONENTS REQUIRED

- Arduino Uno - 1
- Breadboard Mini - 1
- LEDs – 4 of Different Colors
- Jumper Cable(M-M) - 10
- Piezo Buzzer - 1
- Pushutton - 5

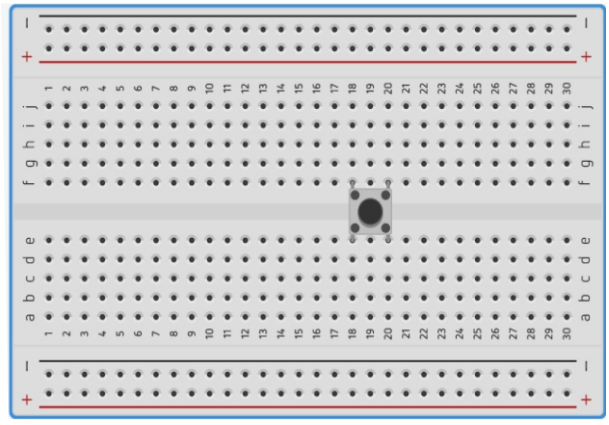
1 Take an Arduino Board



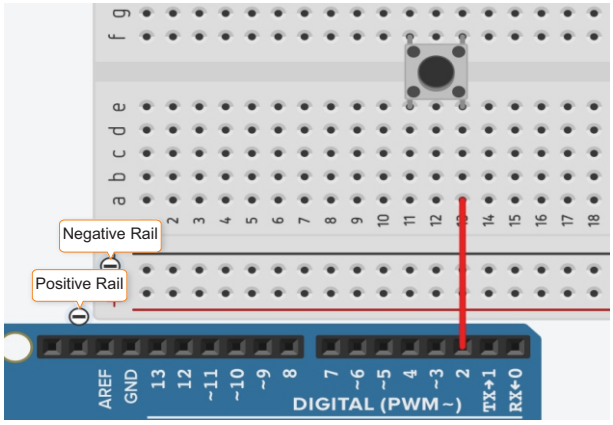
2 Place a breadboard along the Arduino board



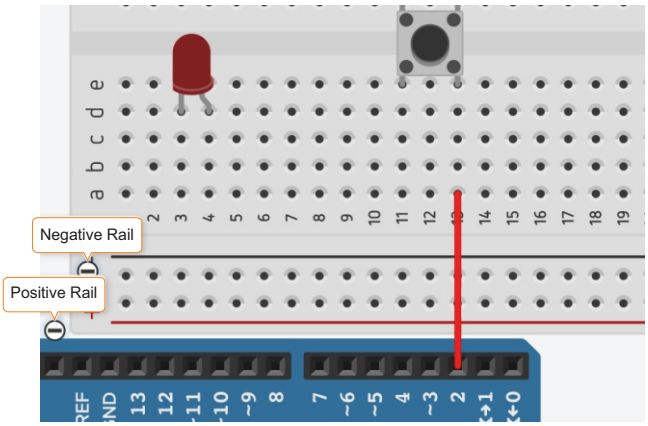
3 Place a button on the breadboard, as shown.



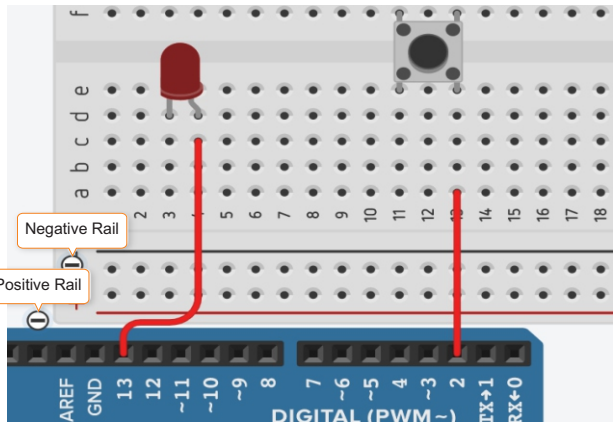
4 Connect one leg of the push button to the Arduino pin number 2 (or any pin that you want).



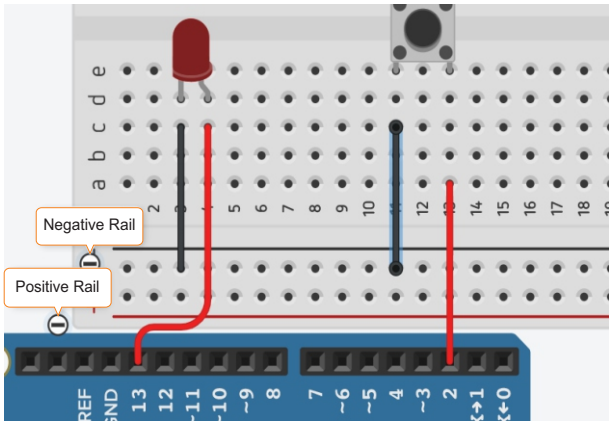
5 Place an LED on the breadboard as shown



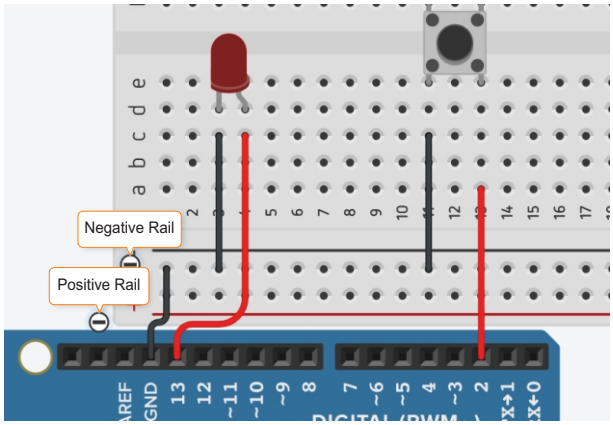
6 Connect the long leg(+ve) of the LED to pin number 13 on the Arduino board



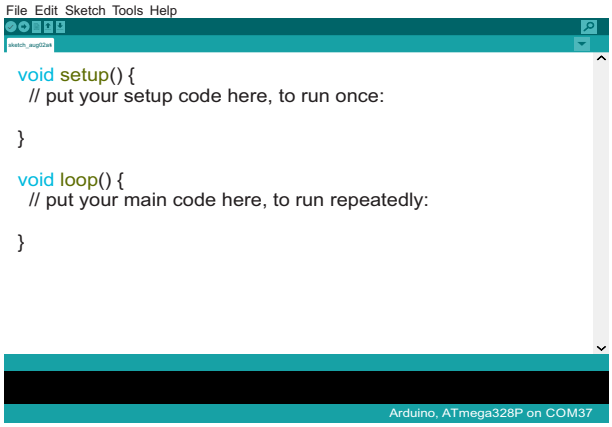
7 Connect the short leg of the LED (-ve) and one leg of the button to the negative power rail of the breadboard.



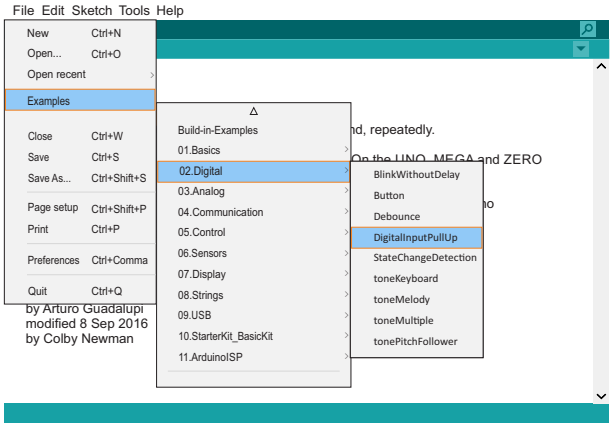
8 Connect the GND (ground) pin of the Arduino board to the negative rail of the breadboard



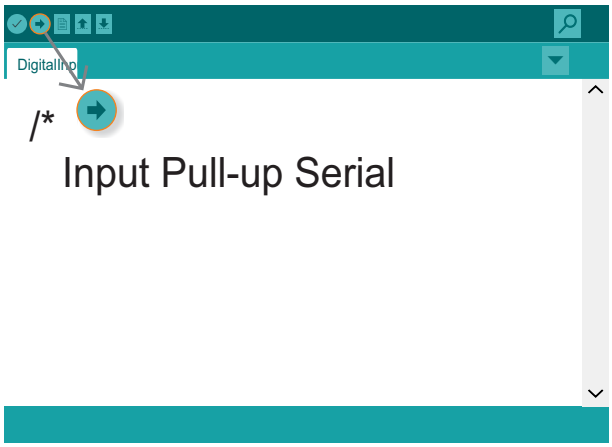
9 Open up the Arduino IDE software



10 Open up the DigitalInputPullup example code as shown



11 Upload the code

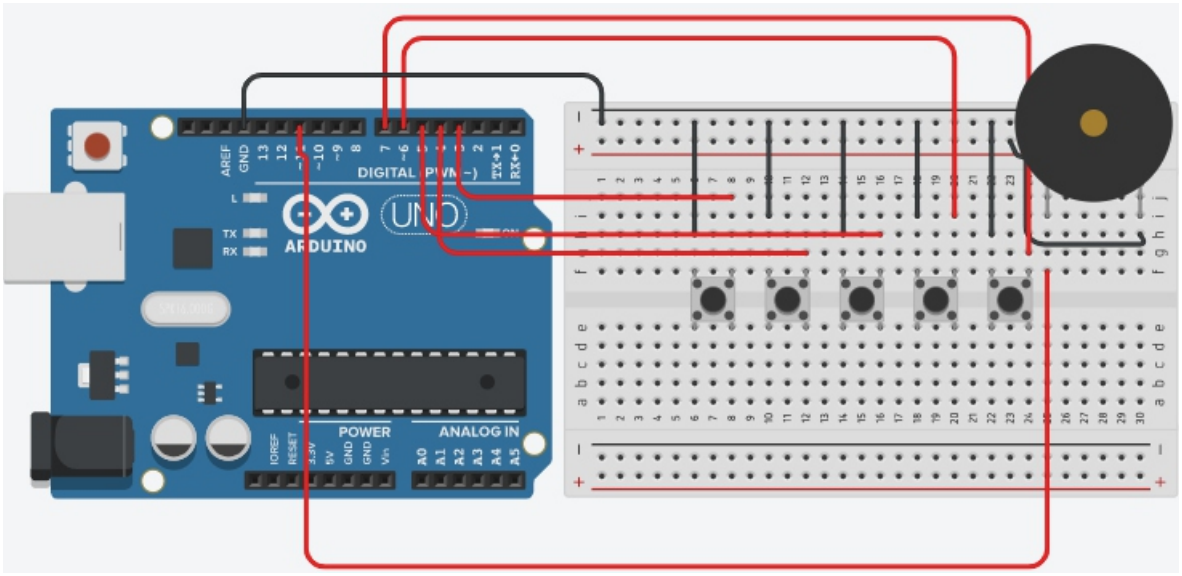


CODE

```
void setup() {
 //start serial connection
 Serial.begin(9600);
 //configure pin 2 as an input and enable the internal pull-up resistor
 pinMode(2, INPUT_PULLUP);
 pinMode(13, OUTPUT);

}

void loop() {
 //read the pushbutton value into a variable
 int sensorVal = digitalRead(2);
 //print out the value of the pushbutton
 Serial.println(sensorVal);

 // Keep in mind the pull-up means the pushbutton's logic is inverted. It goes
 // HIGH when it's open, and LOW when it's pressed. Turn on pin 13 when the
 // button's pressed, and off when it's not:
 if (sensorVal == HIGH) {
   digitalWrite(13, LOW);
 } else {
   digitalWrite(13, HIGH);
 }
}
```

Activity:- 2 - Piano

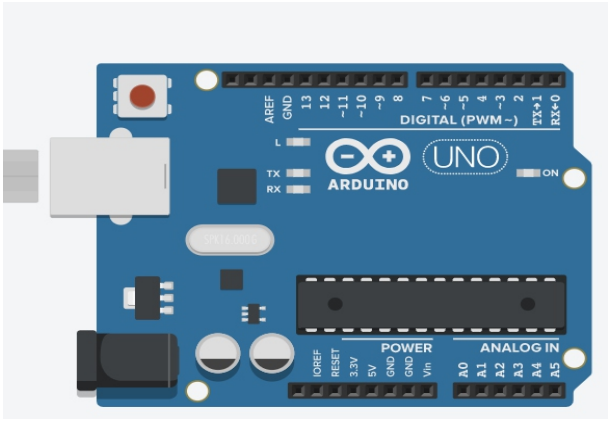
The tone library generates a square wave of the specified frequency on a pin. A duration can be specified, otherwise the wave continues until a call to noTone(). The pin can be connected to a piezo buzzer or other speaker to play tones. Only one tone can be generated at a time, while the minimum frequency that can be played is 31 Hz



