Index

Introduction

The **Internet of Things (IoT)** refers to a network of interconnected physical devices, embedded with sensors, software, and other technologies, that communicate and exchange data over the internet. These devices range from everyday objects like smartwatches and home appliances to complex industrial machinery and autonomous vehicles. By leveraging internet connectivity, IoT enables real-time data collection, processing, and analysis, offering unprecedented automation and efficiency.

Features of IoT

IoT systems are characterized by features like **interconnectivity**, allowing seamless communication between devices; **automation**, reducing the need for human intervention; and **real-time data monitoring**, enabling timely insights and actions. Other key features include scalability to support millions of devices, remote access capabilities, and intelligent decision-making powered by machine learning and AI.

Characteristics of IoT

- **Connectivity**

IoT systems enable seamless communication between devices through various protocols such as Wi-Fi, Bluetooth, Zigbee, and LoRaWAN. This interconnected network allows devices to exchange data efficiently, ensuring a continuous flow of information.

- **Intelligence and Automation**

IoT leverages AI and machine learning to analyze data and make intelligent decisions. This results in automation, where devices perform tasks with minimal human intervention, such as adjusting a thermostat based on weather conditions or detecting faults in machinery.

- **Heterogeneity**

IoT systems integrate diverse devices, sensors, and actuators from various manufacturers, each with its unique functionalities and standards. This diversity is managed through interoperability standards and middleware solutions.

- **Dynamic Adaptation**

IoT systems are capable of adapting to dynamic environments. Devices can connect, disconnect, and communicate seamlessly without disrupting the overall system, making IoT networks highly flexible and resilient.

- **Scalability**

With billions of devices expected to join IoT networks, scalability is critical. IoT systems are designed to accommodate this exponential growth without compromising performance or efficiency.

- **Data-Driven Insights**

IoT devices continuously generate massive volumes of data. By analyzing this data, organizations can uncover valuable insights, optimize operations, and provide personalized services.

- **Security and Privacy**

Ensuring the security of IoT devices and the privacy of the data they collect is a core characteristic. Encryption, authentication, and secure communication protocols are essential to protecting the system.

- **Energy Efficiency**

Many IoT devices are battery-powered, requiring energy-efficient designs. Technologies like low-power communication protocols and energy harvesting help maintain device longevity.

- **When, Why, and How We Use IoT**

IoT is used when there is a need to automate processes, monitor environments remotely, or enhance user experiences. For example, it is employed in smart homes to control lighting and temperature, in healthcare for remote patient monitoring, and in agriculture for precision farming.

We use IoT because it improves efficiency, reduces costs, and enhances decision-making through real-time insights. It also enables new business models, such as predictive maintenance and personalized services, which were previously unattainable.

IoT is implemented through the integration of sensors, connectivity protocols (like Wi-Fi, Bluetooth, or LoRaWAN), and platforms for data storage and analytics. Its deployment requires collaboration between hardware, software, and cloud technologies to achieve seamless operation.

Scope of IoT

Smart Homes and Buildings

IoT enables intelligent control of appliances, lighting, security, and energy management systems. Examples include smart thermostats, automated lighting, and connected security cameras.

Healthcare and Wearables

In healthcare, IoT facilitates remote patient monitoring, fitness tracking, and real-time health data collection using wearable devices. This enhances patient care and reduces the burden on healthcare facilities.

Industrial IoT (IIoT)

IoT is revolutionizing industries through predictive maintenance, supply chain optimization, and process automation. IIoT applications include monitoring equipment health, tracking assets, and optimizing manufacturing processes.

Agriculture

IoT supports precision farming by providing real-time data on soil health, weather conditions, and crop status. This helps farmers optimize resource usage and improve yields.

Transportation and Logistics
IoT improves fleet management, traffic monitoring, and navigation. Smart logistics solutions use IoT for tracking shipments, managing inventory, and optimizing delivery routes.

Smart Cities
IoT is integral to the development of smart cities, enabling efficient management of resources like water, energy, and waste. Applications include smart parking, traffic control, and public safety systems.

Retail and E-commerce
IoT enhances customer experience through personalized shopping, inventory tracking, and automated checkout systems. Smart shelves and IoT-enabled POS systems are common examples.

Energy Management
IoT plays a key role in energy monitoring and management, such as in smart grids, renewable energy systems, and home energy usage optimization.

Environmental Monitoring
IoT devices monitor air quality, water quality, and weather patterns, helping address environmental challenges and improve disaster management strategies.

Autonomous Vehicles
IoT facilitates communication between vehicles and infrastructure (V2X) to enable safer and more efficient transportation systems.

