Geetanjali Institute of Technical Studies

(Approved by AICTE, New Delhi and Affiliated to Rajasthan Technical University Kota (Raj.))

DABOK, UDAIPUR, RAJASTHAN 313022

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

B. Tech - VII SEMESTER

तमसो मा ज्योतिर्गमय्

GITS

Darkness to Light

ACADEMIC YEAR – 2022-23

INTERNET OF THINGS LABORATORY MANUAL

7CS4-21

Submitted To:
Dr. Vijendra K. Maurya

Submitted by

Name:XXXXXXXX

Roll No:XXXXXXX

Section: A

INDEX – LAB MANUAL

VISSION & MISSION OF INSTITUTE

INSTITUTE VISION

TO ACHIEVE EXCELLENCE IN TECHNICAL AND MANAGEMENT EDUCATION THROUGH QUALITY TEACHING, RESEARCH AND INNOVATION.

INSTITUTE MISSION

TO PROVIDE A CONDUCIVE ENVIRONMENT IN ORDER TO PRODUCE SOCIALLY RESPONSIBLE AND PRODUCTIVE PROFESSIONALS.

VISION & MISSION OF DEPARTMENT

VISION

To provide quality education through research & innovation to cater the need of industry & society.

MISSION

To nurture knowledge of students in theoretical and practical aspects in collaboration with industries.

To inculcate the students towards research and innovation to fulfill the need of industry & society.

To develop socially responsible professionals with values and ethics.

PROGRAMME EDUCATIONAL OBJECTIVES (PEOs)

The Programme Educational Objectives of the programme offered by the department are listed below:

- **PEO1:** Deliver comprehensive knowledge and skills for successful career in the industries.
- **PEO2:** To Provide conducive environment for becoming a successful entrepreneur and life-long learning.
- **PEO3:** Inculcate research through innovative solution of the real-life problems with help of industries.

PROGRAM SPECIFIC OUTCOMES (PSO's)

- **PSO1: Professional Skills:** The ability to understand, analyze and develop electronic systems in the areas related to hardware and software development, communication systems and networking for efficient design of electronic-based systems of varying complexity.

- **PSO2: Problem-Solving Skills:** The ability to apply standard practices and strategies in electronic system project development on both hardware and software environments to deliver a quality product for business success.

- **PSO3: Successful Career and Entrepreneurship:** The ability to employ modern electronic solutions on different platforms, in creating innovative career paths to be an entrepreneur, and a zest for higher studies.

PROGRAMME OUTCOMES (POs)

A student will develop:

1. **ENGINEERING KNOWLEDGE:** An ability to apply knowledge of Mathematics, Science and Engineering Fundamentals in Electronics and Communication Engineering.
2. **PROBLEM ANALYSIS:** An ability to analyze and interpret data by designing and conducting experiments. Develop the knowledge of developing algorithms, designing, implementation and testing applications in electronics and communication related areas.
3. **DESIGN/ DEVELOPMENT OF SOLUTION:** An ability to Design a system Component or process to meet desired needs within realistic constraints such as economic, environmental, social, political, ethical, health and safety, manufacturability, and sustainability.
4. **CONDUCTION OF INVESTIGATION OF COMPLEX PROBLEMS:** An ability to Identify, formulate and solve engineering problems.
5. **MODERN TOOL USAGE:** An ability to use the techniques, skills and modern engineering tools necessary for engineering practice.
6. **THE ENGINEERING AND SOCIETY:** Broad education necessary to understand the impact of engineering solutions in a global, economic, environmental and societal context.
7. **ENVIRONMENT & SUSTAINABILITY:** Understand the impact of professional engineering solution in societal and environmental contexts, and demonstrate the knowledge of, and need of sustainable development.
8. **ETHICS:** An ability to understand the professional, social and ethical responsibility.
9. **INDIVIDUAL AND TEAM WORK:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **COMMUNICATION:** An ability to Communicate effectively in order to succeed in their profession such as, being able to write effective reports and design documentation, make effective presentations.
11. **PROJECT MANAGEMENT & FINANCE:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in team, to manage projects and in multidisciplinary environment.
12. **LIFE-LONG LEARNING:** Recognize the need and an ability to engage in life-long learning.

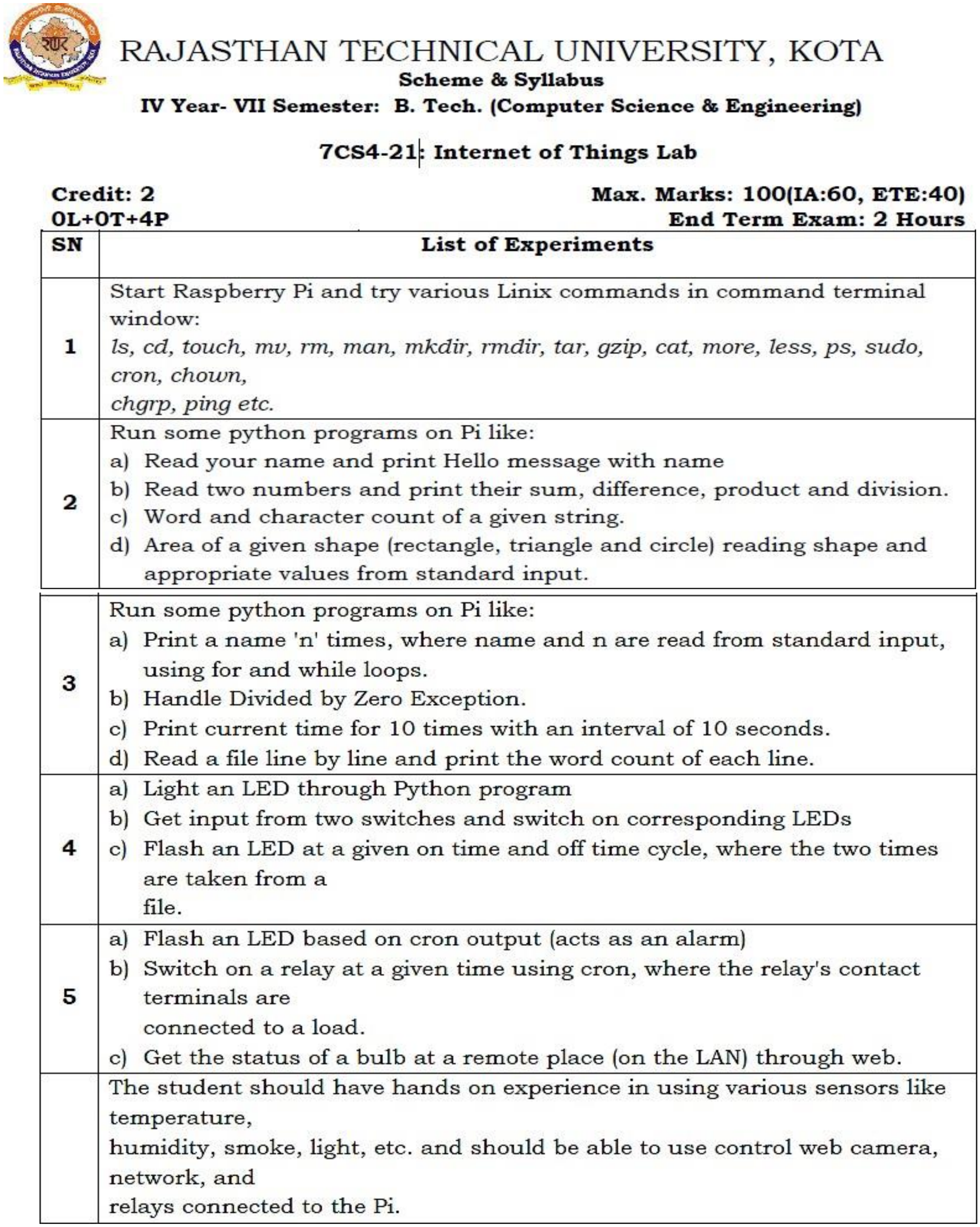
COURSE OUTCOMES (COs)

CO1	Understand the definition and significance of the programming language which used for Internet of Thing
CO2	Understand and explore the different platform/software/hardware used for IOT
CO3	Apply the knowledge of Basic IOT techniques and protocols to develop innovative applications with number of sensors and components.

COs MAPPING WITH POs AND PSOs

Course Outcome	**PO 1**	**PO 2**	**PO 3**	**PO 4**	**PO 5**	**PO 6**	**PO 7**	**PO 8**	**PO 9**	**PO 10**	**PO 11**	**PO 12**	**PSO 1**	**PSO 2**	**PSO 3**
CO1	2	2	2	2	1	0	1	1	2	0	0	1	3	2	1
CO2	3	2	3	2	1	0	1	1	2	1	1	1	3	3	1
CO3	3	3	3	2	2	2	1	1	2	1	2	1	3	3	3

COURSE SYLLABUS



RAJASTHAN TECHNICAL UNIVERSITY, KOTA

Scheme & Syllabus

IV Year- VII Semester: B. Tech. (Computer Science & Engineering)

7CS4-21: Internet of Things Lab

Credit: 2 **Max. Marks: 100(IA:60, ETE:40)**

0L+0T+4P **End Term Exam: 2 Hours**

SN	**List of Experiments**
1	Start Raspberry Pi and try various Linix commands in command terminal window: *ls, cd, touch, mv, rm, man, mkdir, rmdir, tar, gzip, cat, more, less, ps, sudo, cron, chown, chgrp, ping etc.*
2	Run some python programs on Pi like: a) Read your name and print Hello message with name b) Read two numbers and print their sum, difference, product and division. c) Word and character count of a given string. d) Area of a given shape (rectangle, triangle and circle) reading shape and appropriate values from standard input.
3	Run some python programs on Pi like: a) Print a name 'n' times, where name and n are read from standard input, using for and while loops. b) Handle Divided by Zero Exception. c) Print current time for 10 times with an interval of 10 seconds. d) Read a file line by line and print the word count of each line.
4	a) Light an LED through Python program b) Get input from two switches and switch on corresponding LEDs c) Flash an LED at a given on time and off time cycle, where the two times are taken from a file.
5	a) Flash an LED based on cron output (acts as an alarm) b) Switch on a relay at a given time using cron, where the relay's contact terminals are connected to a load. c) Get the status of a bulb at a remote place (on the LAN) through web.
	The student should have hands on experience in using various sensors like temperature, humidity, smoke, light, etc. and should be able to use control web camera, network, and relays connected to the Pi.

TEXT / REFERENCE BOOKS

- Internet of Things connecting objects to the web, by Hakima Chaouchi, Wiley.
- Internet of Things (A Hands-on-Approach) by Arshdeep Bhaga and Vijay Madisetti.
- The Internet of Things (MIT Press) by Samuel Greengard.
- The Internet of Things (Connecting objects to the web) by Hakima Chaouchi (Wiley Publications).
- RFID and the Internet of Things, by Herve chabanne, Wiley
- Introduction to Computation and Programming using Python, by John Guttag
- Fundamentals of Navigation and Inertial Sensors, By Amitava Bose, K N Bhat and Thomas Kurian
- Internet of Things: A Hands-on Approach, By Arshdeep Bahga and Vijay Madisetti

VIDEO REFERENCE

- https://nptel.ac.in/courses/106/105/106105166/

IV Year- VII Semester: B. Tech. (Computer Science & Engineering)
Scheme & Syllabus of 4thYear B. Tech. (CS) for students admitted in Session 2017-18

7CS4-21: Internet of Things Lab

Credit: 2

Max. Marks: 100(IA:60, ETE:40) 0L+0T+4P End Term Exam: 2 Hours

Sr. No.	LIST OF EXPERIMENT	CO MAPPING
1.	Study and Understand Python with the help of program • Write a Program for arithmetic operation in Python. • Write a Program for looping statement in Python.	CO1
2.	Run some programs on IOT simulator - Cupcarbon like: • Read Hello message • Read two numbers and print their sum, difference, product and division.	CO1, CO2
3.	Study and Install IDE of Arduino and different types of Arduino and • Write program using Arduino IDE for Blink LED • Get input from two switches and switch on corresponding LEDs	CO2
4.	Write Program for RGB LED using Arduino.	CO2
5.	Study the Temperature sensor and Write Program foe monitor temperature using Arduino.	CO3
6.	Demonstrate and program for PIR Motion Sensor (Digital Input).	CO3

Prerequisites:

- Students are expected to have knowledge of basic component of electronics.
- Students are expected to have knowledge different Sensor

Software requirements:

- Python IDLE
- CupCarbon
- TinkerCad

PRACTICALS BEYOND RTU SYLLABUS

Sr. No.	List of Experiment	CO Mapping
1.	Demonstrate and program for Smoke Sensor.	CO1, CO2, CO3
2.	Demonstrate and program for Basic Calculator	CO1, CO2, CO3