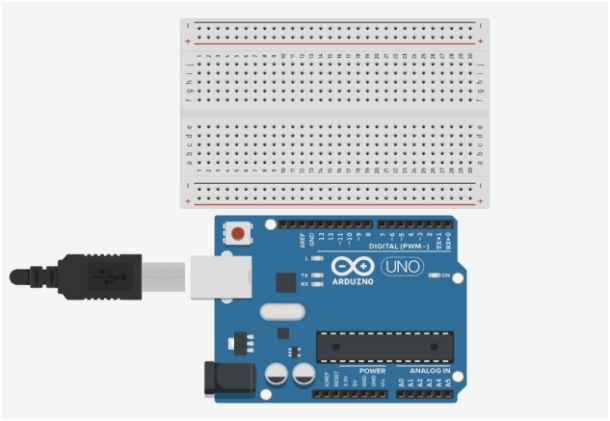
Definitions & Syntax:

Syntax:	tone(pin, frequency)
	tone(pin, frequency, duration)
Parameters	pin: the pin on which to generate the tone.
	frequency: the frequency of the tone in hertz.
	duration: the duration of the tone in milliseconds (optional).
noTone-	Stops the generation of a sound wave triggered by tone().
	Has no effect if no tone is being generated.
Syntax:	noTone(pin)
Parameters	pin: the pin on which to stop generating the tone

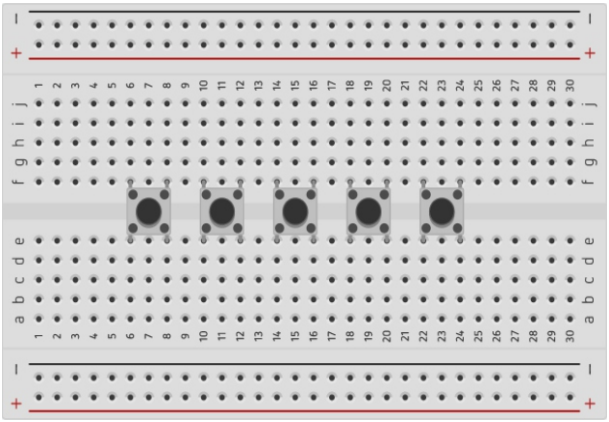
1 Take an Arduino Board.



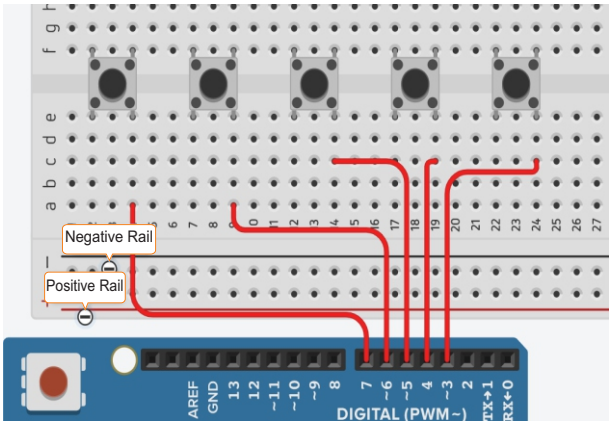
2 Place a breadboard along the Arduino board.



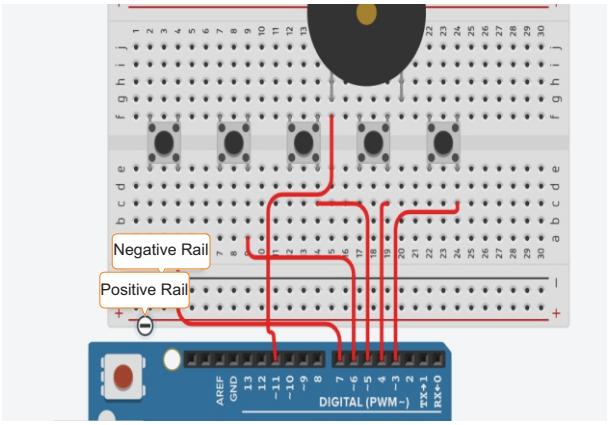
3 Connect 5 buttons on the breadboard as shown.



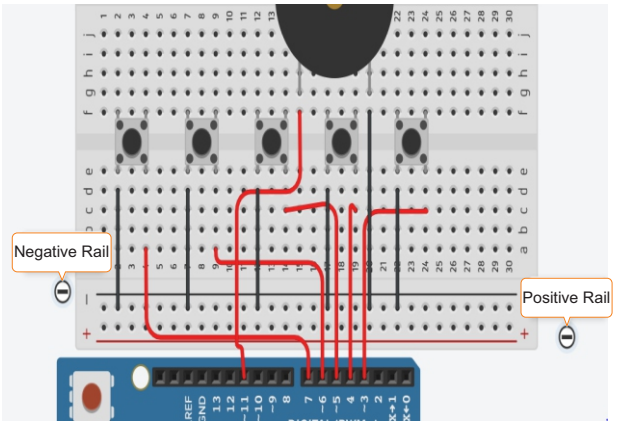
4 Connect one terminal of the buttons to Arduino's pin number 3, 4,5,6,7 respectively, as shown.



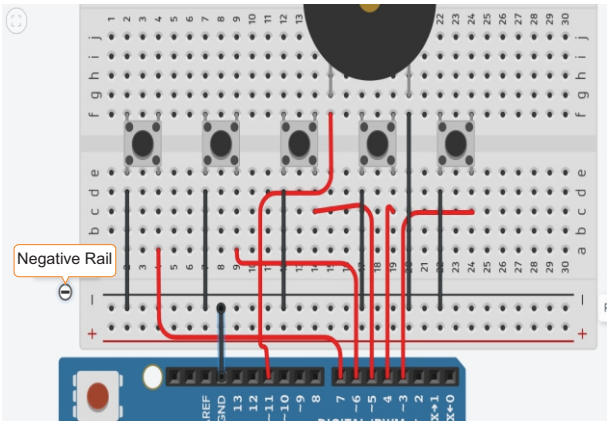
5 Now connect a piezo buzzer on the bread board and connect its longer leg (+ve) to Arduino's pin number 11 as shown.



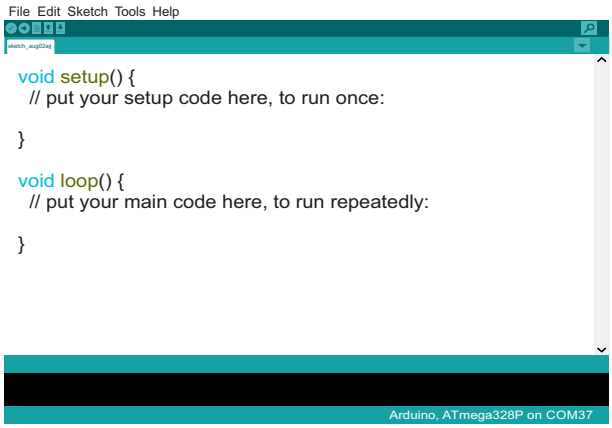
6 Connect the short leg(-ve) of the buzzer and the remaining one terminals of all the buttons to the negative rail of the breadboard.



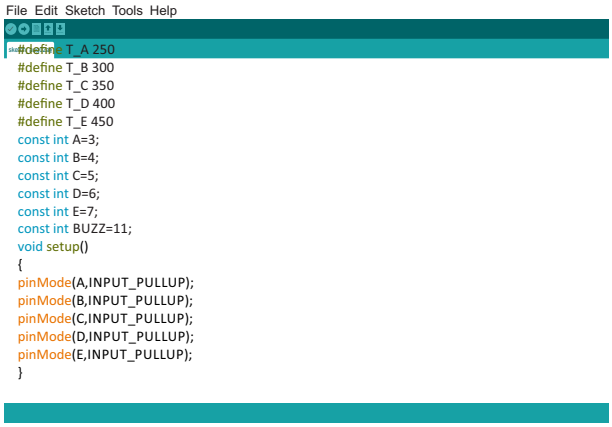
7 Connect the GND (ground) pin of the Arduino board to the negative rail of the breadboard, as shown.



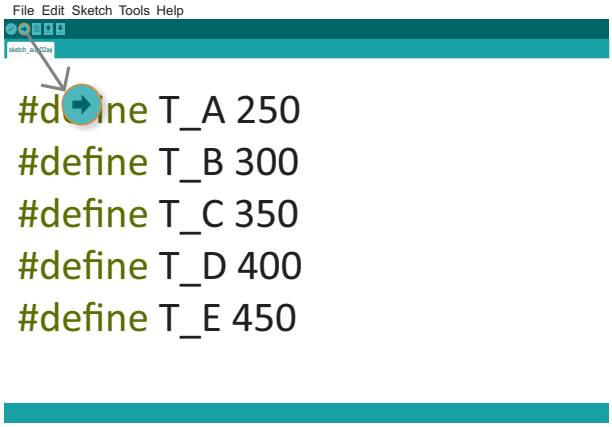
8 Open the Arduino IDE software



9 Write the code that has been given with the activity



10 Upload the code



CODE

```
#define T_A 250
#define T_B 300
#define T_C 350
#define T_D 400
#define T_E 450
const int A=3;
const int B=4;
const int C=5;
const int D=6;
const int E=7;
const int BUZZ=11;
```

```
void setup()
{
pinMode(A,INPUT_PULLUP);
pinMode(B,INPUT_PULLUP);
pinMode(C,INPUT_PULLUP);
pinMode(D,INPUT_PULLUP);
pinMode(E,INPUT_PULLUP);
}
void loop()
{
while(digitalRead(A)==LOW)
{
tone(BUZZ,T_A);
}
while(digitalRead(B)==LOW)
{
tone(BUZZ,T_B);
}
while(digitalRead(C)==LOW)
{
tone(BUZZ,T_C);
}
while(digitalRead(D)==LOW)
{
tone(BUZZ,T_D);
}
while(digitalRead(E)==LOW)
{
tone(BUZZ,T_E);
}
noTone(BUZZ);
}
```

Assessment

Tick The Correct One:

Q1. What kind of output does a push button give out?

- ☐ 0/1
- ☐ HIGH/LOW
- ☐ ON/OFF
- ☐ None of the above

Q2. What is the minimum frequency of tones that can be played?

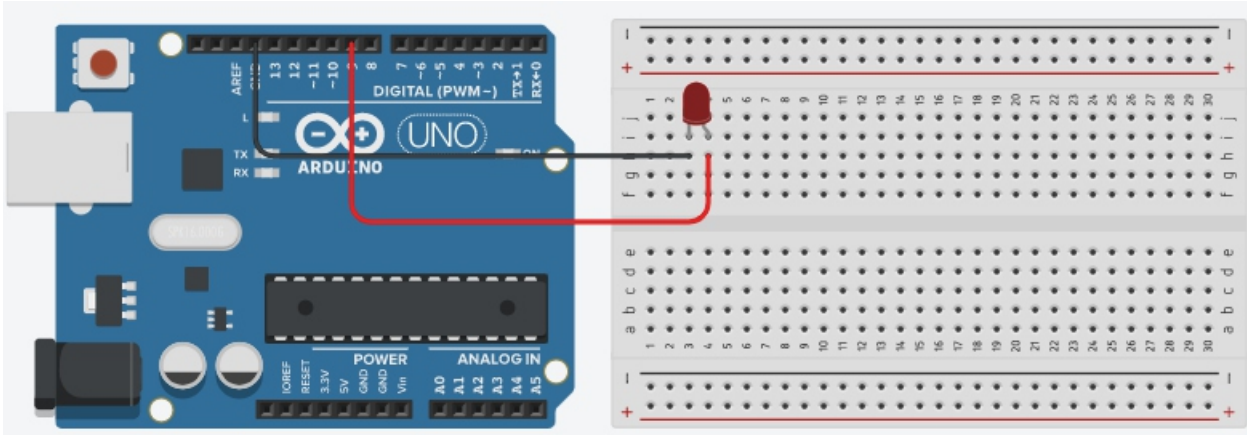
- ☐ 31hz
- ☐ 20hz
- ☐ 20000hz
- ☐ 1000hz

tone notone delay digitalRead digitalWrite

Concept

Analog Output- unlike Digital logic, where the output is a simple ON or OFF, analog output provides multiple variable output states between ON and OFF. Here, the voltage levels can have infinite values between 0 volts to 5 volts (i.e. 0.1, 0.01, 0.002 etc.). This output can be used to display values which are generally varying, like temperatures, amount of light etc. In this activity, we will use an Arduino to obtain digital output on specially marked PWM pins (with a ~ symbol) to obtain a dimming (or fading) effect on a LED

Activity:- 1 - LED Fading



Definitions & Syntax:

PWM Pins- Specially marked pins on an Arduino board, capable of producing an Analog output. The output produced can have 256 different voltage values (0-Low, 255-Highest). They are marked with a tilde (~) symbol on the board

analogWrite- Writes an analog value (PWM wave) to a pin. Can be used to light a LED at varying brightnesses or drive a motor at various speeds.