In summary, IoT is a transformative technology with applications that extend across diverse sectors, driving innovation, efficiency, and new business opportunities. Its potential to shape the future is vast, making it a cornerstone of technological advancement.

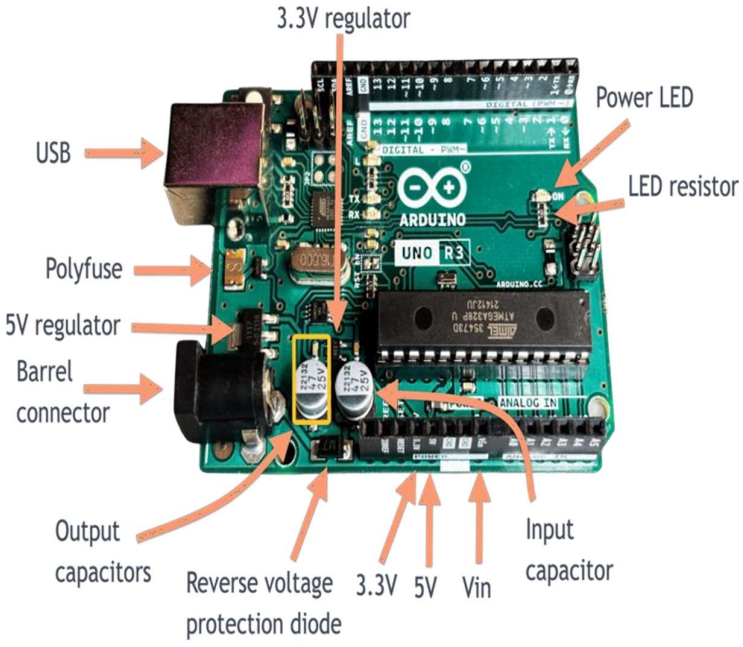
1. Learn the basics of Arduino MCU boards, features and pinouts of Arduino UNO. differentiate between READ and WRITE pins, install and configure the Arduino IDE. und basics of soldering

Arduino is on open source hardware and software company,project and user community that designs and manufactures single-board microcontrollers and their kits for building digital devices

	Specifications
Developer	arduino.cc
Manufactures	Arduino
Type	Single-board microcontroller
Operating System	None(default)/Xinu
CPU	Atmel AVR(8-bit), ARM cortex-MOT(32-bit), ARM cortex-M3(32-bit), Intel Quark (x86) (32-bit)
Memory	SRAM
Storage	Flash, EEPROM

Arduino board designs use a variety of microprocessors and controllers. The boards are equipped with sets of digital input/output (I/O) pins that may be interfaced to various expansion boards or bread boards and other circuits.
The boards feature serial communications interface, including Universal Serial Bus (USB) on some models, which is used for loading programs. The microcontrollers can be programmed using the C and C++ programming languages.

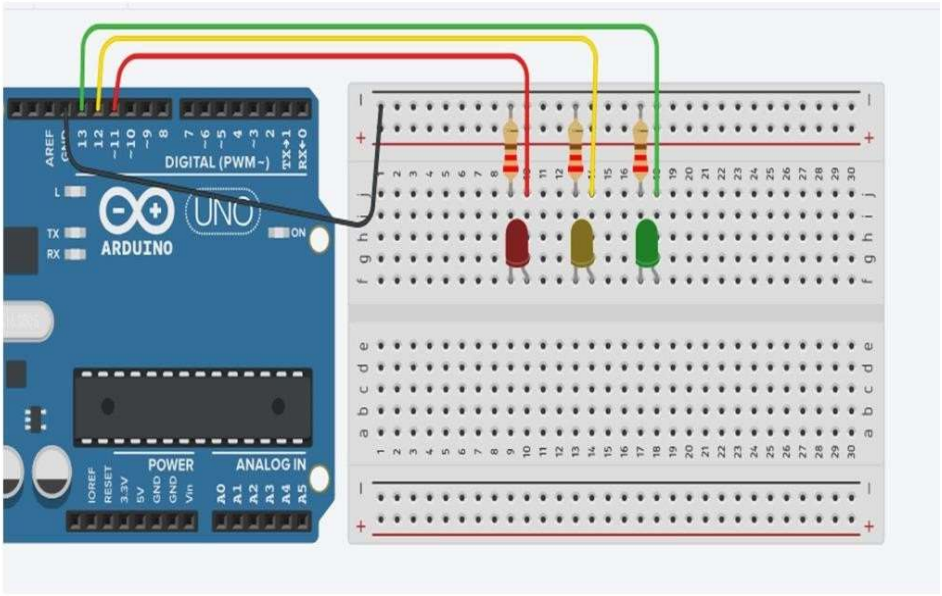
Output



2. Arduino program to blink an LED and implement a traffic signal system using digitalWrite() and pinMode() functions.

```
#define green 13
#define yellow 12
#define red  11
void setup () {
  pinMode(green,output);
  pinMode(yellow,output);
  pinMode(red,output);
}
Void lights(int value1, int value2, int value3){
          digitalWrite(value1,high);
          digitalWrite(value2,low);
          digitalWrite(value3,low);
}
Void loop() {
        lights(green,yellow,red);
        delay(1000);
        lights(yellow,green,red);
        delay(1000);
        lights(red,yellow,green);
        delay(1000);
}
```