EXPERIMENT NO. 1

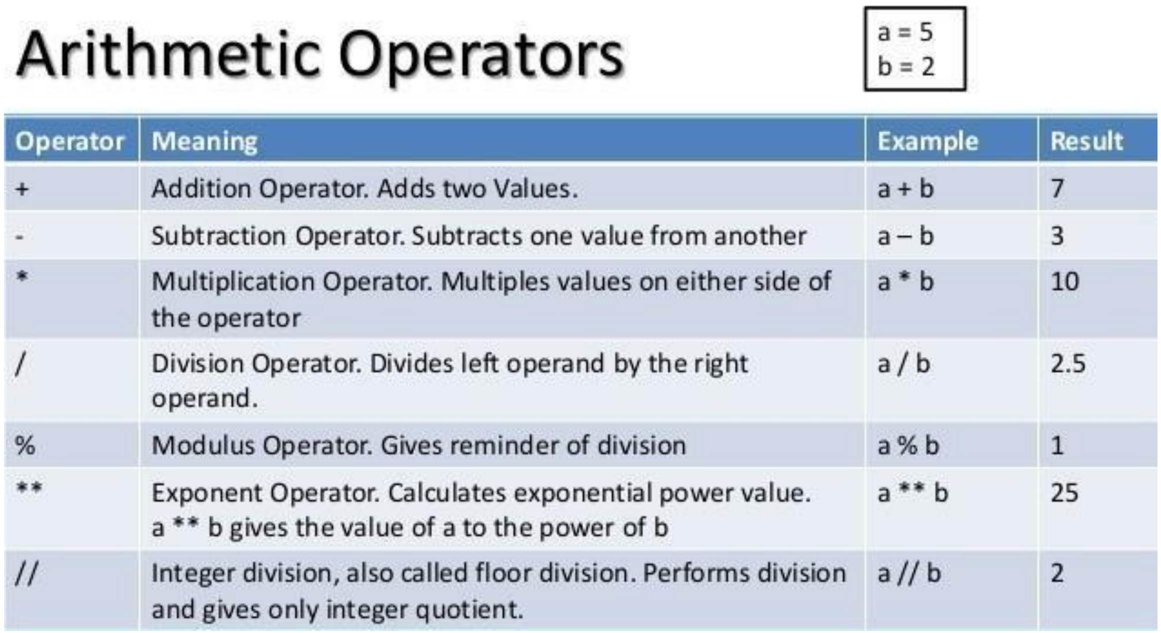
AIM: Study and Understand Python with the help of program.

a) Write a Program for arithmetic operation in Python.

b) Write a Program for looping statement in Python.

PROGRAM:

a) Arithmetic operation in Python



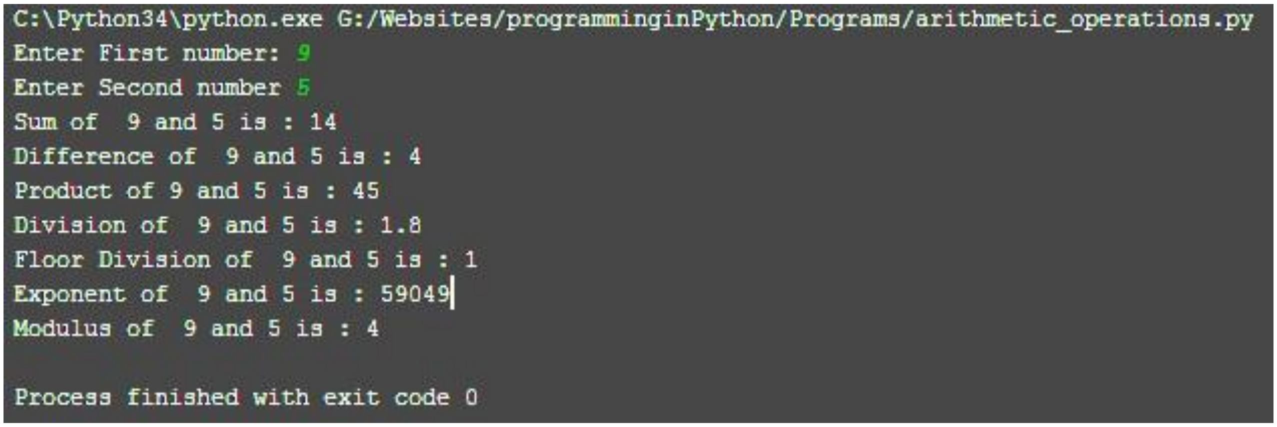
Arithmetic Operators

a = 5
b = 2

Operator	Meaning	Example	Result
+	Addition Operator. Adds two Values.	a + b	7
-	Subtraction Operator. Subtracts one value from another	a – b	3
*	Multiplication Operator. Multiples values on either side of the operator	a * b	10
/	Division Operator. Divides left operand by the right operand.	a / b	2.5
%	Modulus Operator. Gives reminder of division	a % b	1
**	Exponent Operator. Calculates exponential power value. a ** b gives the value of a to the power of b	a ** b	25
//	Integer division, also called floor division. Performs division and gives only integer quotient.	a // b	2

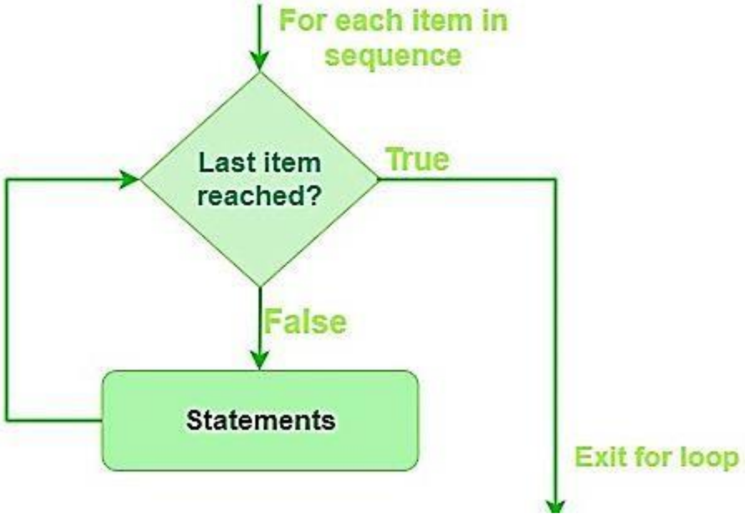
```
num1 = int(input('Enter First number: '))
num2 = int(input('Enter Second number '))
add = num1 + num2
dif = num1 - num2
mul = num1 * num2
div = num1 / num2
floor_div = num1 // num2
power = num1 ** num2
modulus = num1 % num2
print('Sum of ',num1 ,'and' ,num2 ,'is :',add)
print('Difference of ',num1 ,'and' ,num2 ,'is :',dif)
print('Product of' ,num1 ,'and' ,num2 ,'is :',mul)
print('Division of ',num1 ,'and' ,num2 ,'is :',div)
print('Floor Division of ',num1 ,'and' ,num2 ,'is :',floor_div)
print('Exponent of ',num1 ,'and' ,num2 ,'is :',power)
print('Modulus of ',num1 ,'and' ,num2 ,'is :',modulus)
```

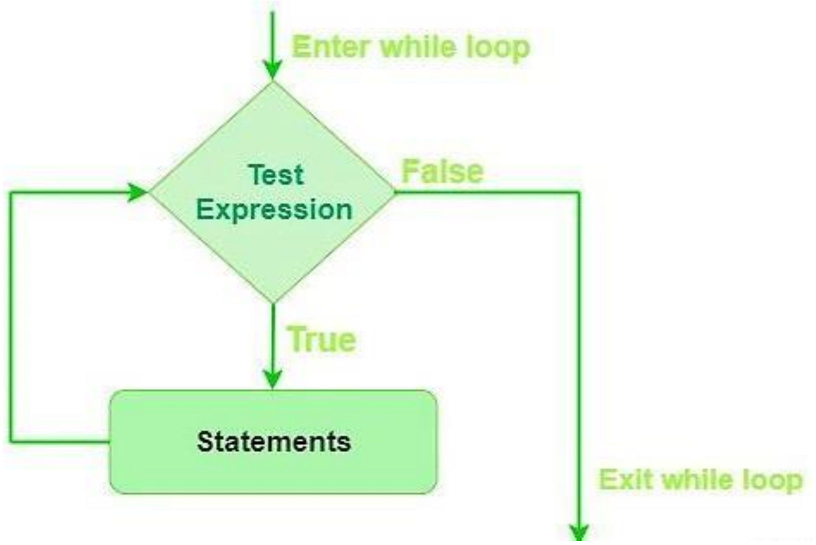
CODE OUTPUT:



```
C:\Python34\python.exe G:/Websites/programminginPython/Programs/arithmetic_operations.py
Enter First number: 9
Enter Second number 5
Sum of  9 and 5 is : 14
Difference of  9 and 5 is : 4
Product of 9 and 5 is : 45
Division of  9 and 5 is : 1.8
Floor Division of  9 and 5 is : 1
Exponent of  9 and 5 is : 59049
Modulus of  9 and 5 is : 4

Process finished with exit code 0
```

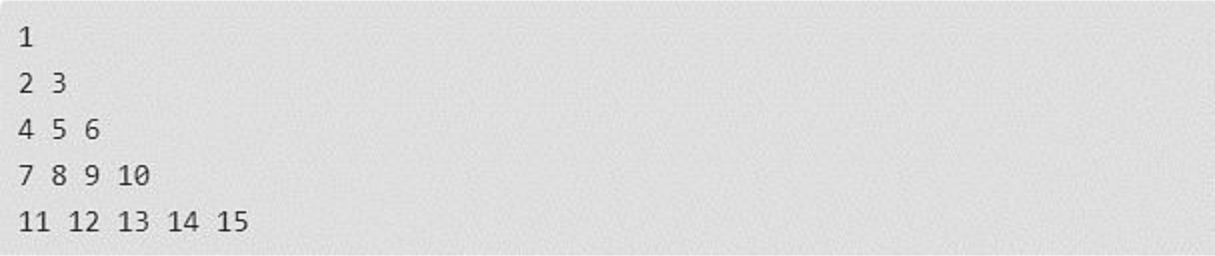
b) *Looping statement in Python*





```
# Python 3.x code to demonstrate star pattern

# Function to demonstrate printing pattern of numbers
def contnum(n):

    # initializing starting number
    num = 1

    # outer loop to handle number of rows
    for i in range(0, n):

        # not re assigning num
        # num = 1

        # inner loop to handle number of columns
        # values changing acc. to outer loop
        for j in range(0, i+1):

            # printing number
            print(num, end=" ")

            # incrementing number at each column
            num = num + 1

        # ending line after each row
        print("\r")

n = 5

# sending 5 as argument
# calling Function
contnum(n)
```

CODE OUTPUT:



```
1
2 3
4 5 6
7 8 9 10
11 12 13 14 15
```

EXPERIMENT NO. 2

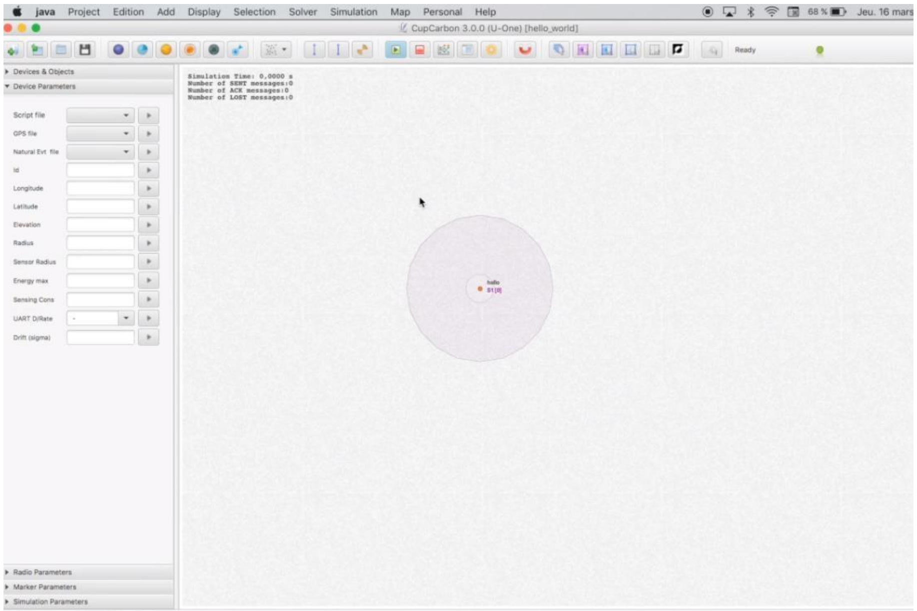
AIM: Run some programs on IOT simulator - Cupcarbon like:

a) Read Hello message

b) Read two numbers and print their sum, difference, product and division.