Syntax analogWrite(pin, value)

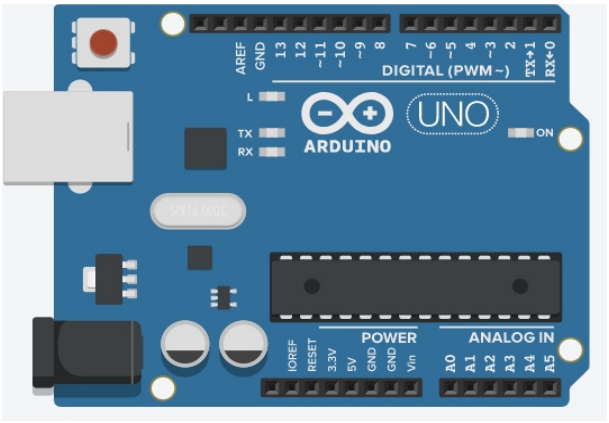
Parameters- pin: the pin to write to

value: the duty cycle: between 0 (always off) and 255 (always on).

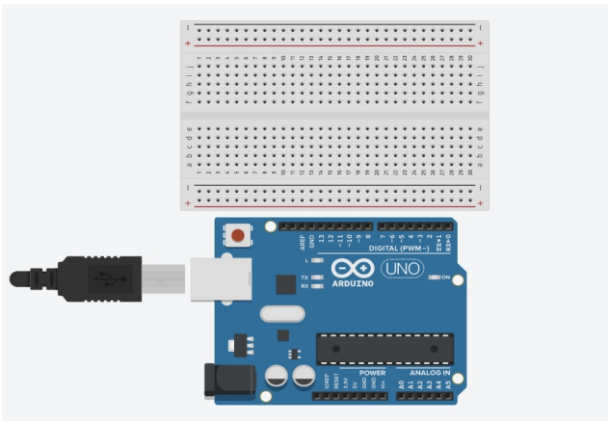
COMPONENTS REQUIRED

- Arduino Uno - 1
- Breadboard mini - 1
- Red, Green And Blue LEDs
- Plastic Dome
- Jumper Cable - 6

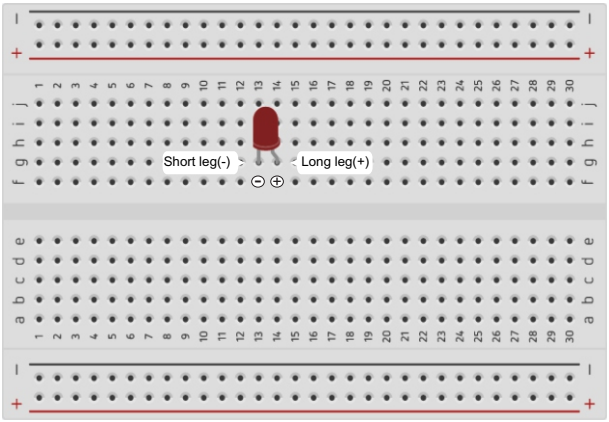
1 Take an Arduino Board.



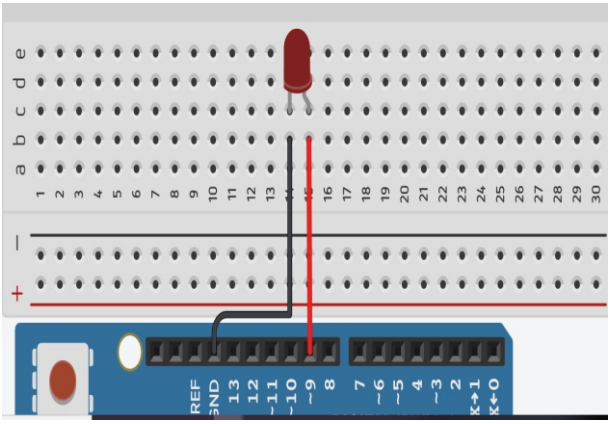
2 Place a breadboard as shown.



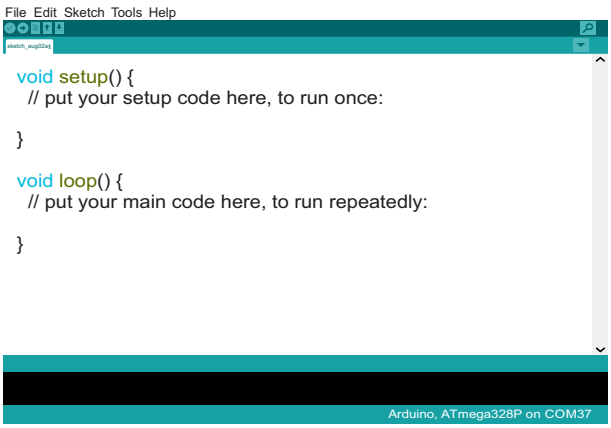
3 Take an LED and place it on the breadboard as shown



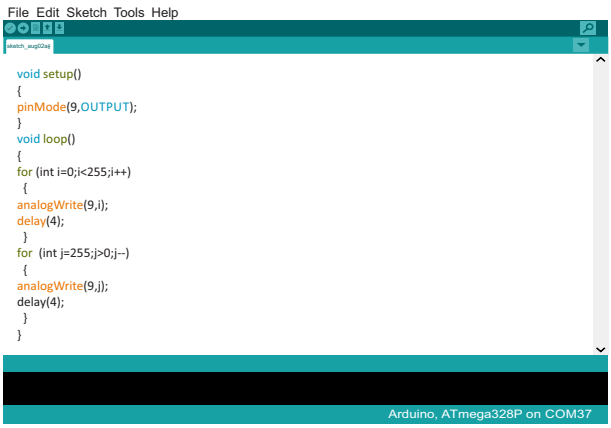
4 Connect the long leg of the LED to pin number 9 of the Arduino Board and connect the shorter leg to the GND pin of the Arduino board.



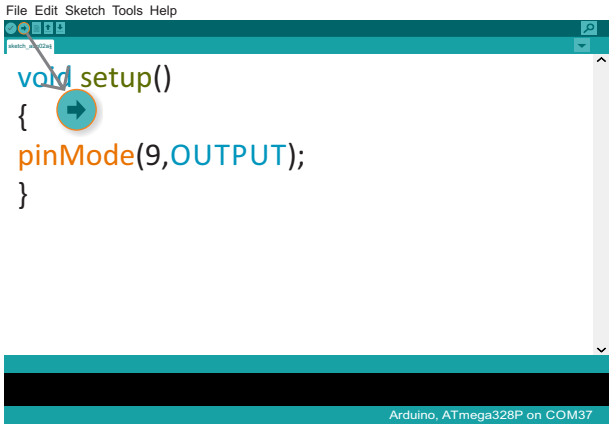
5 Open the Arduino IDE software



6 Write the code that has been given with the activity



7 Upload the code

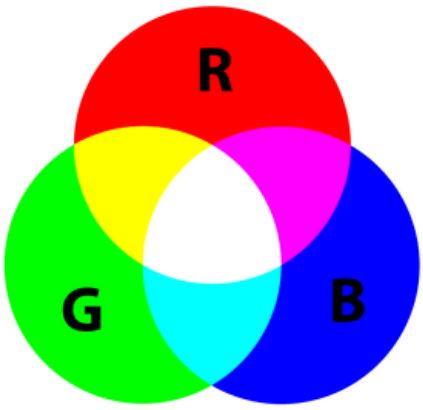


CODE

```
void setup()
{
pinMode(9,OUTPUT);
}
void loop()
{
for (int i=0;i<255;i++)
  {
analogWrite(9,i);
delay(4);
  }
for  (int j=255;j>0;j--)
  {
analogWrite(9,j);
delay(4);
  }
}
```

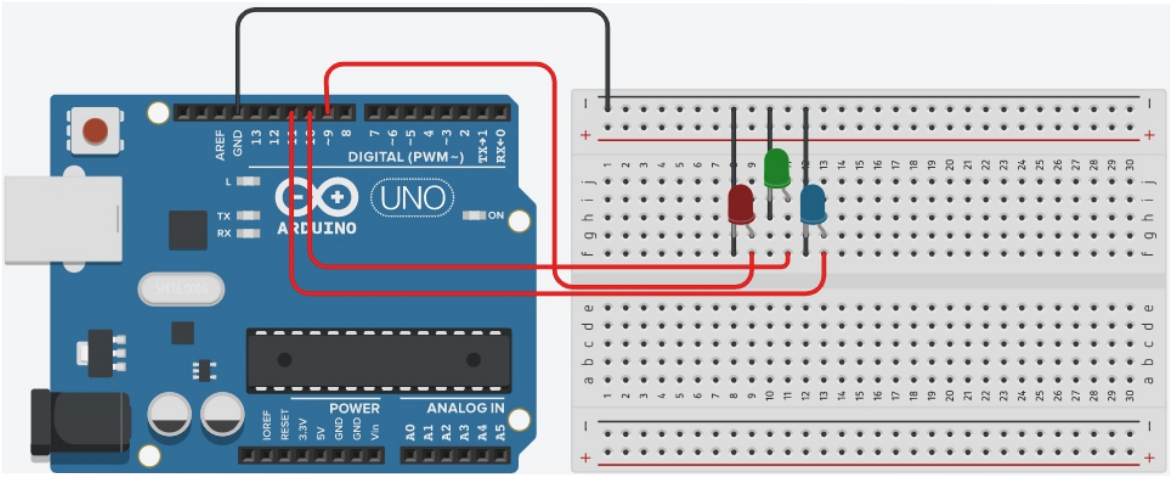
Concept

Mixing colors- To produce other colors, we can combine the three colors in different intensities. To adjust the intensity of each LED we can use a PWM signal. Our eyes see the result of the combination of colors, rather than the three colors individually. To have an idea on how to combine the colors, take a look at the following chart. This is the simplest color mixing chart, but gives you an idea how it works and how to produce different colors.



Activity:- 2 - Mood Lamp

In this activity we will learn to mix different colored LEDs to create a mood lamp that glows in various colors.



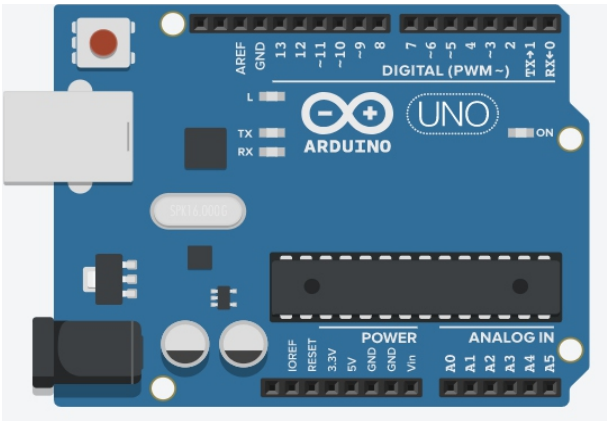
Definitions & Syntax:

With an RGB LED you can produce almost any color.
An **RGB LED** is a combination of 3 LEDs :

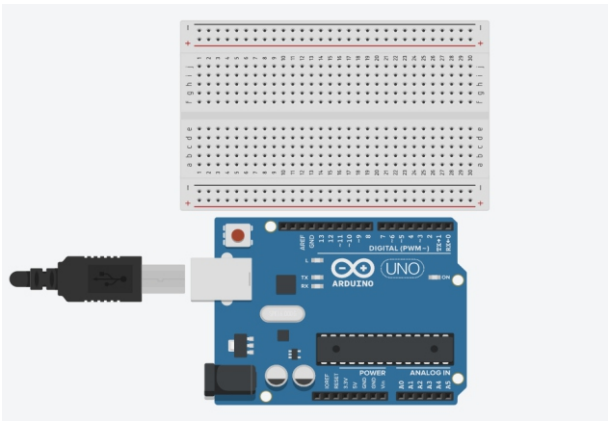
- 1x **R**ed LED
- 1x **G**reen LED
- 1x **B**lue LED

With RGB LEDs you can produce red, green, and blue light, and by configuring the intensity of each LED, you can produce other colors as well.

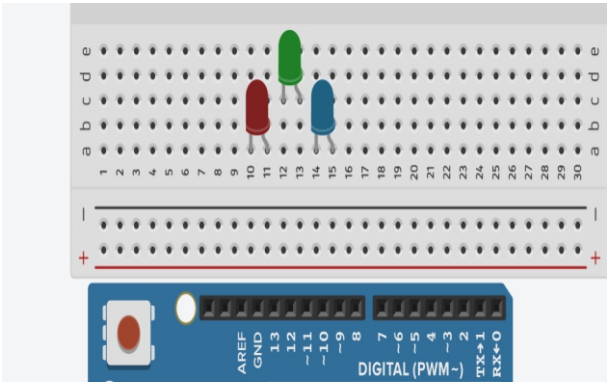
1 Take an Arduino Board



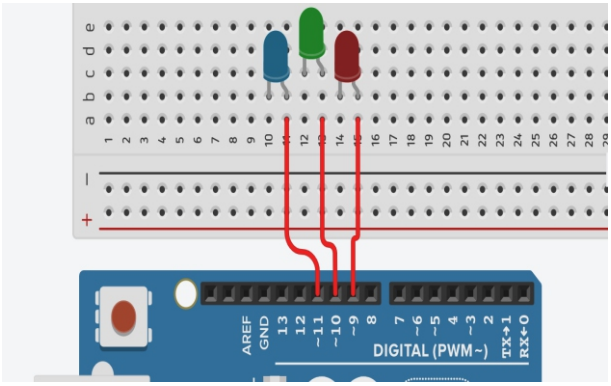
2 Place a breadboard as shown.