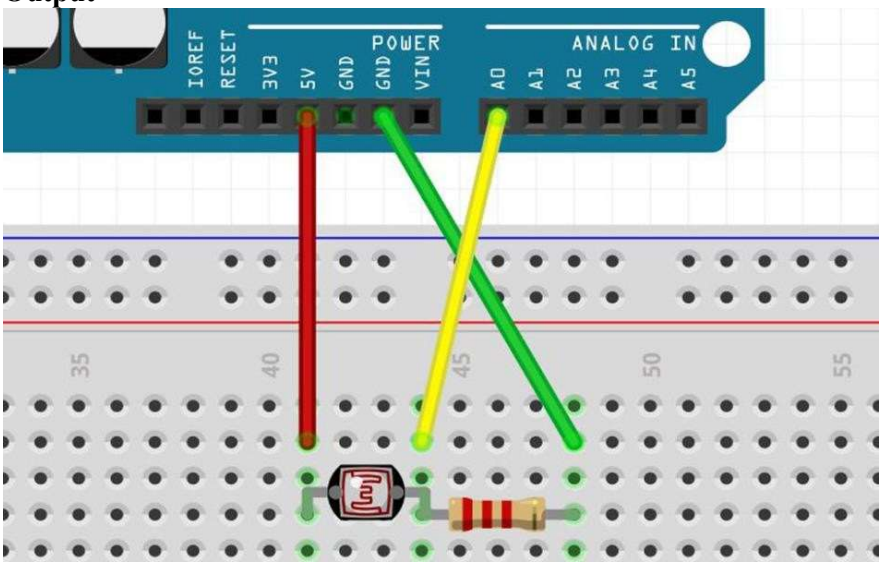
Output



3. Arduino program to vary the intensity of LED based on the reading of Light Dependent Resistor (LDR) using analogRead() and analogWrite() functions.

```
int intensity;
void setup () {
      pinMode(A0,INPUT);
      pinMode(5V,OUTPUT);
}
void loop() {
        intensity =analogRead();
      delay(1000);
      analogwrite(5v,intensity);
}
```

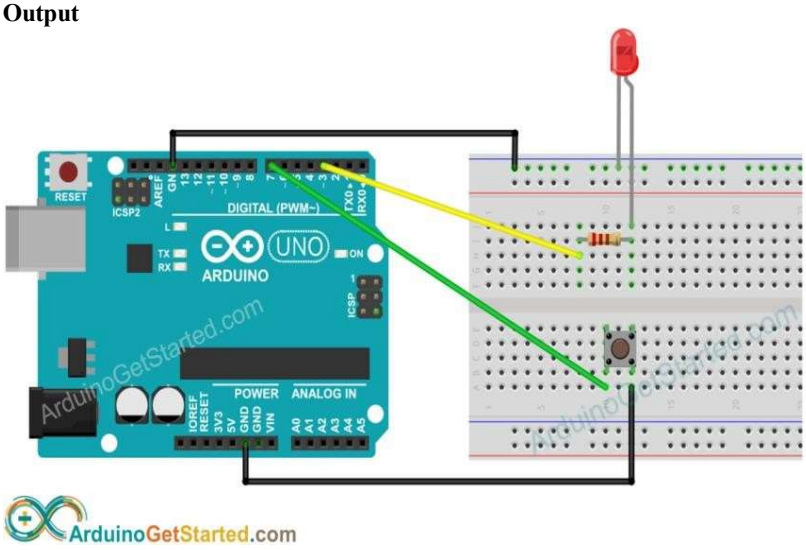
Output



4. Arduino program to toggle LED by pressing a button and to implement a switch debounce circuit to prevent glitches in user input.

```
const int Buttonpin=7;
const int ledpin=3
int ledstate=HIGH;
int buttonState;
int lastButtonState=LOW;
unsigned long lastdebouncetimw=0;
unsigned long debouncedelay=50;
void setup(){
      int reading= digitalRead(Buttonpin);
      if(reading !=lastButtonState)
            lastdebouncetime=millis();
     if((millis()-lastdebouncetime)>debouncedelay){
            if(reading!=ButtonState){
               ButtonState=reading;
               If(ButtonState ==HIGH)
                  ledState=:ledState;
            }
       }
      digitalWrite(ledpin,ledsate);
lastbuttonstate=reading;
```