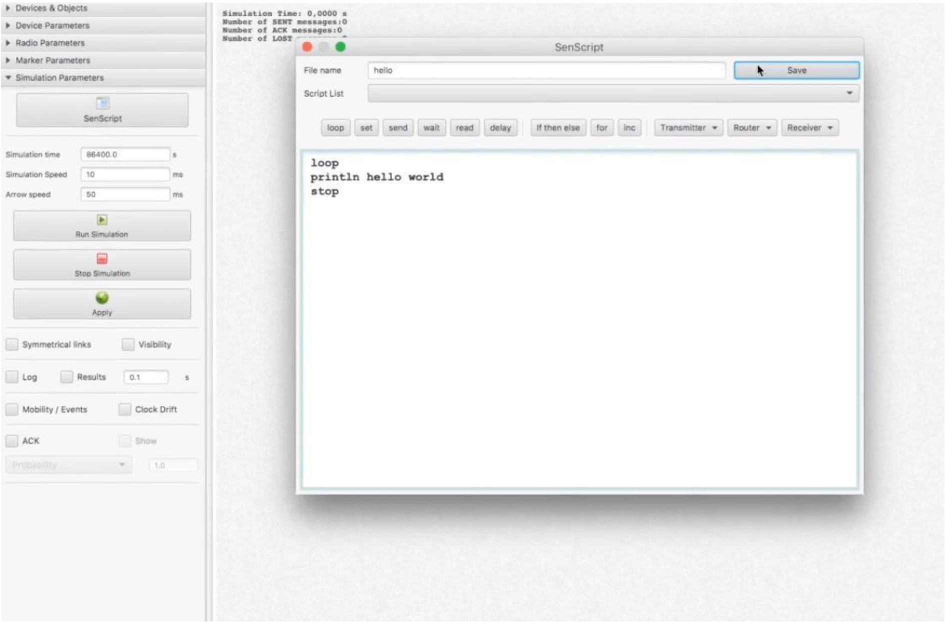
Program:

a) Read Hello message



```
Simulation Time: 0,0000 s
Number of SENT messages:0
Number of ACK messages:0
Number of LOST messages:0
```

(Hello S1 – Sensor)



```
Simulation Time: 0,0000 s
Number of SENT messages:0
Number of ACK messages:0
Number of LOST
```

File name: hello

Simulation time: 86400.0 s

Simulation Speed: 10 ms

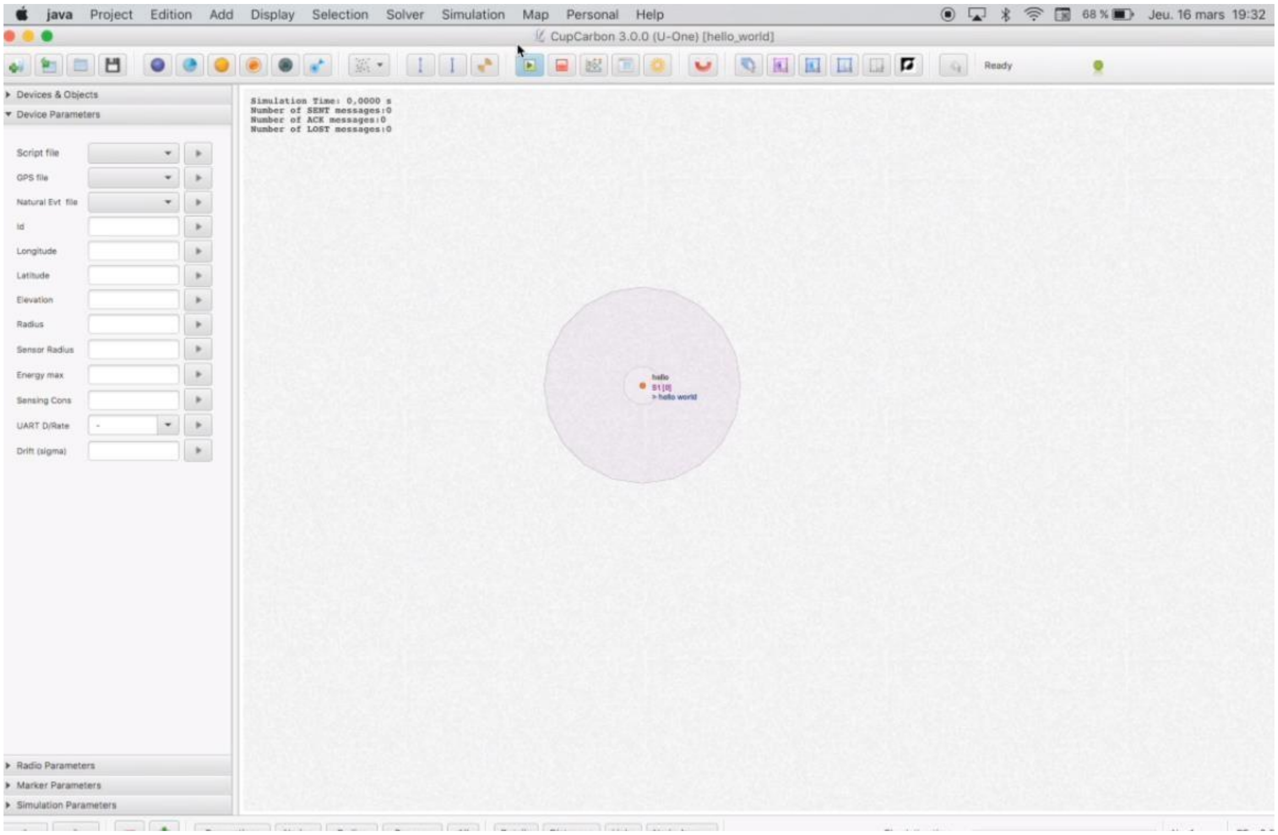
Arrow speed: 50 ms

Results: 0.1 s

```
loop
println hello world
stop
```

(Hello S1 – CODE)

OUTPUT:



```
Simulation Time: 0,0000 s
Number of SENT messages:0
Number of ACK messages:0
Number of LOST messages:0
```

(Sensor displaying the Hello message)

b) *Read two numbers and print their sum, difference, product and division.*

ADDITION

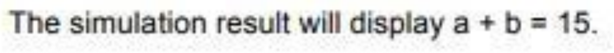
```
CupCarbon v. 3.0.0 (U-One)
----------
name:3_addition
zoom:2
centerposition_la:48.39147687133919
centerposition_lo:-4.489213228225708
map:2
display_details:true
draw_radio_links:true
draw_sensor_arrows:true
radio_links_color:0
draw_marker_arrows:false
display_rl_distance:false
propagation:false
display_marker_distance:false
```

```
display_radio_messages:true
draw_script_file_name:true
```

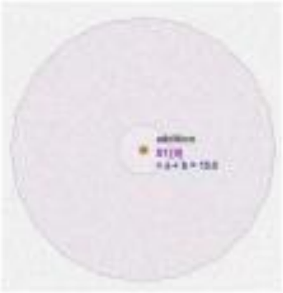
CODE:

```
Loop
set a 7 set b 8
plus x $a $b
print a + b = $x
stop
```

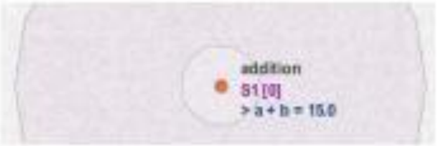
OUTPUT:



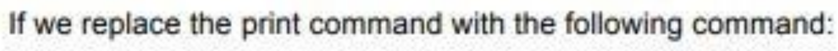
The simulation result will display a + b = 15.



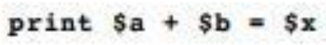
```
addition
S1 [0]
> a + b = 15.0
```



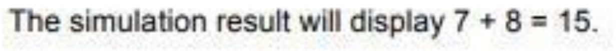
```
addition
S1 [0]
> a + b = 15.0
```



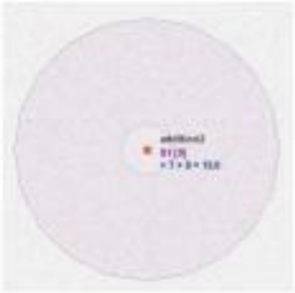
If we replace the print command with the following command:



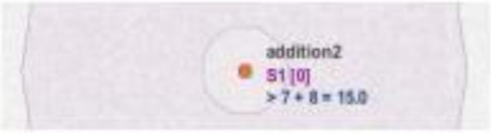
```
print $a + $b = $x
```



The simulation result will display 7 + 8 = 15.



```
addition2
S1 [0]
> 7 + 8 = 15.0
```



```
addition2
S1 [0]
> 7 + 8 = 15.0
```

EXPERIMENT NO. 3

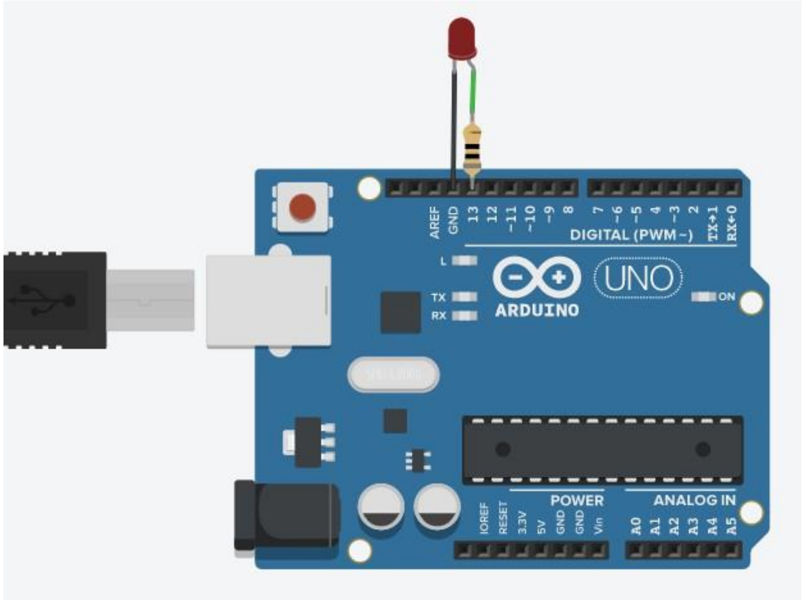
AIM: Study and Install IDE of Arduino and different types of Arduino and

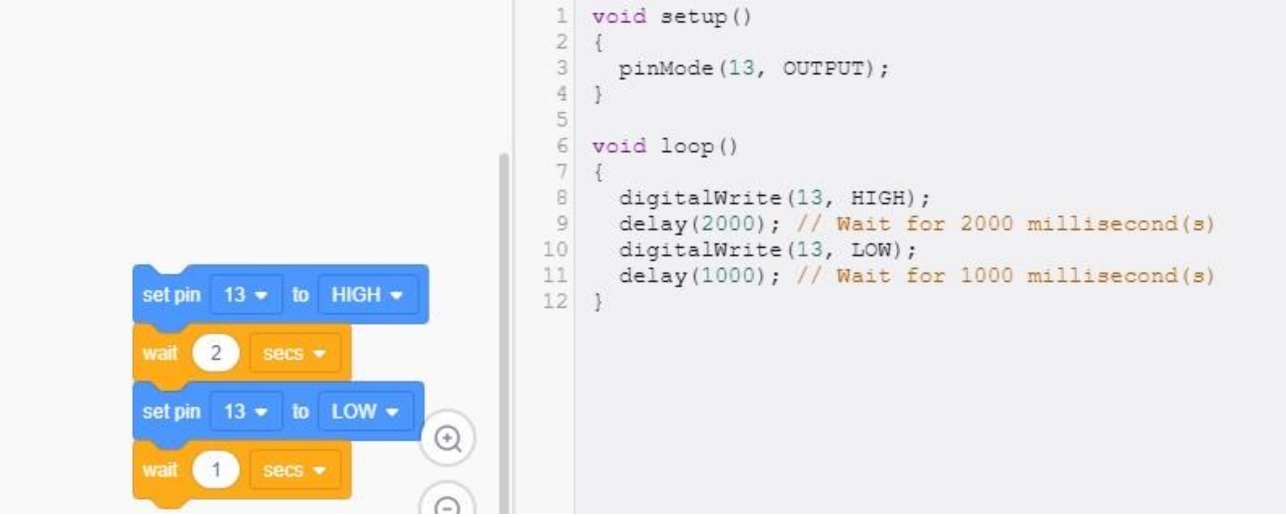
a) Write program using Arduino IDE for Blink LED

b) Get input from two switches and switch on corresponding LEDs

Program:

1. *Write program using Arduino IDE for Blink LED*

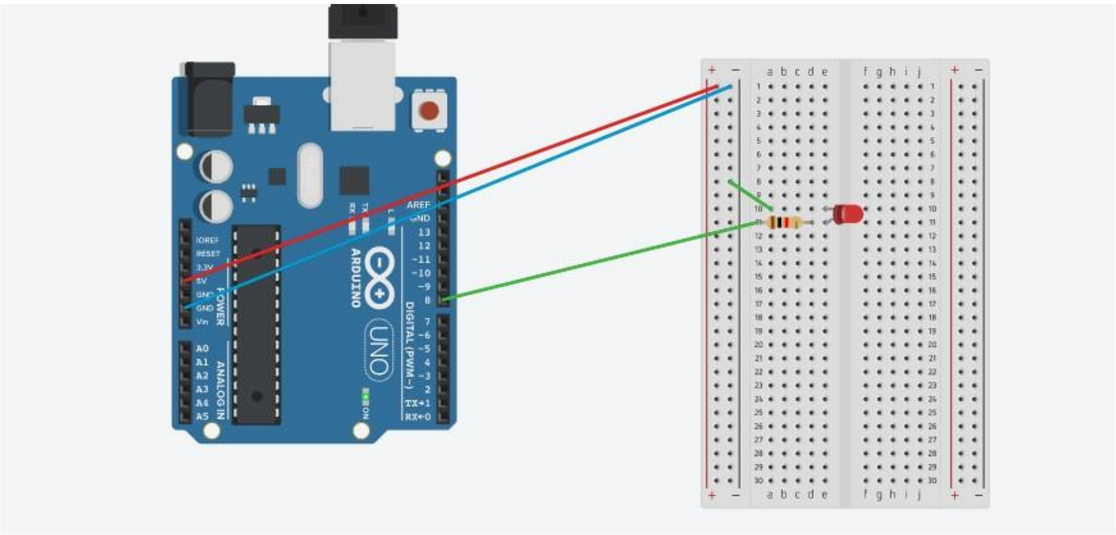




```
void setup()
{
  pinMode(13, OUTPUT);
}

void loop()
{
  digitalWrite(13, HIGH);
  delay(2000); // Wait for 2000 millisecond(s)
  digitalWrite(13, LOW);
  delay(1000); // Wait for 1000 millisecond(s)
}
```

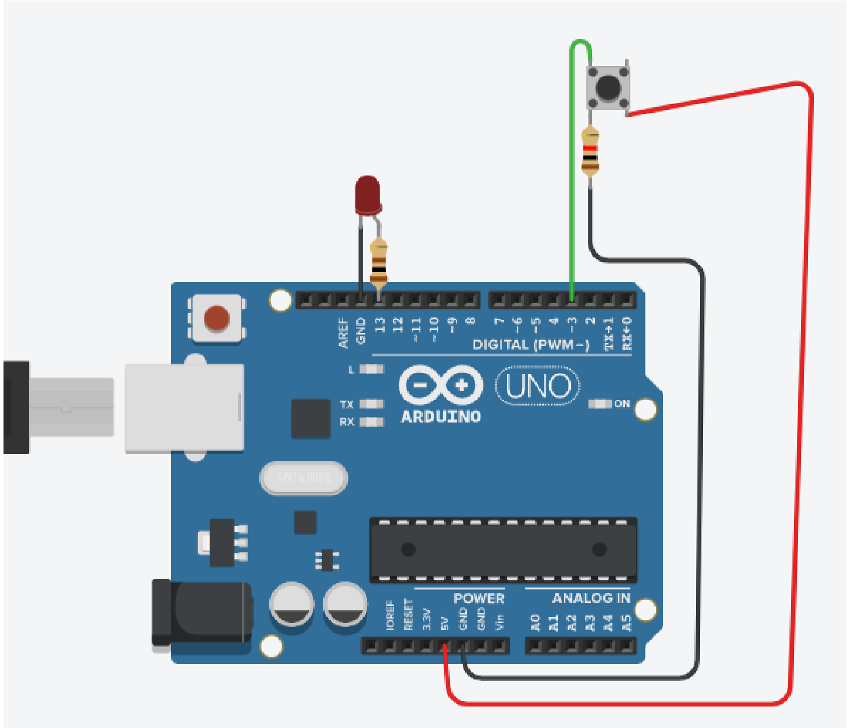
OUTPUT:

Blinking LED

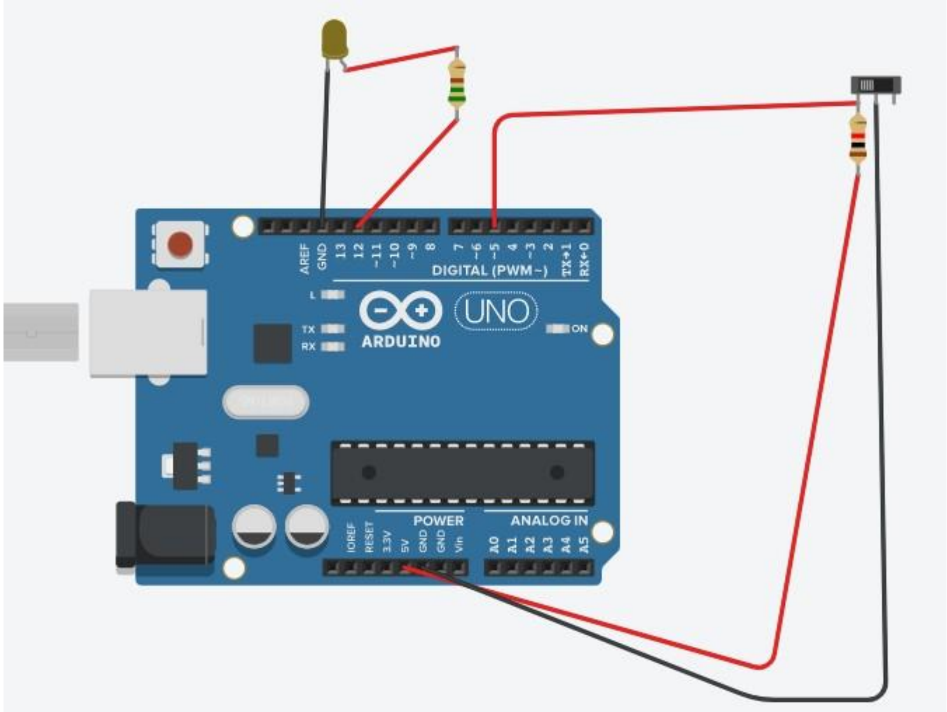


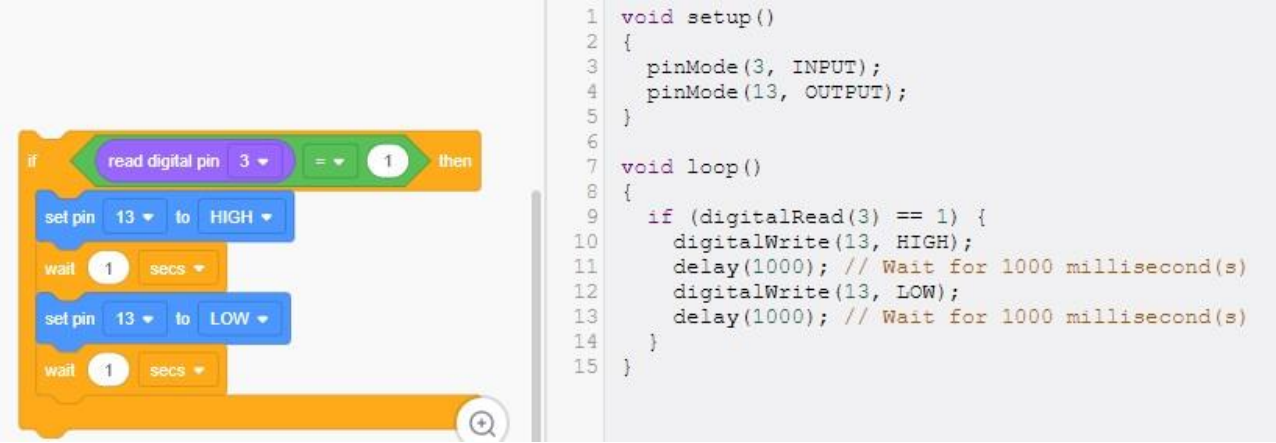
2. *Get input from two switches and switch on corresponding LEDs*

   - *Push Button*



- *Switch button*

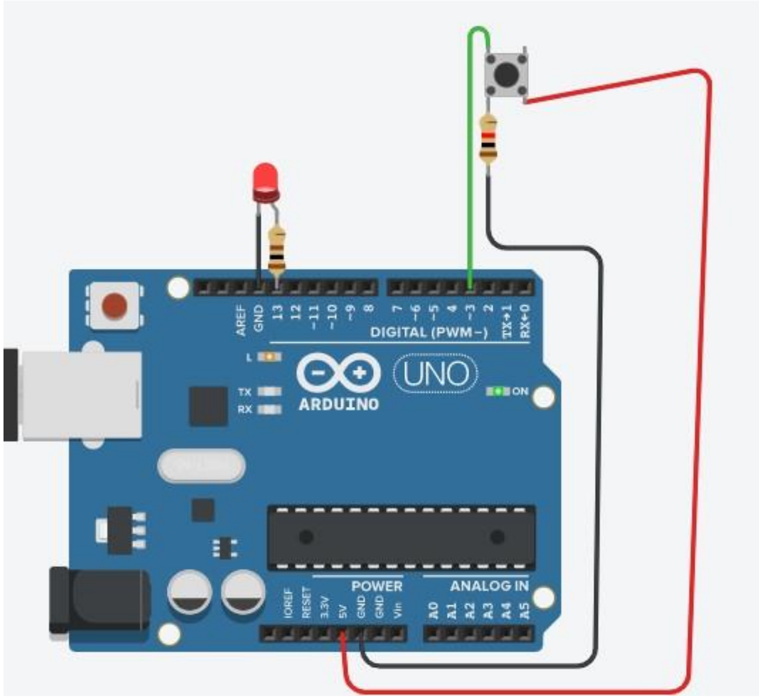




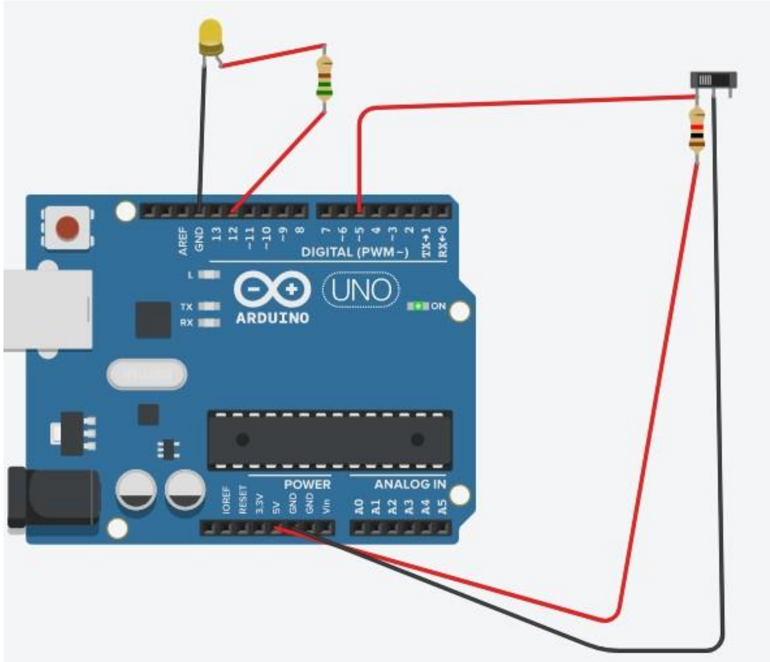
```
void setup()
{
  pinMode(3, INPUT);
  pinMode(13, OUTPUT);
}

void loop()
{
  if (digitalRead(3) == 1) {
    digitalWrite(13, HIGH);
    delay(1000); // Wait for 1000 millisecond(s)
    digitalWrite(13, LOW);
    delay(1000); // Wait for 1000 millisecond(s)
  }
}
```

OUTPUT:

- *Push Button*



- *Switch button*



EXPERIMENT NO. 4

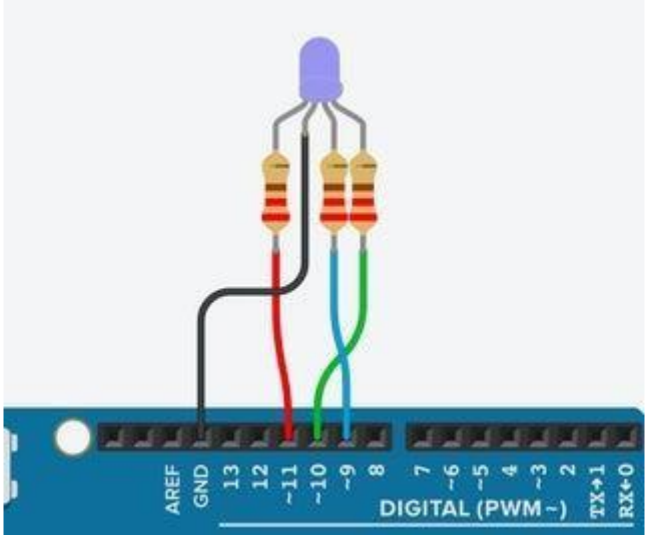
AIM: Write Program for RGB LED using Arduino.