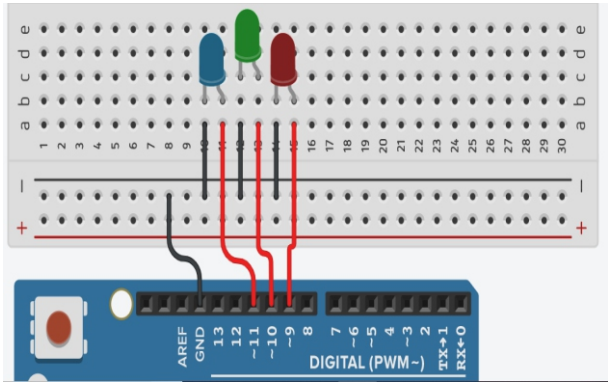
3 Take a Red LED, a Green LED and a Blue LED and place it on the breadboard as shown



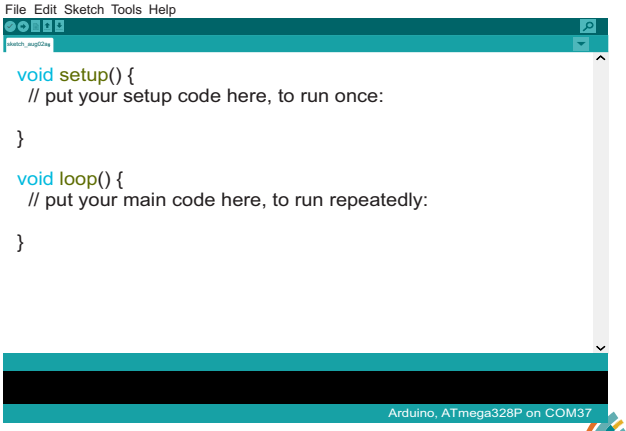
4 Connect the Red LEDs positive terminal to Arduino 's pin number 9, Green LEDs positive terminal to Arduino's pin number 10 and Blue LEDs positive terminal to Arduino's pin number 11.



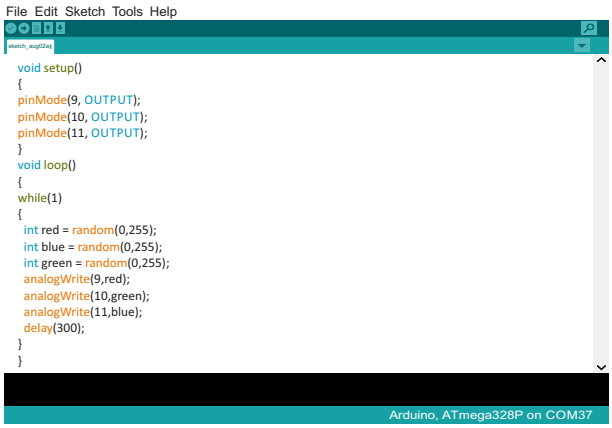
5 Connect the negative legs (short) of the LEDs to the negative rail of the breadboard. Then connect the GND pin of the Arduino to the Negative rail of the breadboard.



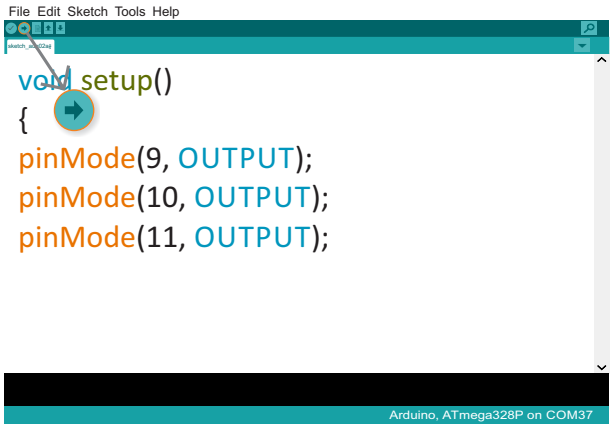
6 Open The Arduino IDE Software and connect your Arduino board to your computer



7 Write the code that is given with the activity



8 Upload the code



CODE

```
void setup()
{
pinMode(9, OUTPUT);
pinMode(10, OUTPUT);
pinMode(11, OUTPUT);
}
void loop()
{
while(1)
{
 int red = random(0,255);
 int blue = random(0,255);
 int green = random(0,255);
 analogWrite(9,red);
 analogWrite(10,green);
 analogWrite(11,blue);
 delay(300);
}
}
```

generates random number between two values given inside the brackets.like between 0-255 in this example.

Assessment

Tick The Correct One:

Q1. How many primary colors do we have?

- ☐ 3
- ☐ 13
- ☐ 7
- ☐ None of the above

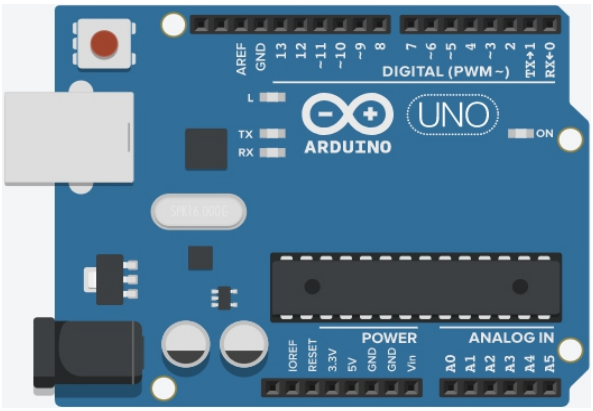
PWM(pulse width modulation)

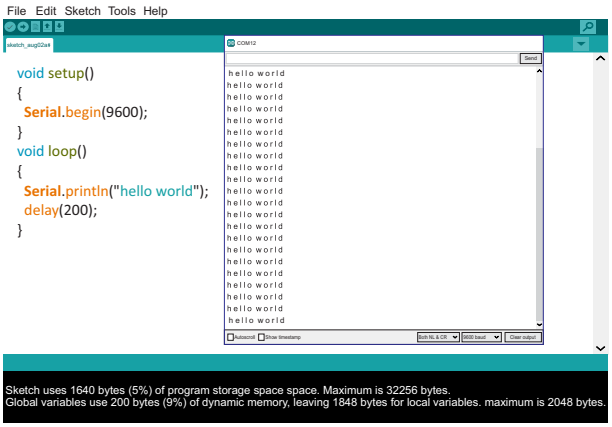
analogWrite

Concept

In this class, we will learn how to create a communication link between an Arduino and a computer and see the data sent by the Arduino on a computer screen

Activity:- 1 - Print Output On Serial Monitor





Definitions & Syntax:

Serial Monitor: Used for communication between the Arduino board and a computer or other devices.
You can use the Arduino IDE's built-in serial monitor to communicate with an Arduino board. Click the serial monitor button in the toolbar and select the same baud rate used in Serial.`begin()`

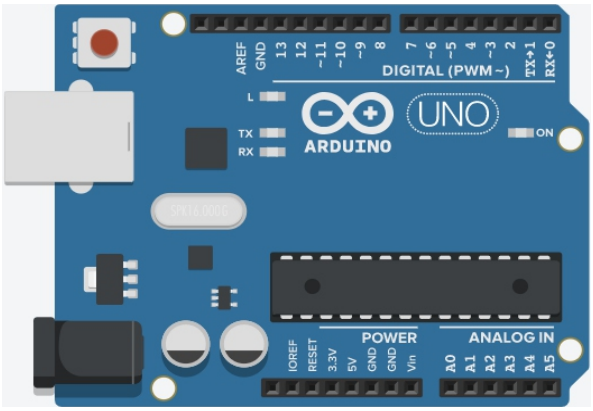
Serial.begin: Sets the data rate in bits per second (baud) for serial data transmission. For communicating with the computer, use one of these rates: 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600, or 115200.

Serial.print: Prints data to the serial port as human-readable ASCII text. `Serial.print("Hello world.")` gives "Hello world."

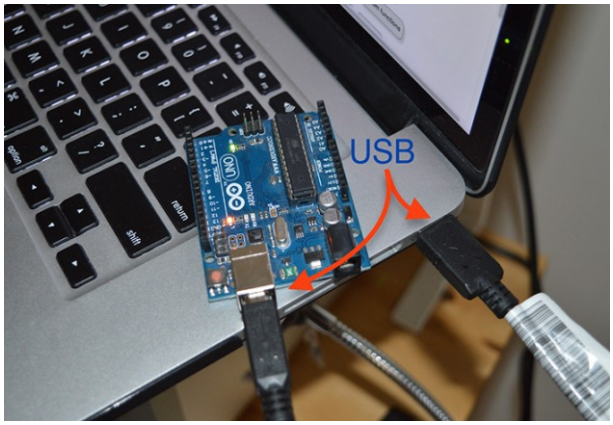
COMPONENTS REQUIRED

- Arduino Uno - 1
- Breadboard mini - 1
- Jumper Cable - 13
- LED - 4 of Different Colors
- Potentiometer - 1
- Pushbutton -1

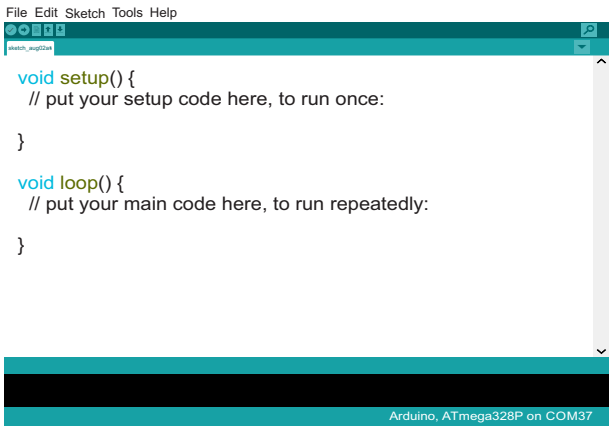
1 Take an Arduino Board



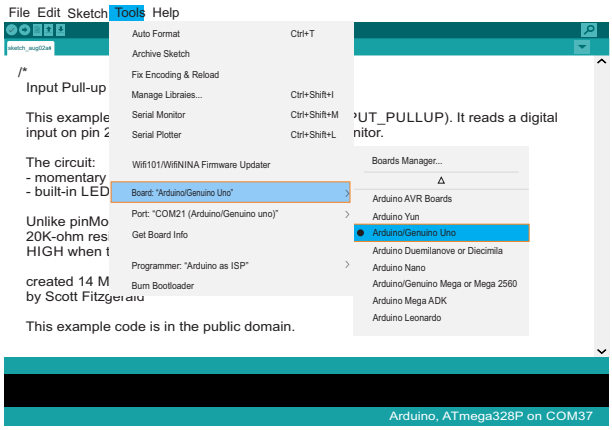
2 Connect Your Arduino board to the computer



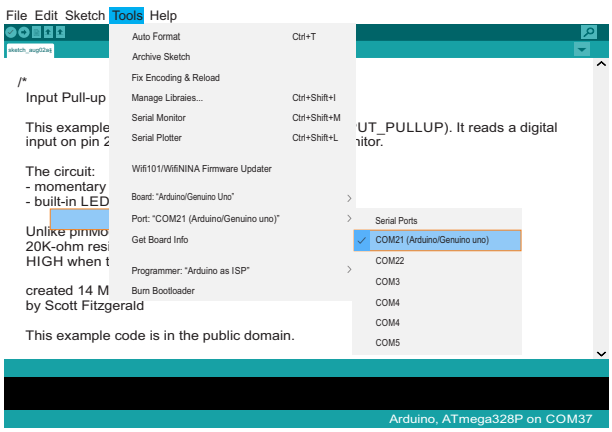
3 Open your Arduino IDE Software



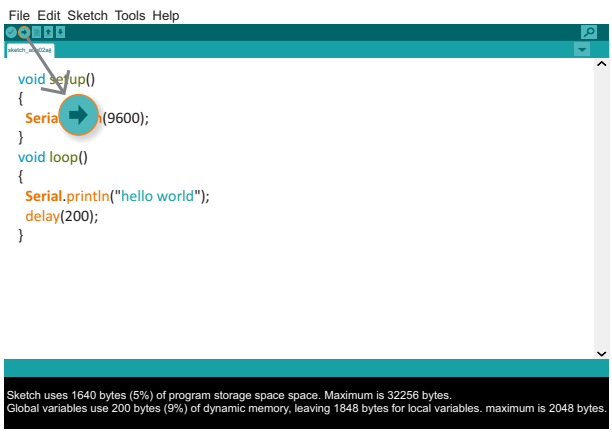
4 In the IDE, set the board and the port of the board you just connected, as shown



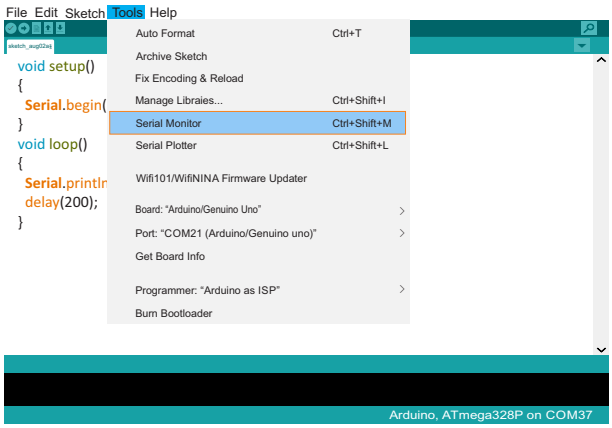
5 Copy the code which is given with the activity.



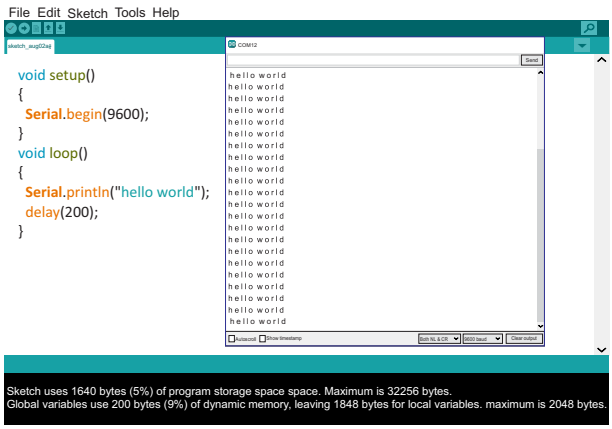
6 Upload the code



7 Take an Arduino Board



8 Look at the Serial monitor and observe the printed data.



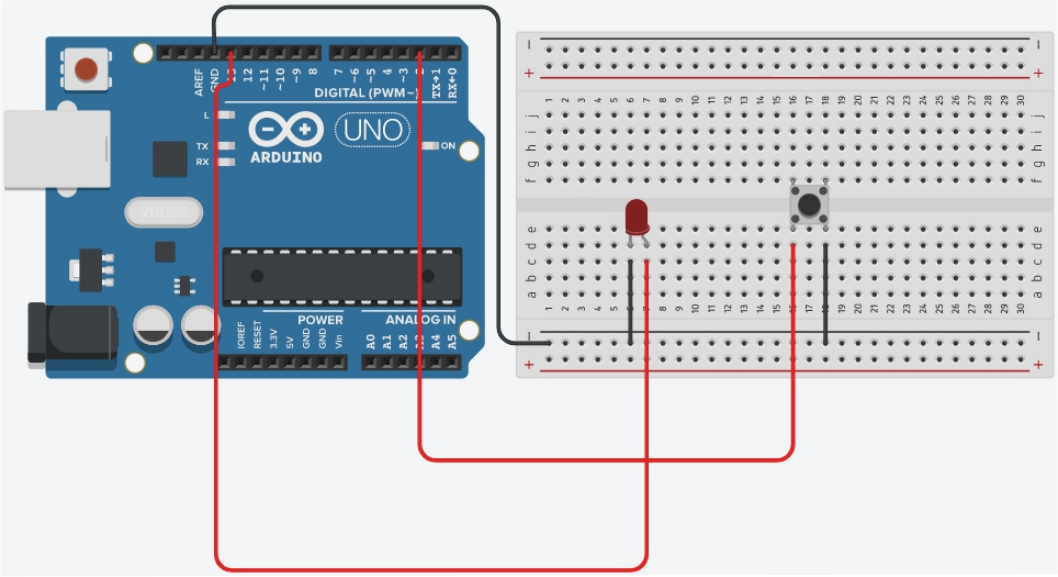
CODE:

```
void setup()
{
 Serial.begin(9600);
}
void loop()
{
 Serial.println("hello world");
 delay(200);
}
```

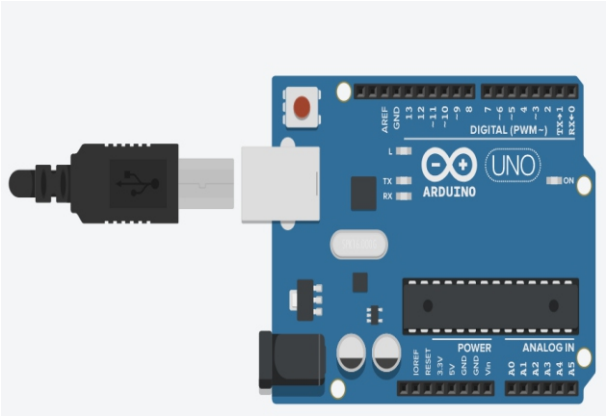
Concept

In this class you will learn to Serial print the values of the analog/digital input devices on the Serial monitor. Like if we press a button to check that if the button is pressed we will set the condition if the button is pressed print “button pressed”. We can Serial print the data/reading of any of the digital or analog sensor on the Serial monitor.

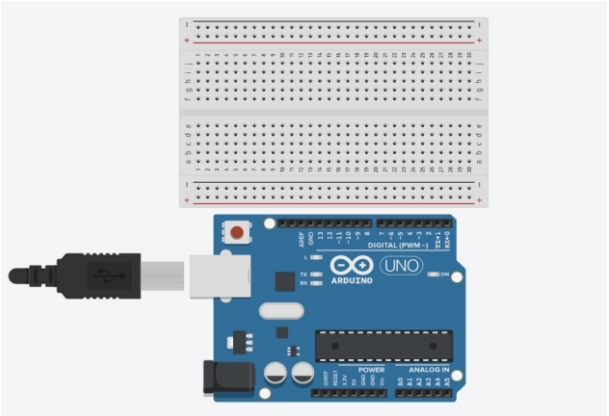
Activity:- 2 - Print The Inputs of Various Devices on the Serial Monitor



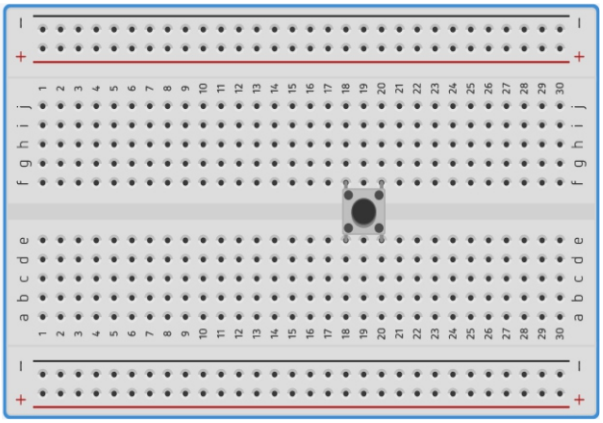
1 Take an Arduino Board



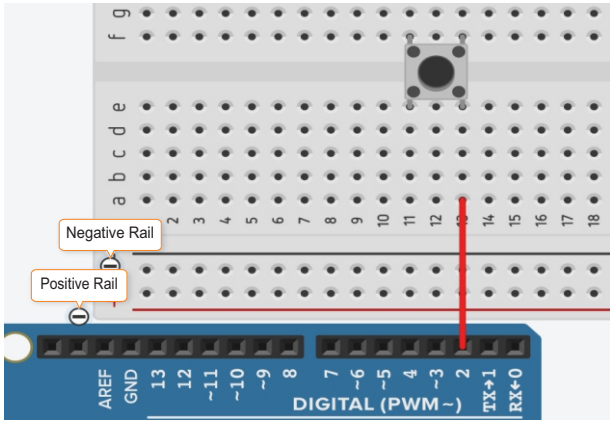
2 Place a breadboard along the Arduino board



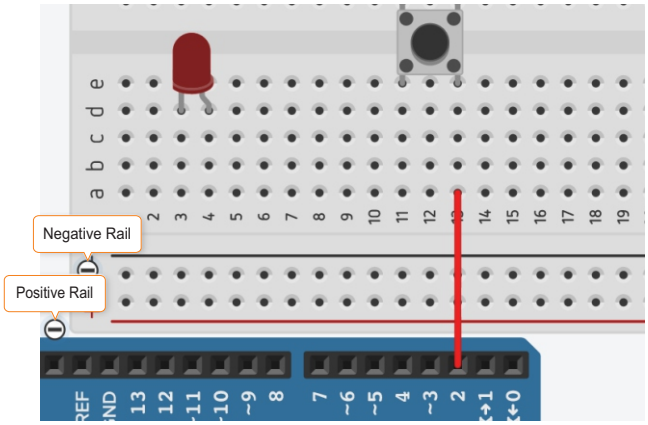
3 Place a button on the breadboard, as shown.



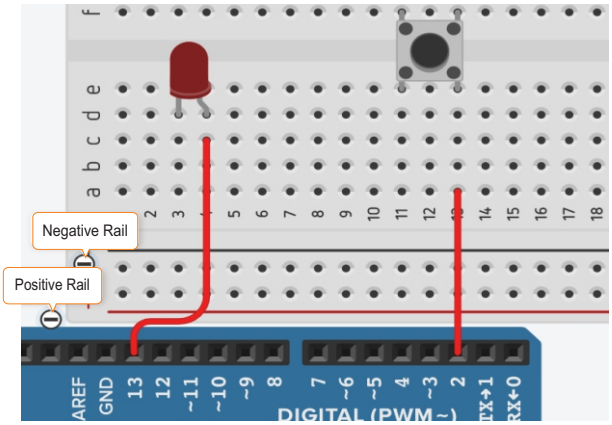
4 Connect one leg of the push button to the Arduino pin number 2 (or any pin that you want).



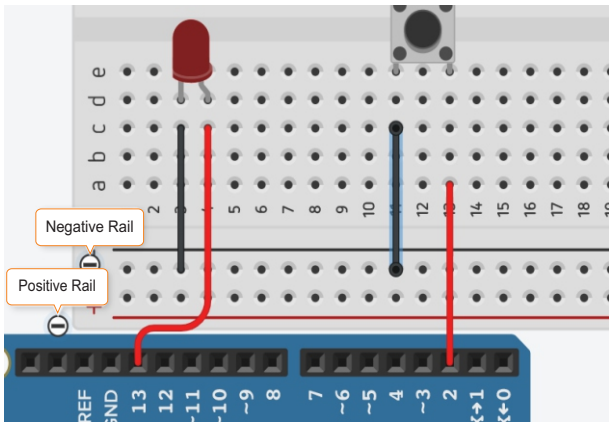
5 Place an LED on the breadboard as shown



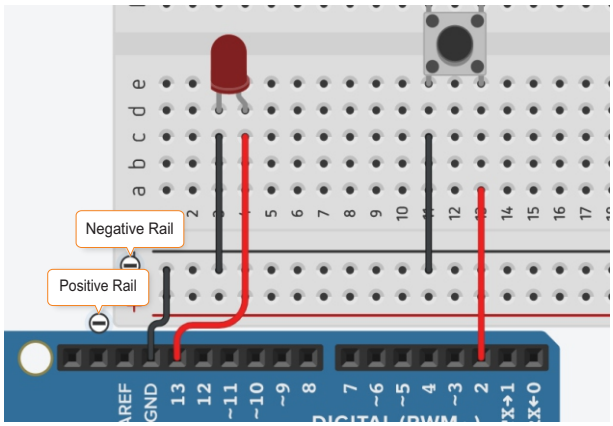
6 Connect the long leg(+ve) of the LED to pin number 13 on the Arduino board



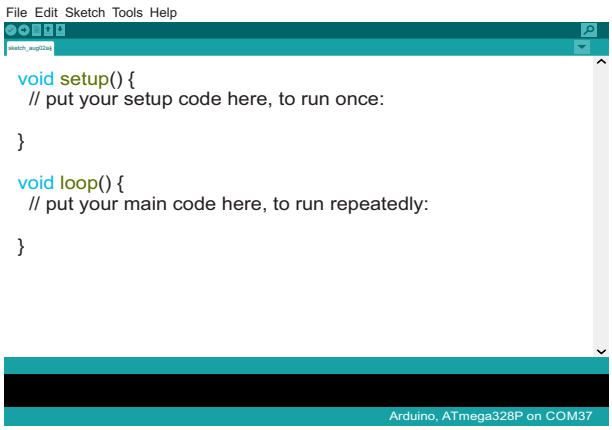
7 Connect the short leg of the LED (-ve) and one leg of the button to the negative power rail of the breadboard.