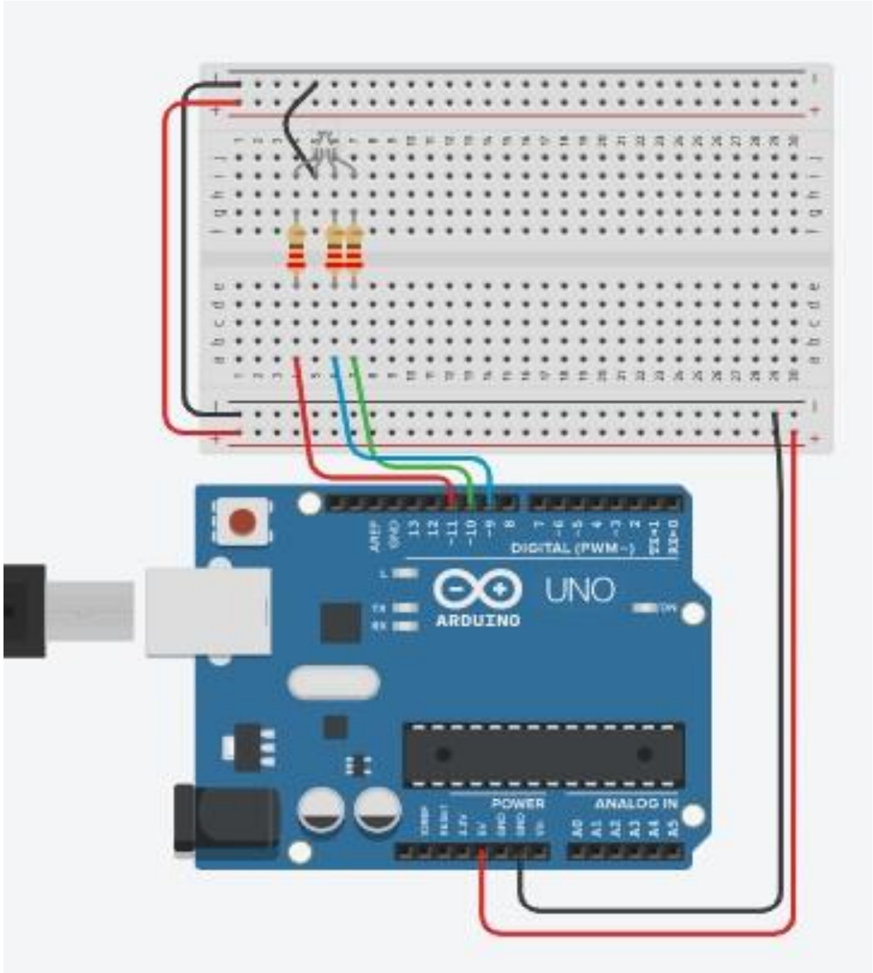
Program:

RGB LED Color Mixing

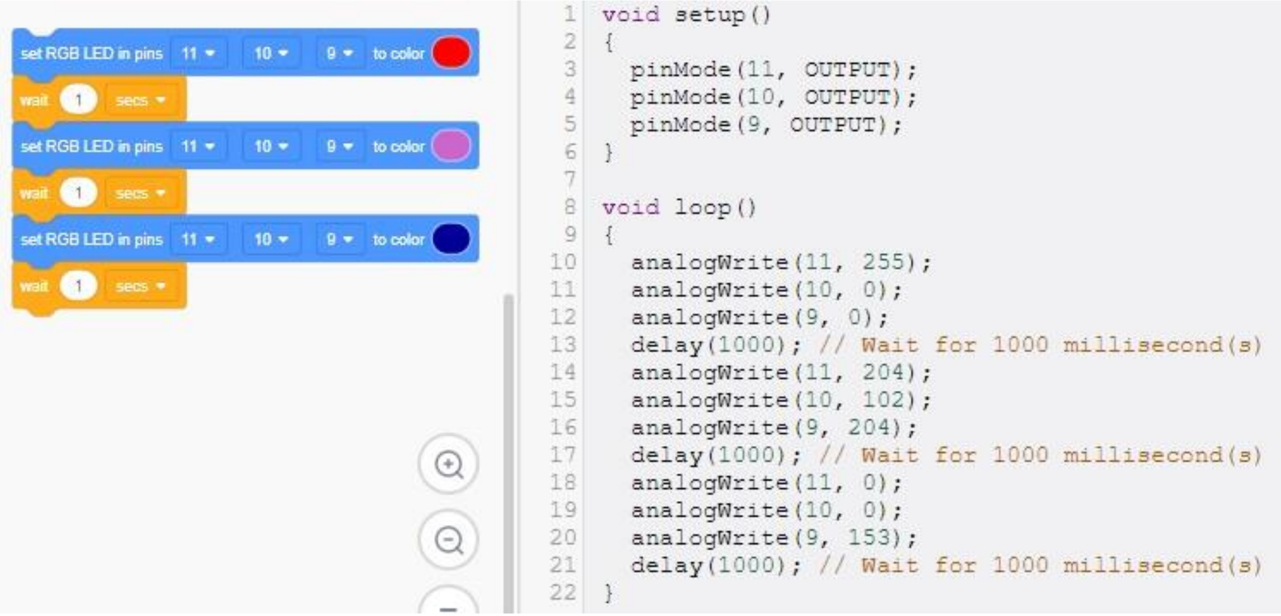
Let's learn how to control multi color LEDs using Arduino's analog outputs. We'll connect an RGB LED to the Arduino Uno and compose a simple program to change its color. Here in Tinkercad Circuits, you can explore the sample circuit (click Start Simulation to watch the LED change color) and build your 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.

Additive, or light-based color has three primary colors: red, green, and blue. Mixing these three colors in different levels of intensity can create almost any color of light. Color changing LEDs work the same way, but the LEDs are all together in a small package we call an RGB LED. They have four legs, one for each color and one for either ground or power, depending on the configuration. The types are called "common cathode" and "common anode," respectively.





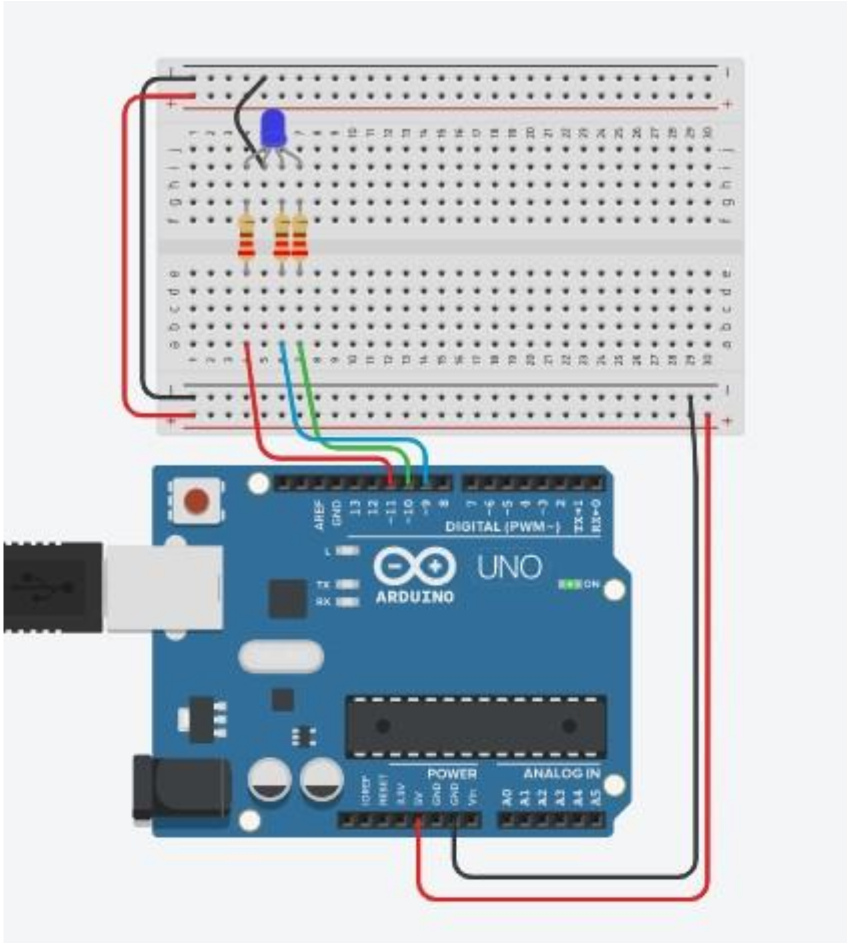
```
void setup()
{
  pinMode(11, OUTPUT);
  pinMode(10, OUTPUT);
  pinMode(9, OUTPUT);
}

void loop()
{
  analogWrite(11, 255);
  analogWrite(10, 0);
  analogWrite(9, 0);
  delay(1000); // Wait for 1000 millisecond(s)
  analogWrite(11, 204);
  analogWrite(10, 102);
  analogWrite(9, 204);
  delay(1000); // Wait for 1000 millisecond(s)
  analogWrite(11, 0);
  analogWrite(10, 0);
  analogWrite(9, 153);
  delay(1000); // Wait for 1000 millisecond(s)
}
```



set RGB LED in pins 11 ▾ 10 ▾ 9 ▾ to color 🔴
wait 1 secs ▾
set RGB LED in pins 11 ▾ 10 ▾ 9 ▾ to color 🟣
wait 1 secs ▾
set RGB LED in pins 11 ▾ 10 ▾ 9 ▾ to color 🔵
wait 1 secs ▾

```
void setup()
{
  pinMode(11, OUTPUT);
  pinMode(10, OUTPUT);
  pinMode(9, OUTPUT);
}

void loop()
{
  analogWrite(11, 255);
  analogWrite(10, 0);
  analogWrite(9, 0);
  delay(1000); // Wait for 1000 millisecond(s)
  analogWrite(11, 204);
  analogWrite(10, 102);
  analogWrite(9, 204);
  delay(1000); // Wait for 1000 millisecond(s)
  analogWrite(11, 0);
  analogWrite(10, 0);
  analogWrite(9, 153);
  delay(1000); // Wait for 1000 millisecond(s)
}
```

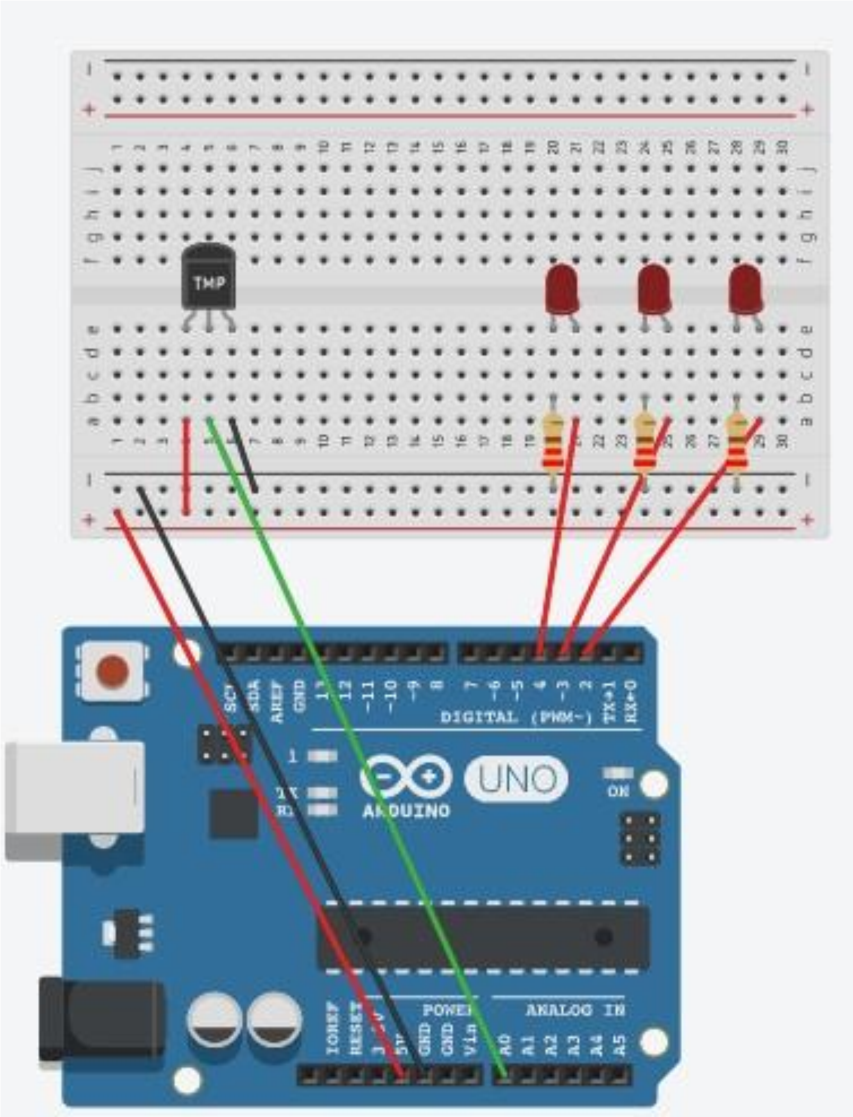
OUTPUT:



EXPERIMENT NO. 5

AIM: Study the Temperature sensor and Write Program foe monitor temperature using Arduino.