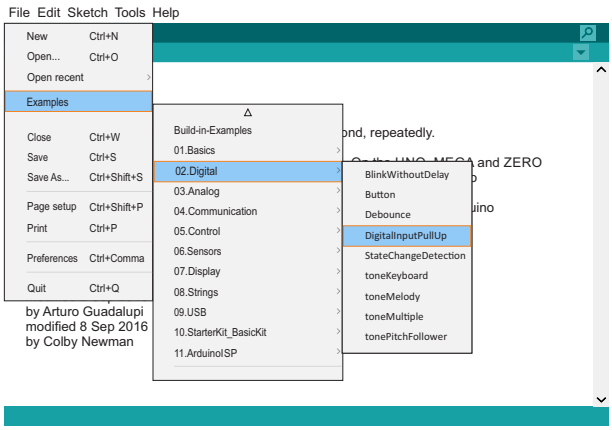
8 Connect the GND (ground) pin of the Arduino board to the negative rail of the breadboard



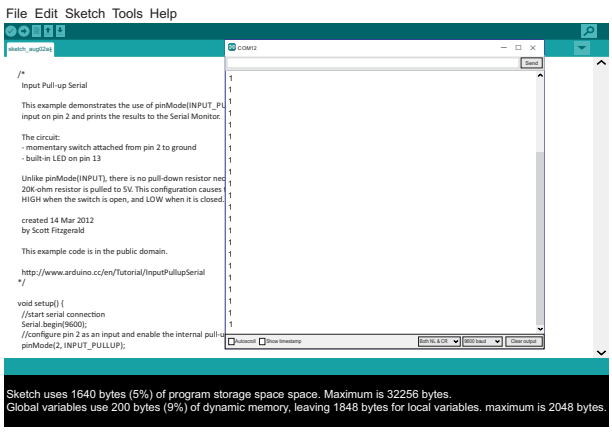
9 Open up the Arduino IDE software



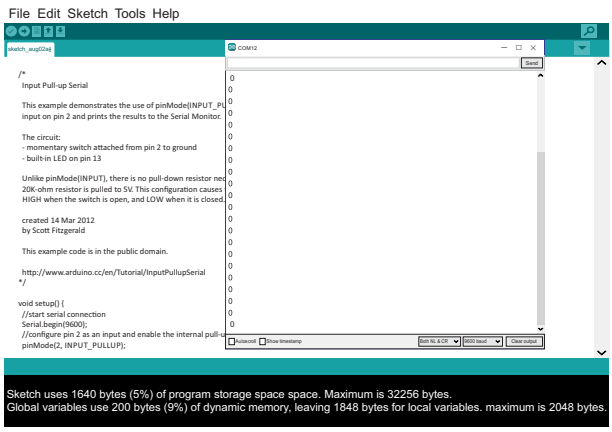
10 Open up the DigitalInputPullup example code as shown and upload the code.



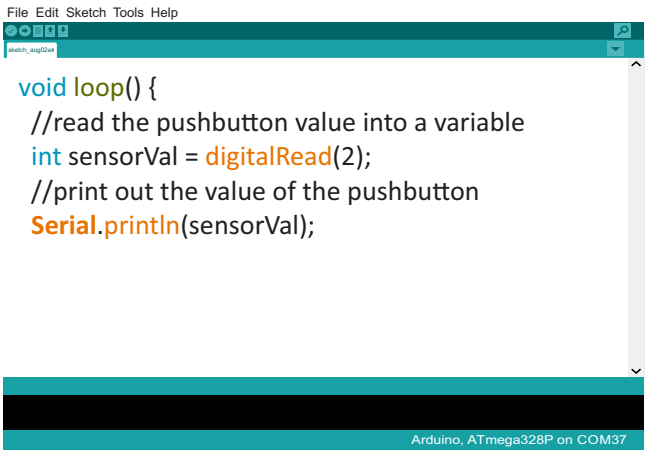
11 Open up the Serial Monitor and press the button.



12 Observe the change in values of the button when you press or release the button.



13 Notice that when you press the button, the state that is displayed on the serial monitor is 0, which means OFF, and when it is not pressed, it displays 1, which means its ON. Which is basically the reverse of what it is supposed to do. So we will have to change the code a little bit.



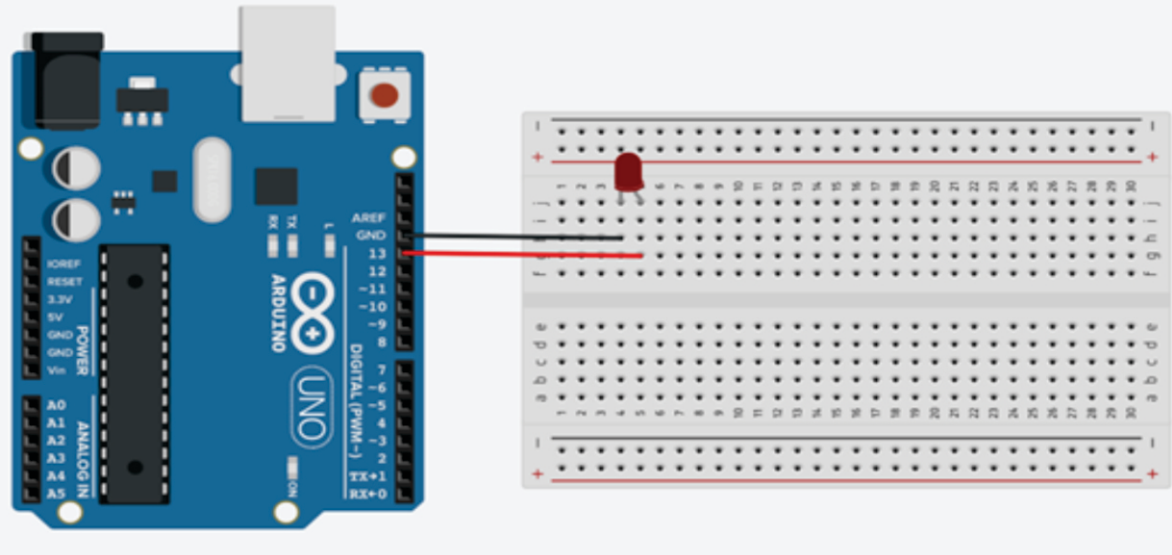
CODE:

```
void setup()
{
 //start serial connection
 Serial.begin(9600);
 //configure pin 2 as an input and enable the internal pull-up resistor
 pinMode(2, INPUT_PULLUP);
 pinMode(13, OUTPUT);
}
void loop()
{
 //read the pushbutton value into a variable
 int sensorVal = digitalRead(2);
 //print out the value of the pushbutton
 Serial.println(sensorVal);

 // Keep in mind the pull-up means the pushbutton's logic is inverted. It goes
 // HIGH when it's open, and LOW when it's pressed. Turn on pin 13 when the
 // button's pressed, and off when it's not:
 if (sensorVal == HIGH)
{
   digitalWrite(13, LOW);
 }
else
{
   digitalWrite(13, HIGH);
 }
}
```

Concept

In this experiment, we will learn how to turn on or off LEDs through a computer and the Serial Monitor. Serial Monitor is a built-in software in the Arduino IDE**.** We can send and receive data via the serial port on the Arduino board.

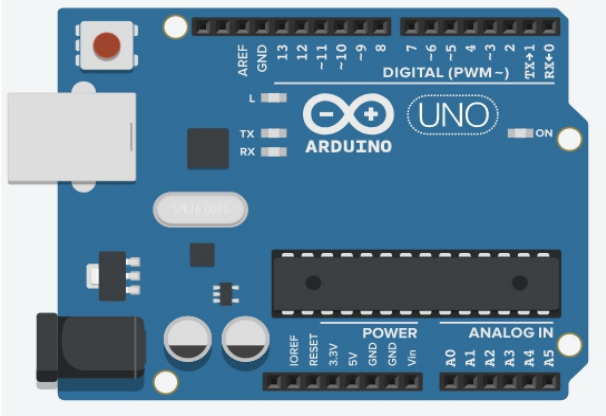
Activity:- 3 - Computer Controlled LED



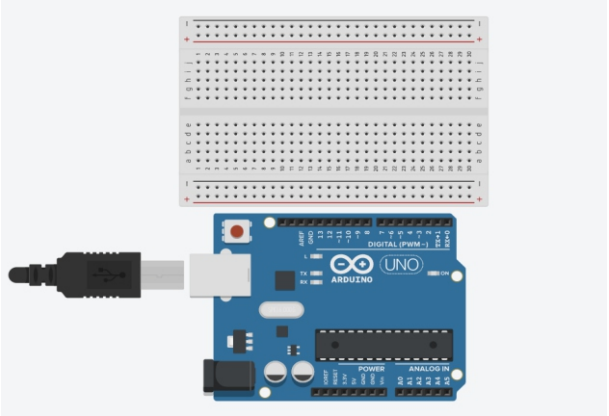
Definitions & Syntax:

Serial.parseInt	Looks for the next valid integer in the incoming serial stream.
Serial.available	Get the number of bytes (characters) available for reading from the serial port.
Syntax	*Serial*.available()
Returns	The number of bytes available to read.

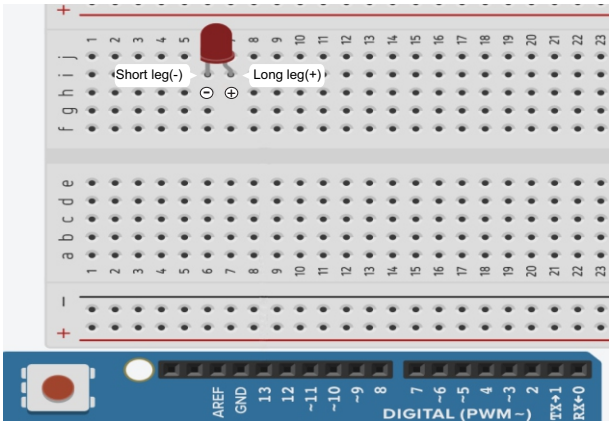
1 Take an UNO board



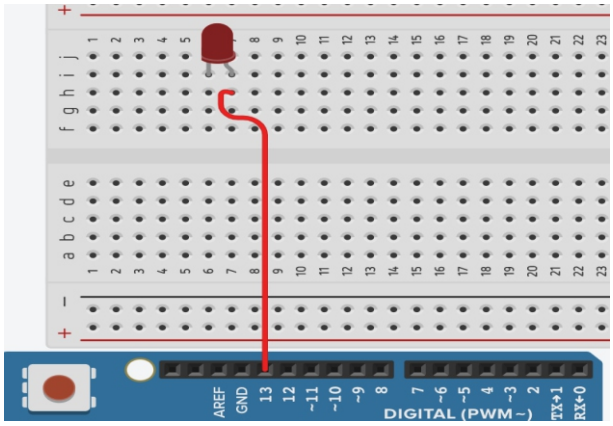
2 Place a breadboard as shown.



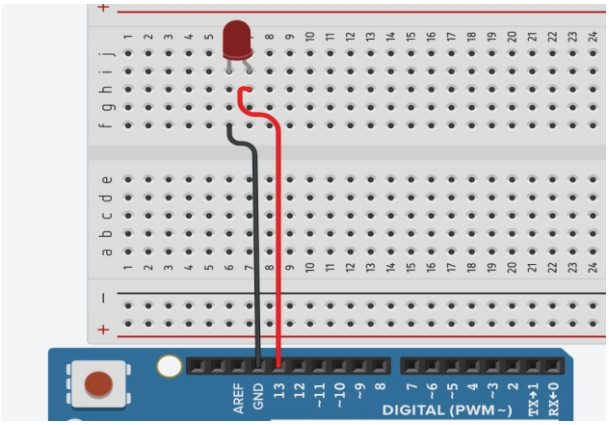
3 Take an LED and place it on the breadboard as shown



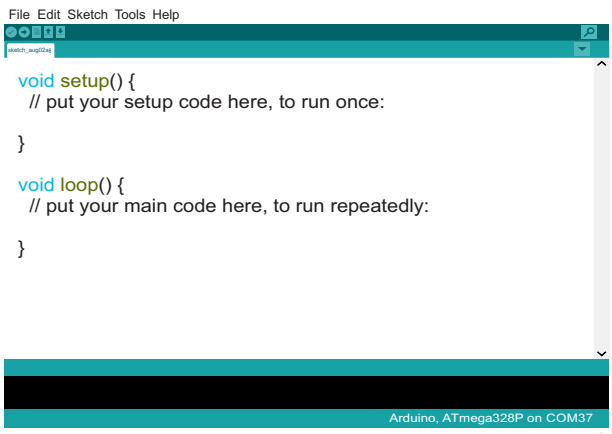
4 Take a red wire and connect it to the pin no. 13 on the Arduino board. Take the other end of the wire and connect it to the positive leg (long) of the LED as shown



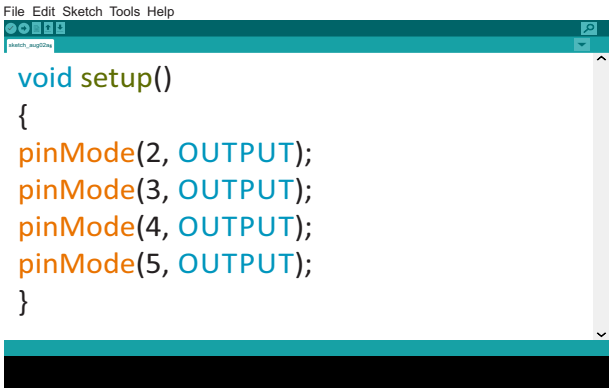
5 Now take a black wire and connect it to the pin named GND on the UNO board. Take the other end of the wire and connect it to the negative leg (short) of the LED.