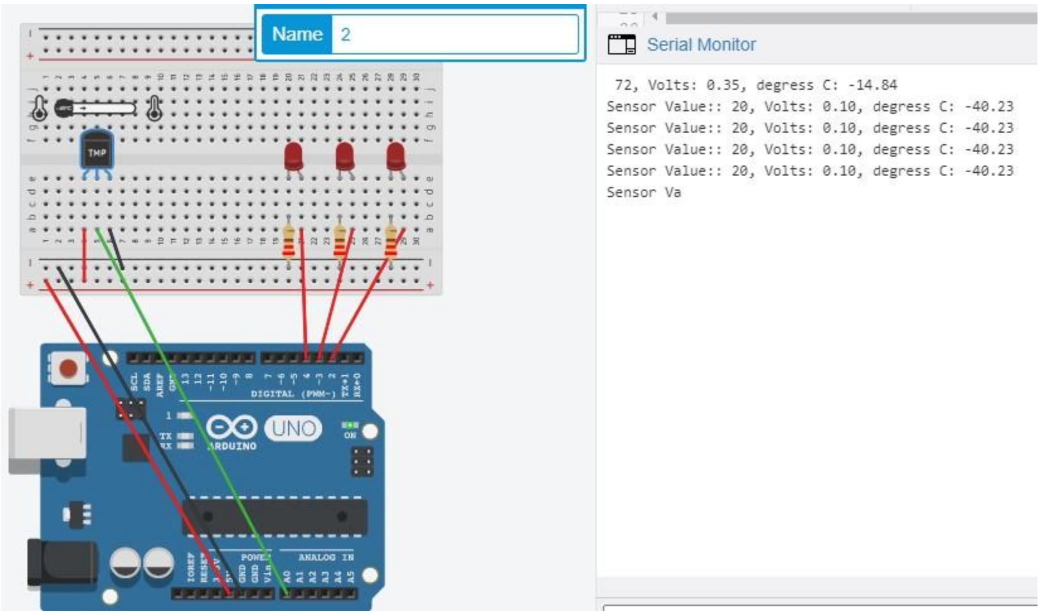
Program:



```
const int sensorPin = A0;
  const float baselineTemp = 20.0;

void setup(){
  Serial.begin(9600); //open a serial port
  for (int pinNumber = 2; pinNumber < 5; pinNumber++){
       pinMode(pinNumber, OUTPUT);
       digitalWrite(pinNumber, LOW);
       }
}

void loop(){
  int sensorVal = analogRead(sensorPin);
  Serial.print("Sensor Value:: ");
  Serial.print (sensorVal);

  //convert the ADC reading to voltage
  float voltage = (sensorVal/1024.0)*5.0;
  Serial.print(", Volts: ");
  Serial.print(voltage);
  Serial.print(", degress C: ");
        //convert the voltage to temperature in degrees
        float temperature = (voltage - .5)*100;
        Serial.println(temperature);
        if(temperature < baselineTemp){
          digitalWrite(2, LOW);
          digitalWrite(3, LOW);
          digitalWrite(4, LOW);
        }
  else if(temperature >= baselineTemp+2 && temperature < baselineTemp+4){
          digitalWrite(2, HIGH);
          digitalWrite(3, LOW);
          digitalWrite(4, LOW);
        }
        else if(temperature >= baselineTemp+4 && temperature < baselineTemp+6){
          digitalWrite(2, HIGH);
          digitalWrite(3, HIGH);
          digitalWrite(4, LOW);
        }
        else if(temperature >= baselineTemp+6){
          digitalWrite(2, HIGH);
          digitalWrite(3, HIGH);
          digitalWrite(4, HIGH);
        }
        delay(1);
 }
```

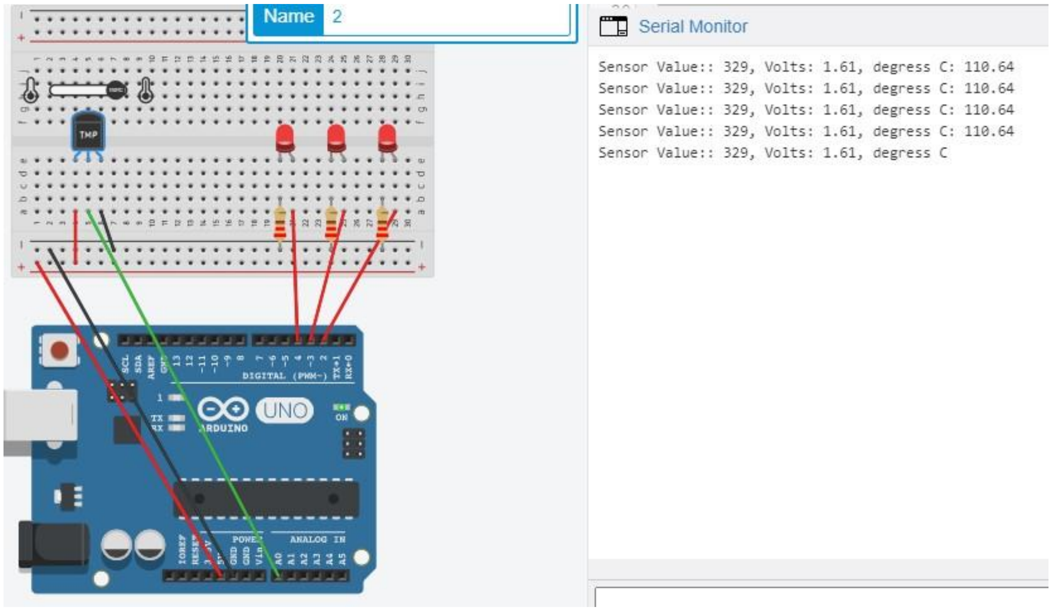
OUTPUT:

- *LED switch off when temperature is low.*



```
 72, Volts: 0.35, degress C: -14.84
Sensor Value:: 20, Volts: 0.10, degress C: -40.23
Sensor Value:: 20, Volts: 0.10, degress C: -40.23
Sensor Value:: 20, Volts: 0.10, degress C: -40.23
Sensor Value:: 20, Volts: 0.10, degress C: -40.23
Sensor Va
```

- *LED switch on when temperature is high.*

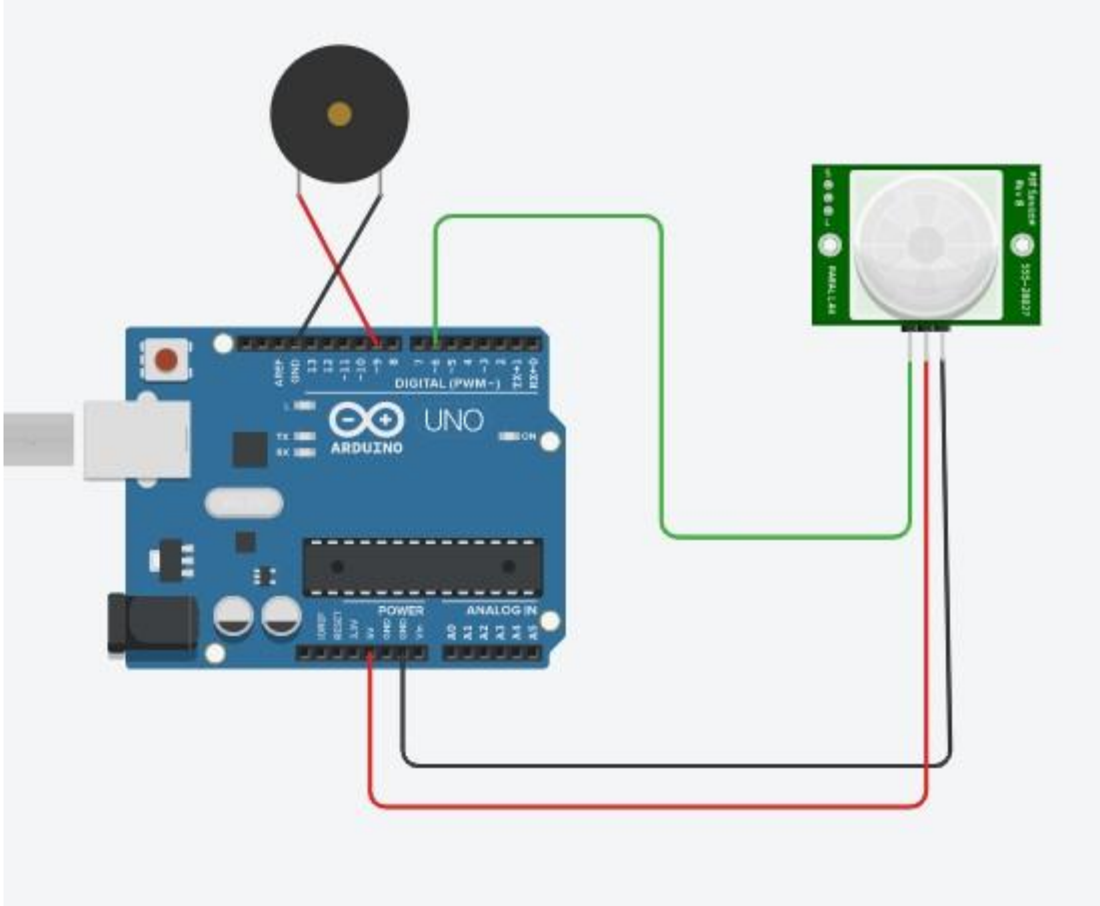


```
Sensor Value:: 329, Volts: 1.61, degress C: 110.64
Sensor Value:: 329, Volts: 1.61, degress C: 110.64
Sensor Value:: 329, Volts: 1.61, degress C: 110.64
Sensor Value:: 329, Volts: 1.61, degress C: 110.64
Sensor Value:: 329, Volts: 1.61, degress C
```

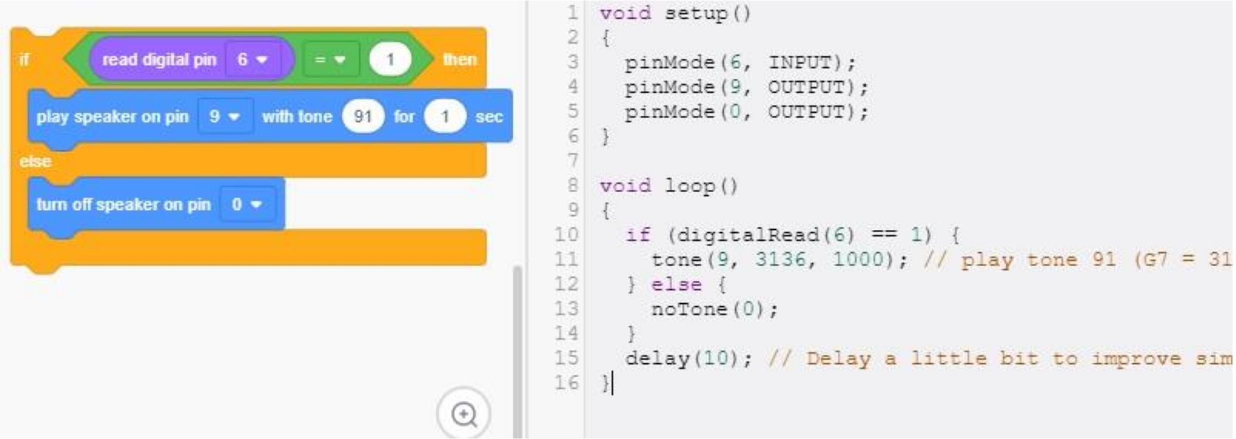
EXPERIMENT NO. 6

AIM: Demonstrate and program for PIR Motion Sensor (Digital Input).

Program:

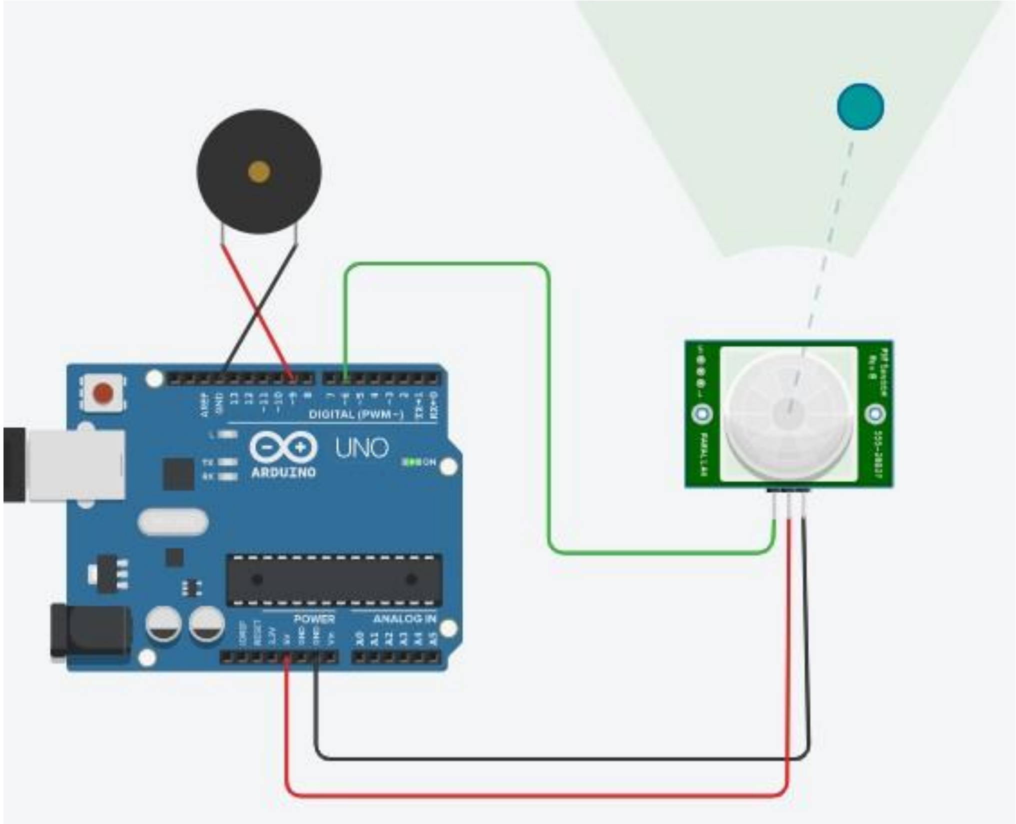


```
void setup()
{
  pinMode(6, INPUT);
  pinMode(9, OUTPUT);
  pinMode(0, OUTPUT);
}

void loop()
{
  if (digitalRead(6) == 1) {
    tone(9, 3136, 1000); // play tone 91 (G7 = 3136 Hz)
  } else {
    noTone(0);
  }
  delay(10); // Delay a little bit to improve simulation performance
}
```



```
void setup()
{
  pinMode(6, INPUT);
  pinMode(9, OUTPUT);
  pinMode(0, OUTPUT);
}

void loop()
{
  if (digitalRead(6) == 1) {
    tone(9, 3136, 1000); // play tone 91 (G7 = 31
  } else {
    noTone(0);
  }
  delay(10); // Delay a little bit to improve sim
}
```

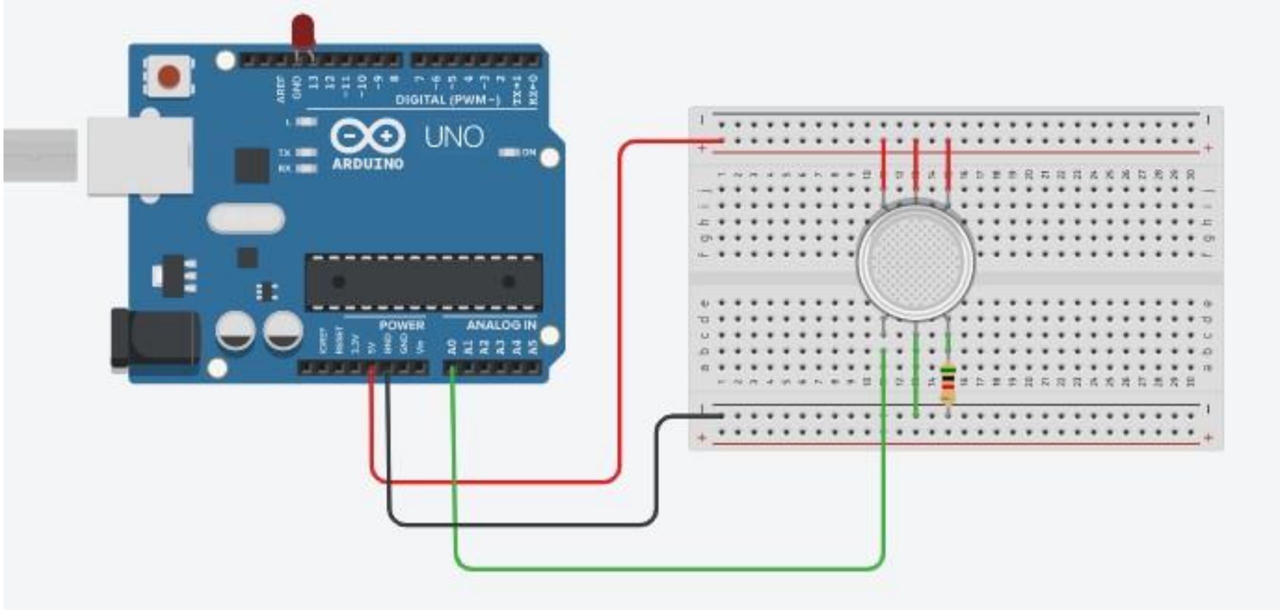
OUTPUT:



PRACTICALS BEYOND RTU SYLLABUS

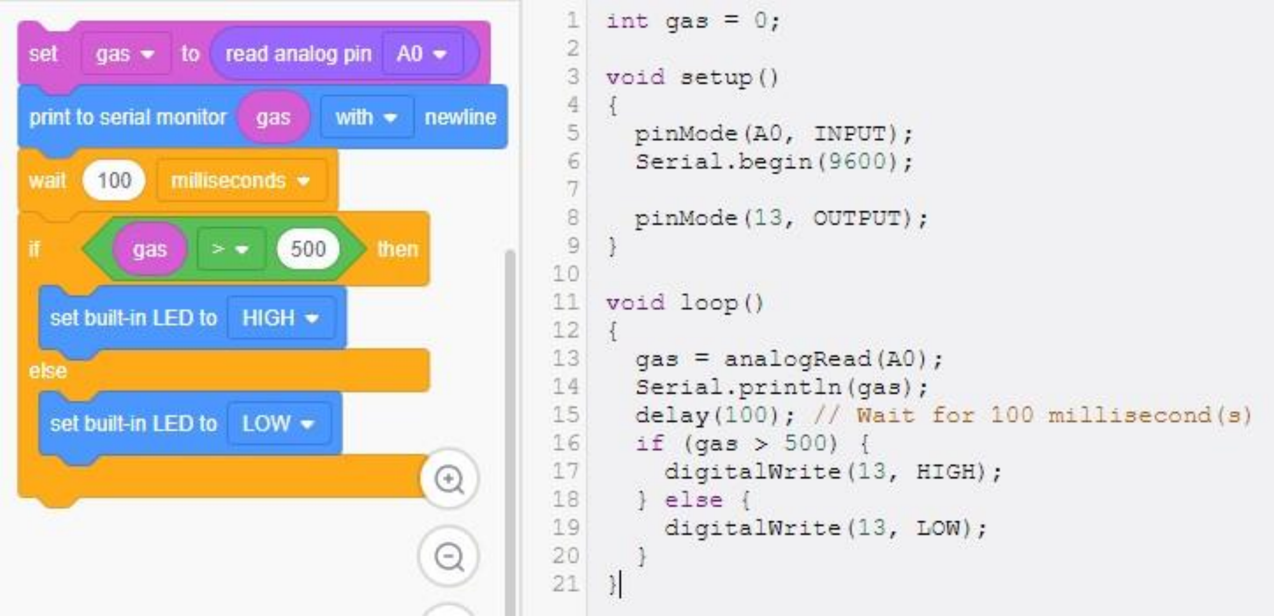
1. **AIM:** Demonstrate and program for Smoke Sensor.

Program:



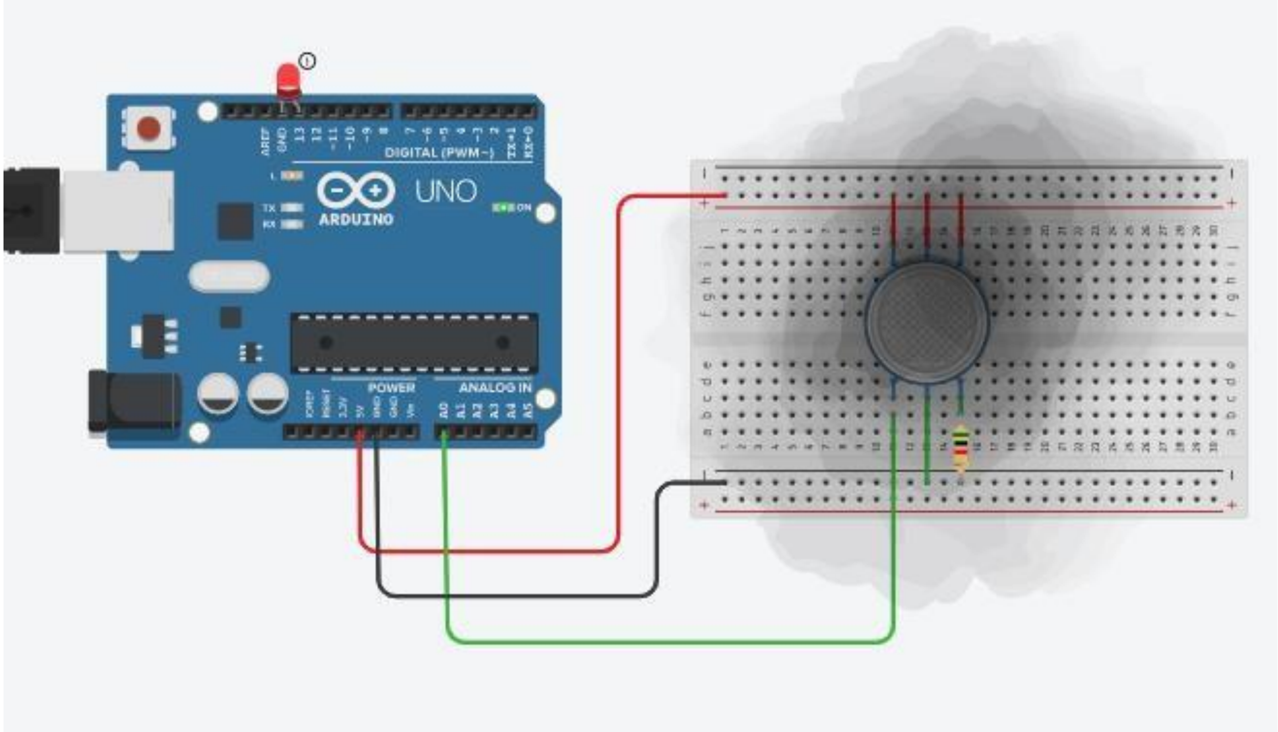
```
int gas = 0;

void setup()
{
  pinMode(A0, INPUT);
  Serial.begin(9600);

  pinMode(13, OUTPUT);
}

void loop()
{
  gas = analogRead(A0);
  Serial.println(gas);
  delay(100); // Wait for 100 millisecond(s)
  if (gas > 500) {
    digitalWrite(13, HIGH);
  } else {
    digitalWrite(13, LOW);
  }
```