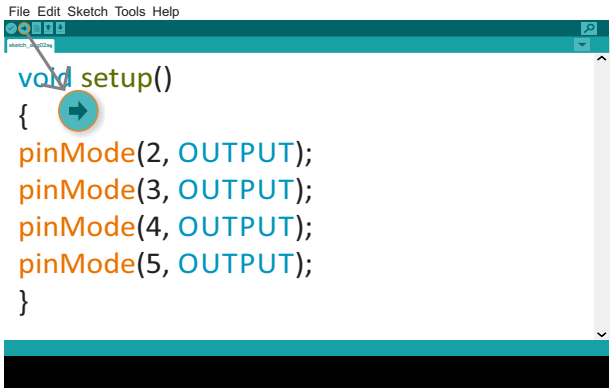
6 Open up the Arduino IDE Software.



7 Write the given code inside the code window



8 Upload the code onto your Arduino Board



CODE

```
void setup()
{
 pinMode(13, OUTPUT);
 Serial.begin(9600);
 while (!Serial);
 Serial.println("Input 1 to Turn LED on and 2 to off");
}
void loop()
{
if (Serial.available())
{
   int state = Serial.parseInt();
   if (state == 1)
{
    digitalWrite(13, HIGH);
    Serial.println("Command completed LED turned ON");
}
   if (state == 2)
{
    digitalWrite(13, LOW);
    Serial.println("Command completed LED turned OFF");
}
} }
```

Assessment

Tick The Correct One:

Q1. What is the speed at which Arduino communicates with the computer?

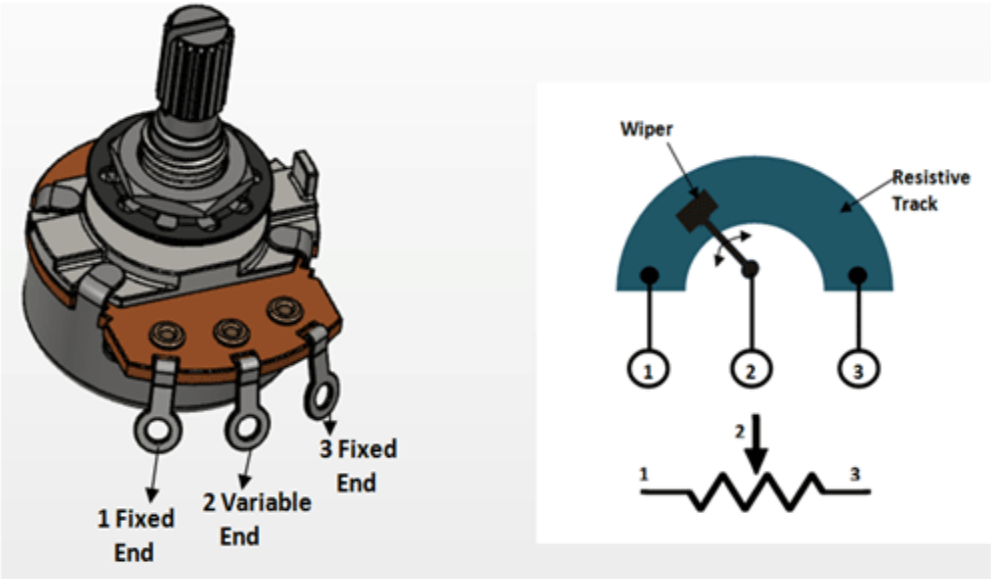
- ☐ 3450
- ☐ 115200
- ☐ 9600
- ☐ None of the above

Serial.print | Serial.parseInt | Serial.begin | Serial.available

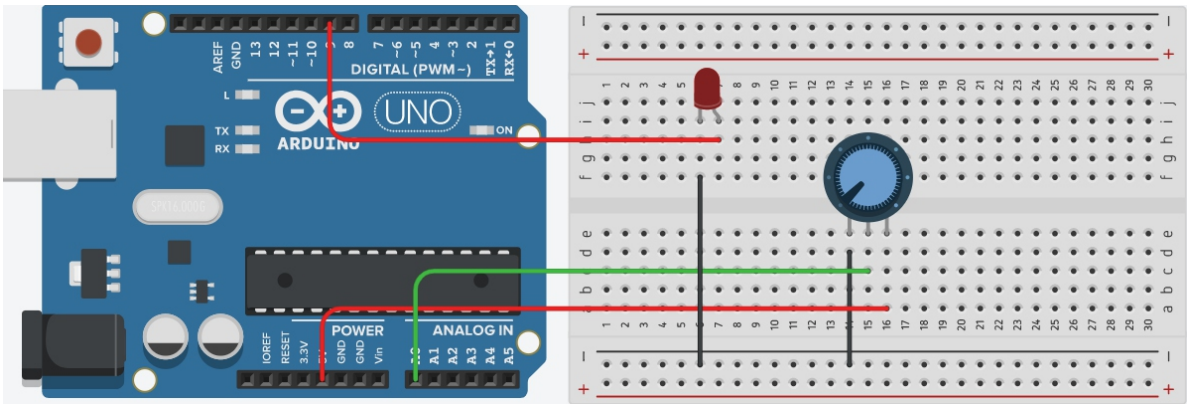
Concept

Analog Logic- Analog logic can be thought as a regulator of a fan, in which there are multiple different values of fan speed. Analog logic has nearly infinite voltage values between HIGH and LOW voltage states. Many electronic devices and sensors provide inputs analog style(i.e. constantly varying values), like temperature, humidity, force etc. In this activity, we will learn to obtain analog inputs from a potentiometer using an Arduino board.

Potentiometer- A potentiometer is a simple knob that provides a variable resistance, which we can read into the Arduino board as an analog value



Activity:- 1 - Potentiometer



Definitions & Syntax:

analogRead Reads the value from the specified analog pin.

Syntax **analogRead(pin)**

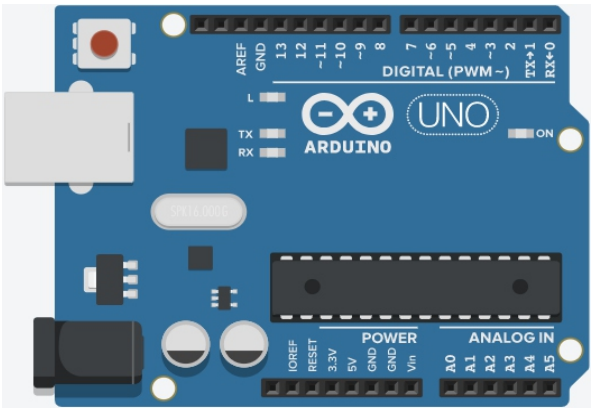
Parameters pin: the name of the analog input pin to read from (A0 to A5 on most boards, A0 to A6 on MKR boards, A0 to A7 on the Mini and Nano, A0 to A15 on the Mega).

Returns The analog reading on the pin. Although it is limited to the resolution of the analog to digital converter (0-1023 for 10 bits or 0-4095 for 12 bits).

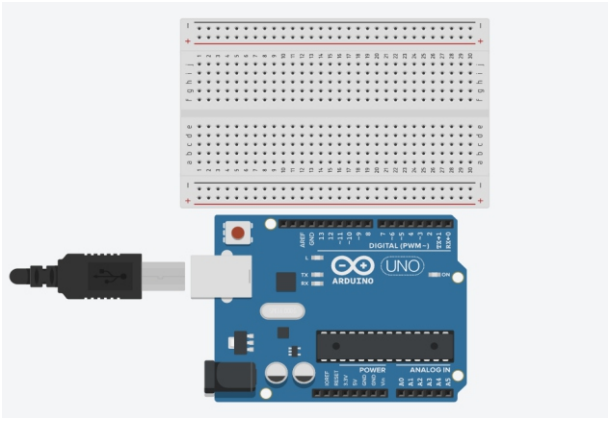
COMPONENTS REQUIRED

- Arduino Uno - 1
- Breadboard mini - 1
- Potentiometer - 3
- RGB Led - 1, LED - 1
- Resistor and Plastic Dome
- Jumper Cable - 11

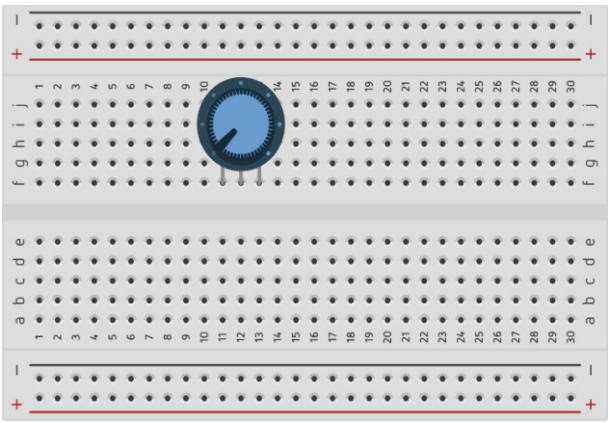
1 Take an Arduino Board



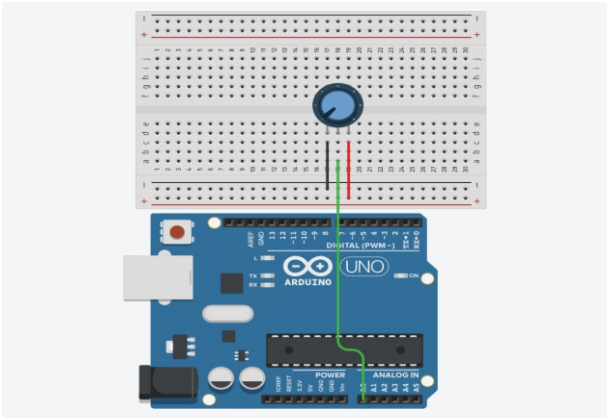
2 Place a breadboard along the Arduino board



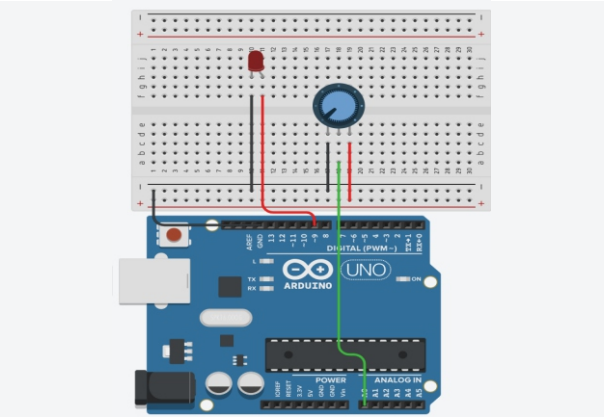
3 Place a potentiometer on the breadboard as shown



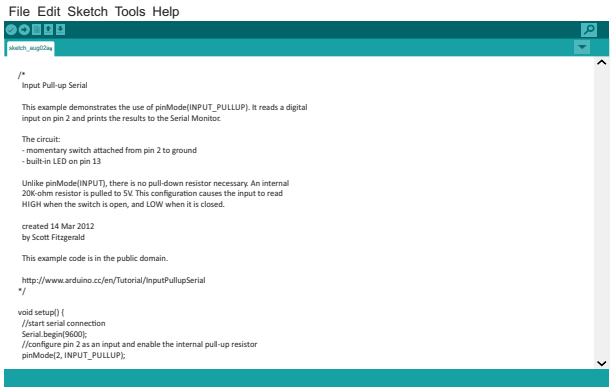
4 Connect the left leg of the potentiometer to the negative rail of the breadboard and the right leg of the potentiometer to the 5V pin of the Arduino. Connect the middle leg to the A0 pin of the Arduino.