}



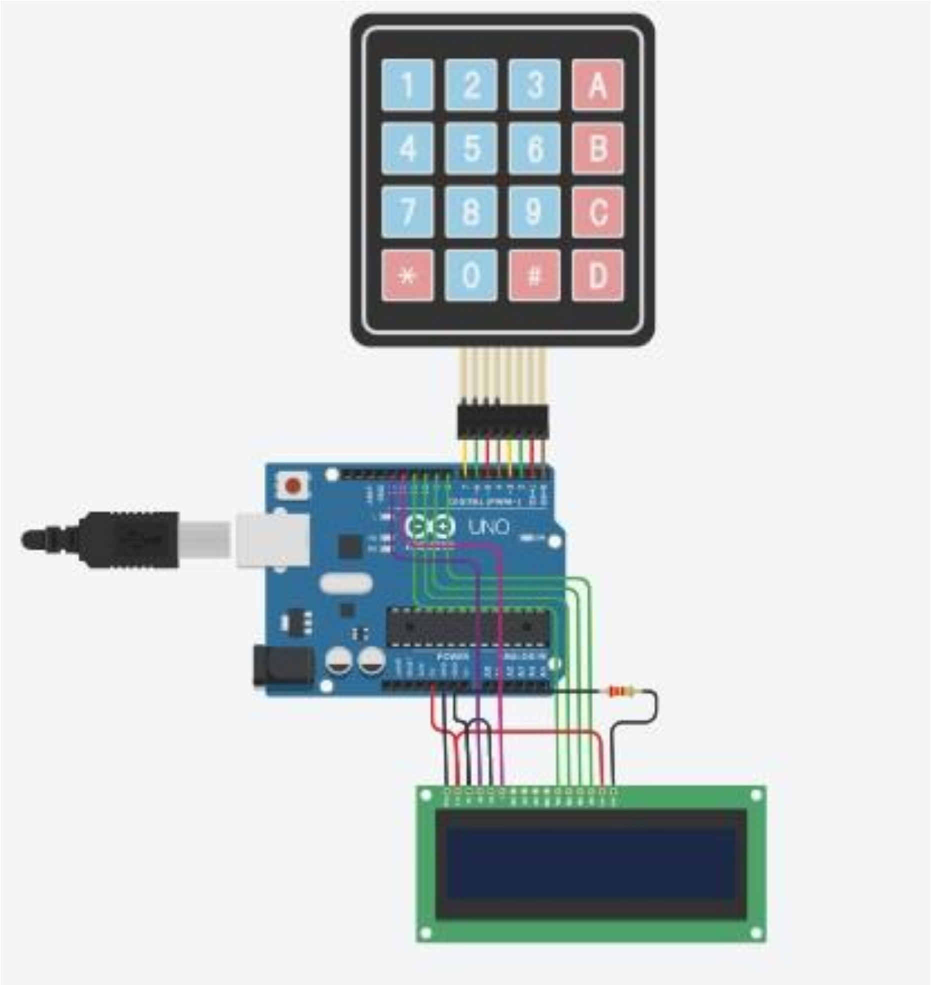
```
int gas = 0;

void setup()
{
  pinMode(A0, INPUT);
  Serial.begin(9600);

  pinMode(13, OUTPUT);
}

void loop()
{
  gas = analogRead(A0);
  Serial.println(gas);
  delay(100); // Wait for 100 millisecond(s)
  if (gas > 500) {
    digitalWrite(13, HIGH);
  } else {
    digitalWrite(13, LOW);
  }
}
```

OUTPUT:



2. AIM: Demonstrate and program for Basic Calculator

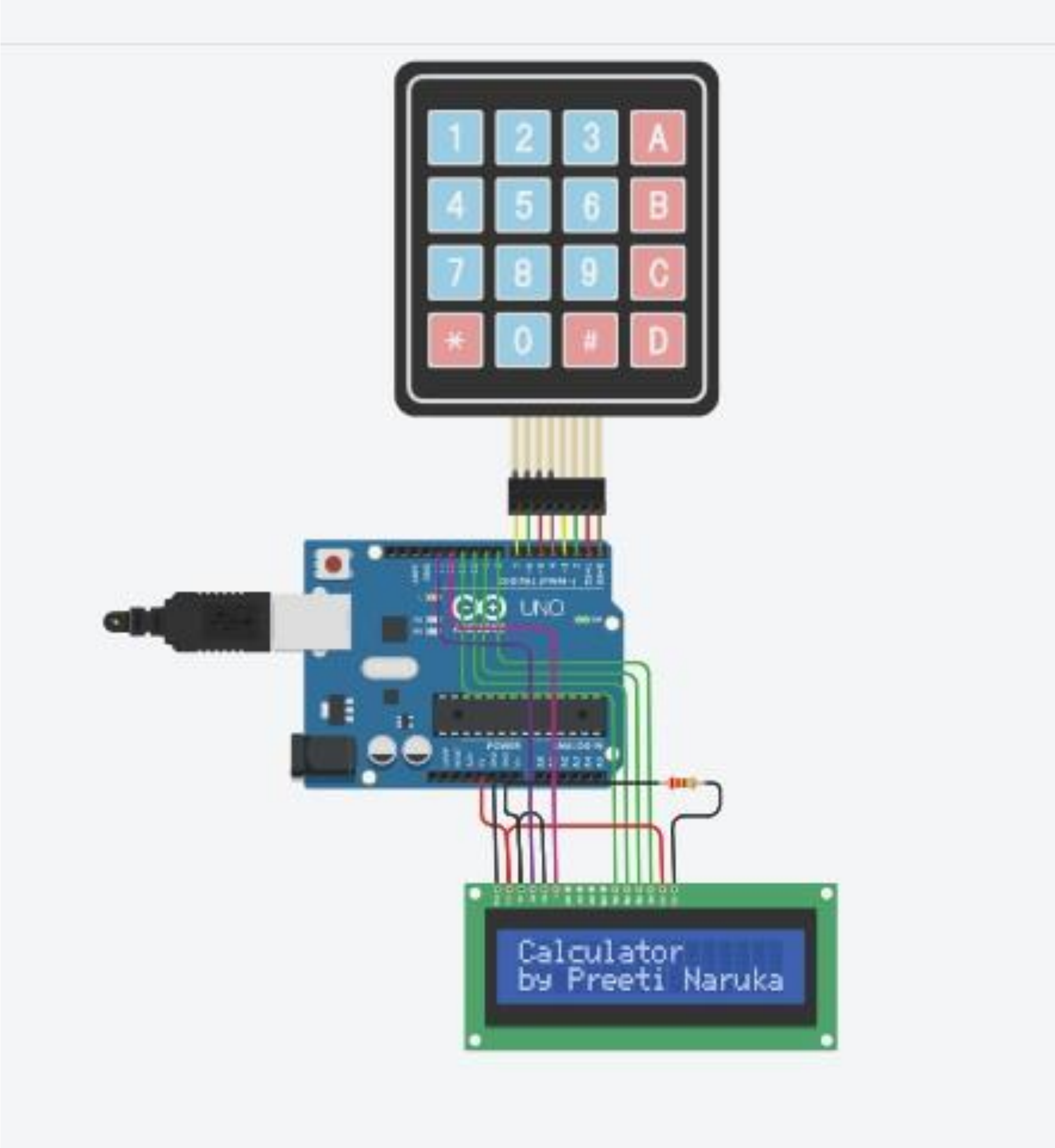
Program:



```
#include <Keypad.h>
#include  <Wire.h>
#include <LiquidCrystal.h>

LiquidCrystal lcd(13, 12, 11, 10, 9, 8);

long first = 0;
long second = 0;
double total = 0;

char  customKey;
const byte ROWS = 4;
const byte COLS = 4;
```

```
char keys[ROWS][COLS] = {
  {'1','2','3','+'},
  {'4','5','6','-'},
  {'7','8','9','*'},
  {'C','0','=','/'}
};
byte rowPins[ROWS] = {7,6,5,4}; //connect to the row pinouts of the keypad
byte colPins[COLS] = {3,2,1,0}; //connect to the column pinouts of the keypad

//initialize an instance of class NewKeypad
Keypad customKeypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS);

void setup()
{
lcd.begin(16, 2);               // start lcd
for(int i=0;i<=3;i++);
lcd.setCursor(0,0);
lcd.print("Calculator");
lcd.setCursor(0,1);
  lcd.print("by Preeti Naruka");
delay(4000);
lcd.clear();
lcd.setCursor(0, 0);
}


void loop()
{

  customKey = customKeypad.getKey();
  switch(customKey)
  {
  case '0' ... '9': // This keeps collecting the first value until a operator is pressed "+-*/"
    lcd.setCursor(0,0);
    first = first * 10 + (customKey - '0');
    lcd.print(first);
    break;

  case '+':
    first = (total != 0 ? total : first);
    lcd.setCursor(0,1);
    lcd.print("+");
    second = SecondNumber(); // get the collected the second number
    total = first + second;
    lcd.setCursor(0,3);
    lcd.print(total);
    first = 0, second = 0; // reset values back to zero for next use
    break;

  case '-':
    first = (total != 0 ? total : first);
    lcd.setCursor(0,1);
    lcd.print("-");
    second = SecondNumber();
    total = first - second;
    lcd.setCursor(0,3);
    lcd.print(total);
    first = 0, second = 0;
    break;

  case '*':
    first = (total != 0 ? total : first);
    lcd.setCursor(0,1);
    lcd.print("*");
    second = SecondNumber();
    total = first * second;
    lcd.setCursor(0,3);
    lcd.print(total);
    first = 0, second = 0;
    break;

  case '/':
    first = (total != 0 ? total : first);
    lcd.setCursor(0,1);
    lcd.print("/");
    second = SecondNumber();
    lcd.setCursor(0,3);

    second == 0 ? lcd.print("Invalid") : total = (float)first / (float)second;

    lcd.print(total);
    first = 0, second = 0;
    break;
  case 'C':
    total = 0;
    lcd.clear();
    break;
  }
}

long SecondNumber()
{
  while( 1 )
  {
    customKey = customKeypad.getKey();
    if(customKey >= '0' && customKey <= '9')
    {
      second = second * 10 + (customKey - '0');
      lcd.setCursor(0,2);
      lcd.print(second);
    }

    if(customKey == '=') break;  //return second;
  }
 return second;
}
```

OUTPUT:



RUBRICS EVALUATION

Performance Criteria	Scale 1 (0-25%)	Scale 2 (26-50%)	Scale 3 (51-75%)	Scale 4 (76-100%)	Score (Numerical)
Understandability Ability to analyse Problem and Identify solution	Unable to understand the problem.	Able to understand the problem partially and unable to identify the solution	Able to understand the problem completely but unable to identify the solution	Able to understand the problem completely and able to provide alternative solution too.	
Logic Ability to specify Conditions & control flow that are appropriate for the problem domain.	Program logic is incorrect	Program logic is on the right track but has several errors	Program logic is mostly correct, but may contain an occasional boundary error or redundant or contradictory condition.	Program logic is correct, with no known boundary errors, and no redundant or contradictory conditions.	
Debugging Ability to execute /debug	Unable to execute program	Unable to debug several errors.	Able to execute program with several warnings.	Able to execute program completely	
Correctness Ability to code formulae and algorithms that reliably produce correct answers or appropriate results.	Program does not produce correct answers or appropriate results for most inputs.	Program approaches correct answers or appropriate results for most inputs, but can contain miscalculations in some cases.	Program produces correct answers or appropriate results for most inputs.	Program produces correct answers or appropriate results for all inputs tested.	
Completeness Ability to demonstrate and deliver on time.	Unable to explain the code.and the code was overdue.	Unable to explain the code and the code submission was late.	Able to explain code and the program was delivered within the due date.	Able to explain code and the program was delivered on time.	
				TOTAL	

OUTCOMES OF LAB

After Completion of all the practical experiment students have achieved:

- Students learned the definition and significance of the programming language which used for Internet of Thing.
- Students have explored the different platform/software/hardware used for IOT
- Students have applied the knowledge of Basic IOT techniques and protocols to develop innovative applications with number of sensors and components.

Computer Lab's Do's and Don't and Safety Rules

DO's

- Please switch off the Mobile/Cell phone before entering Lab.
- Check whether all peripheral are available at your desktop before proceeding for the session
- Arrange all the peripheral and seats before leaving the lab.
- Properly shutdown the system before leaving the lab.
- Keep the bag outside in the racks.
- Enter the lab on time and leave at proper time.
- Maintain the decorum of the lab.

DON'TS

- Don't mishandle the system.
- Don't leave the system on standing for long
- Don't bring any external material in the lab.
- Don't make noise in the lab.
- Don't bring the mobile in the lab.
- Don't enter in the lab without permission of lecturer/laboratory technician immediately
- Don't delete or make any modification in system files.
- Don't bring storage devices like pen drive without permission of lecturer/laboratory technician.

Computer Lab Safety Rules

- Know the location of the fire extinguisher and how to use them in case of an emergency.
- Report fires or accidents to your lecturer/laboratory technician immediately
- Report any broken plugs or exposed electrical wires to your lecturer/laboratory technician immediately.
- Avoid stepping on electrical wires or any other computer cables.
- Do not open the system unit casing or monitor casing particularly when the power is turned on.
- Do not touch, connect or disconnect any plug or cable without your lecturer/laboratory technician's permission.
- Do not bring any food or drinks near the machine.