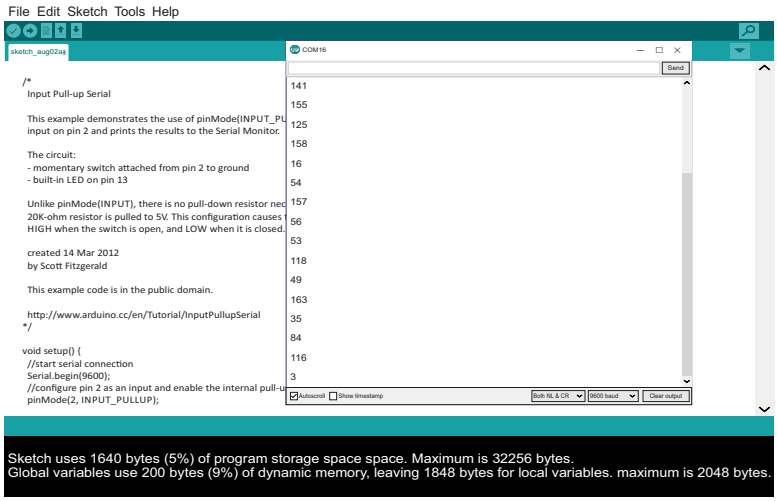
5 Place an LED on the breadboard and connect its positive leg to pin 9 of the Arduino UNO board and connect the negative leg to the negative rail of the breadboard.



6 Open the Arduino IDE Software and copy the code given with the activity



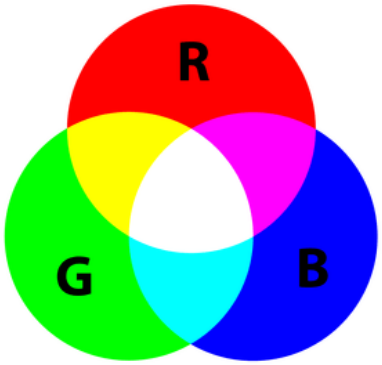
7 Upload the code and turn the potentiometer, and observe how the LED glows from off to full bright. Also, look at the values on the Serial Monitor.



CODE

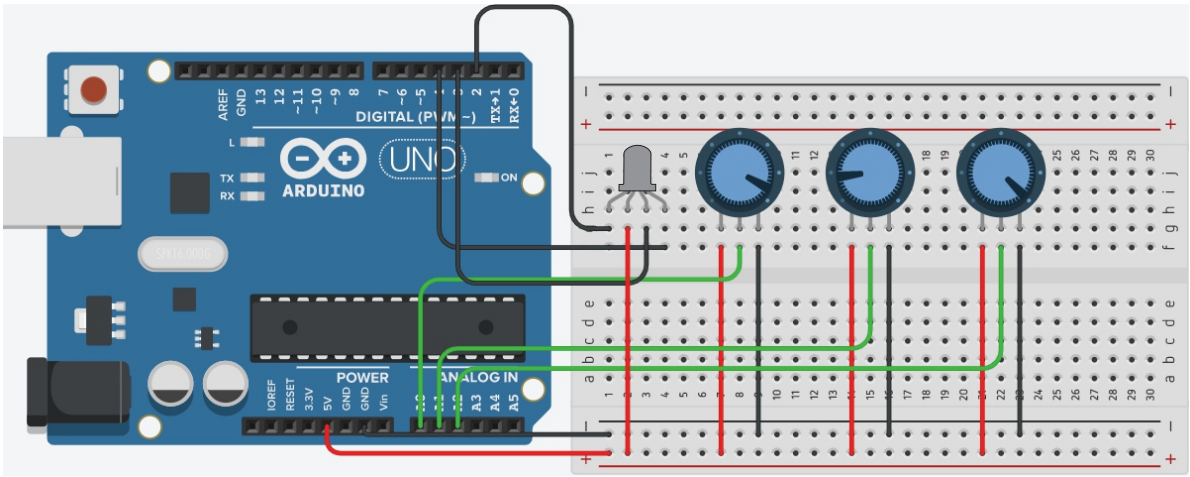
```
void setup()
{
pinMode(A0, INPUT);
Serial.begin(9600);
delay(100);
}
void loop()
{
int x=analogRead(A0);
Serial.println(x,DEC);
delay(100);
}
```

Mixing colors- To produce other colors, we can combine the three colors in different intensities. To adjust the intensity of each LED we can use a PWM signal. Our eyes see the result of the combination of colors, rather than the three colors individually. To have an idea on how to combine the colors, take a look at the following chart. This is the simplest color mixing chart, but gives you an idea how it works and how to produce different colors.



Activity:- 2 - Rgb Color Mixer

RGB Led In this activity, we will learn how to make different colors by controlling a RGB LED



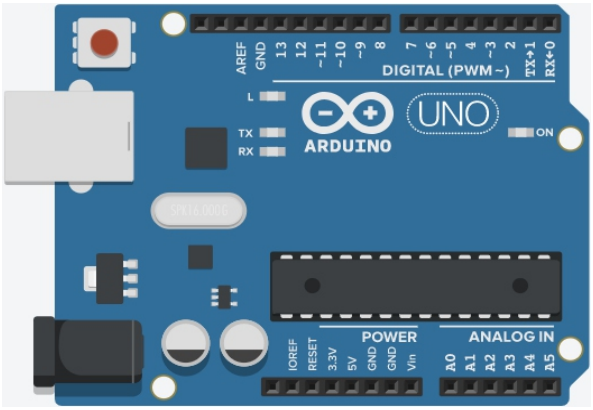
Definitions & Syntax:

With an RGB LED you can produce almost any color.
An RGB LED is a combination of 3 LEDs in just one package:

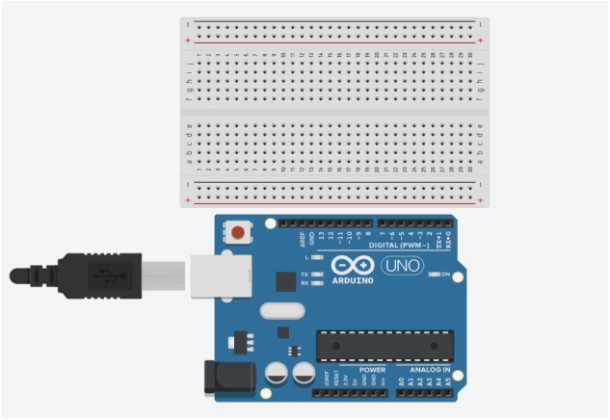
- 1x **R**ed LED
- 1x **G**reen LED
- 1x **B**lue LED

With an RGB LED you can, produce red, green, and blue light, and by configuring the intensity of each LED, you can produce other colors as well.

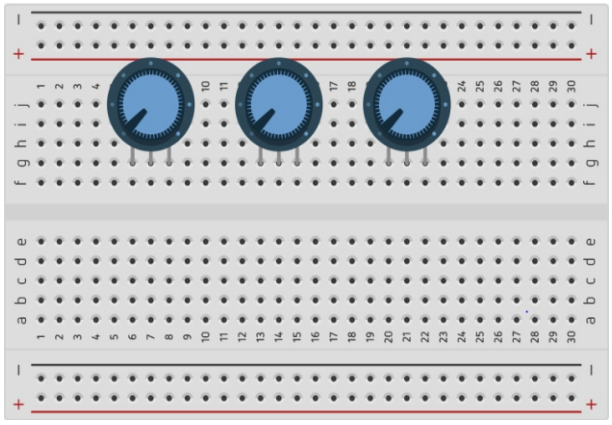
1 Take an Arduino Board



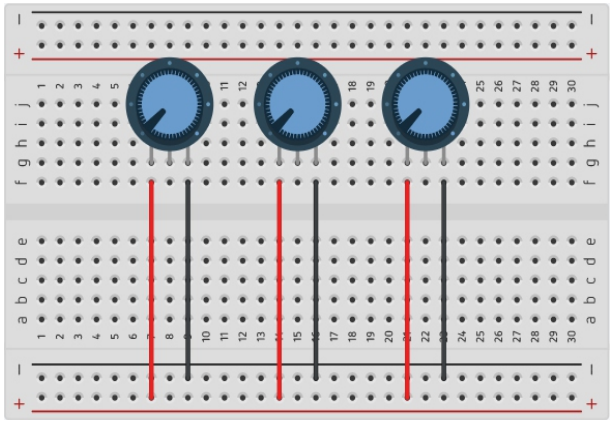
2 Place a breadboard along the Arduino board



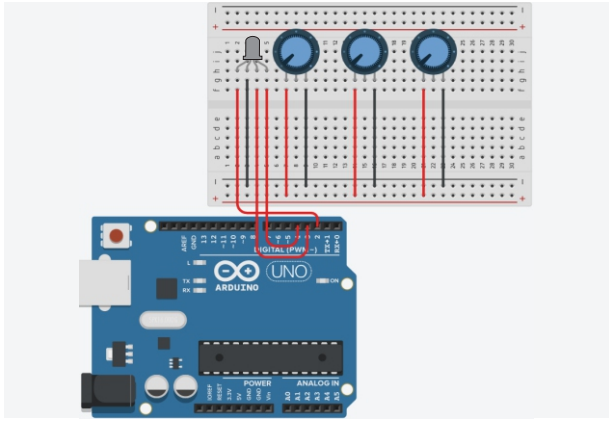
3 Place 3 potentiometers on the breadboard as shown.



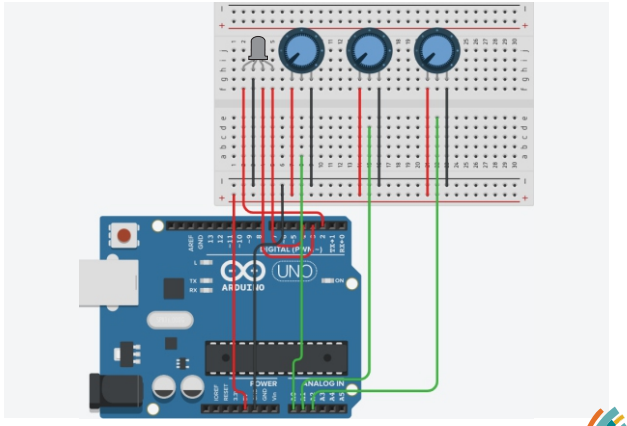
4 Connect the positive and the negative legs of the potentiometers to the positive and the negative rails of the breadboard as shown.



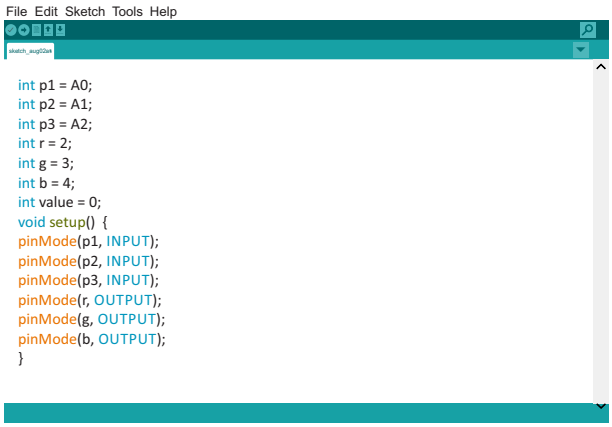
5 Connect a RGB LED on the breadboard. Connect the longest leg of the LED to negative rail of the breadboard and connect the remaining 3 legs of the LED to Arduino's pin number 2, 3 and 4 as shown.



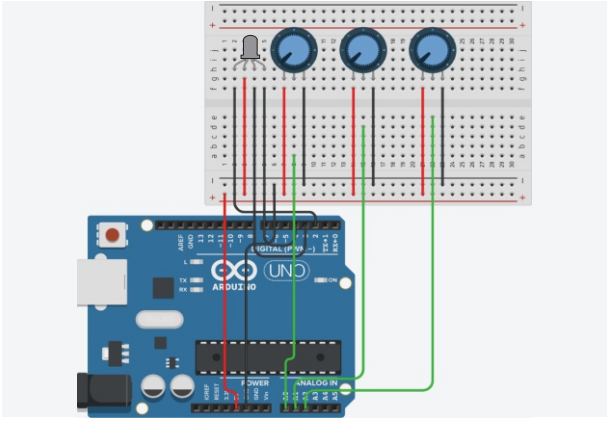
6 Connect the output pin of the potentiometers to the Analog pin A0, A1, A2 of the Arduino board. Connect the negative and the positive rail of the breadboard to the GND and 5v pin of the Arduino board respectively.



7 Open the Arduino IDE Software, Write the given code inside the code window.



8 Upload the code and rotate the potentiometers to create different colours on the LED.



CODE

```
int p1 = A0;
int p2 = A1;
int p3 = A2;
int r = 2;
int g = 3;
int b = 4;
int value = 0;
void setup()  {
pinMode(p1, INPUT);
pinMode(p2, INPUT);
pinMode(p3, INPUT);
pinMode(r, OUTPUT);
pinMode(g, OUTPUT);
pinMode(b, OUTPUT);
}
void loop()   {
value = analogRead(p1);
analogWrite(r, value);
value = analogRead(p2);
analogWrite(g, value);
value = analogRead(p3);
analogWrite(b, value);
}
```

Assessment

Tick The Correct One:

Q1. What are the highest and lowest values of analog input in Arduino?

- ☐ 0-255
- ☐ 5-105
- ☐ 0-1023
- ☐ None of the above

analogRead | Serial.begin | Serial.print | analogWrite


ARDUINO