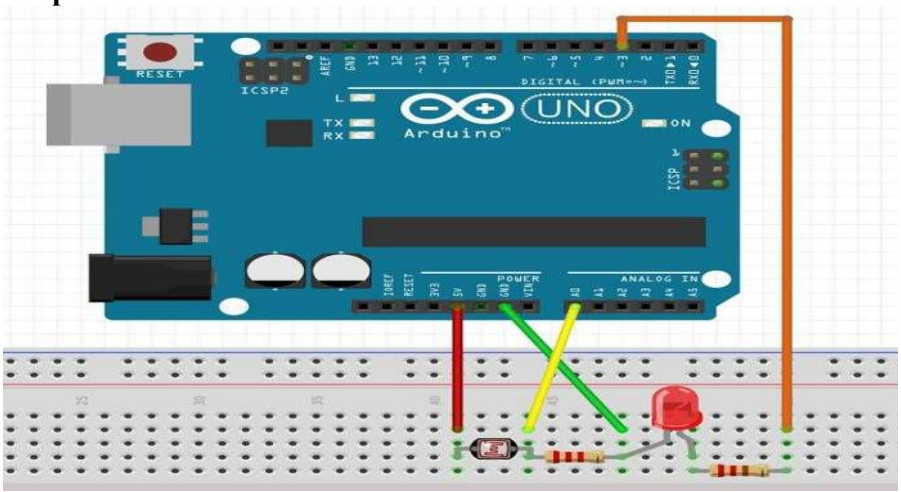
Output



5. Arduino program to implement a serial communication event

```
void setup () {
        pinMode(3,OUTPUT);
       digitalWrite(3,low);
       Serial,begin(9600);
}
void loop(){
        if(Serialavailable()>0){
              char letter= Serial.read();
        if(letter = = '1'){
              digitalWrite(3,HIGH);
              Serial.println("The LED is ON");
        }
        else if(letter = = '0'){
               digitalWrite(3,LOW);
               Serial.println("The LED is off");
        }
         else{
                 Serial.println("Invalid Input");
        }
       delay(200);
}
```

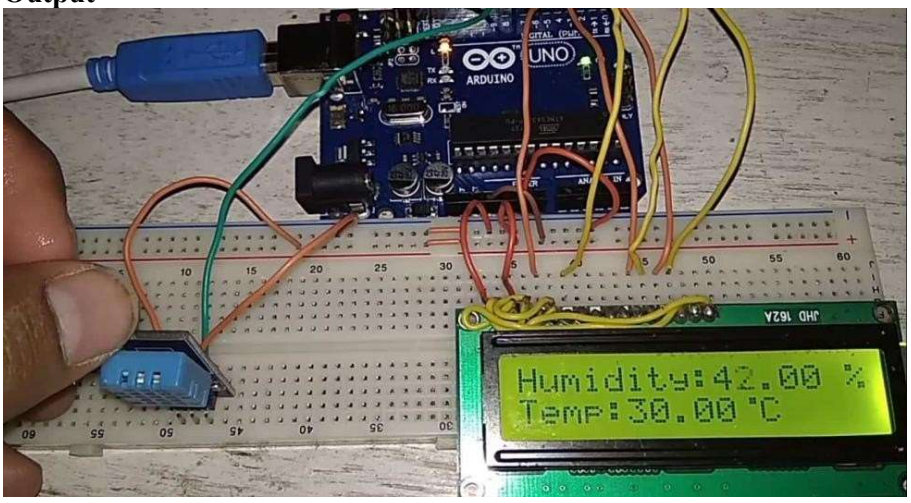
Output



6. Arduino program to implement a temperature and humidity sensor and switch ON an L1.13 the temperature is too hot.

```
#include<SimpleDHT.h>
Int pinDHT11=2;
simpleDHT11 dht(pinDHT11);
Void setup() {
       pinMode(2,OUTPUT);
      Serial.begin(9600);
      Serial.println("Temperature & Humidity")
}
Void loop () {
       Byte temp=0,humid=0;
       Int err=SimpleDHTERRSuccess;
       If((err=dht11.read(&temp,&humid,NULL)!=SimpleDHTERRSucess){
          Serial.println("Read DHT11 failed, err=");
          Serial.println(SimpleDHTERRCode(err));
          Serial.println("DHT11 duration");
          Serial.println("SimpleDHTErrDuration(err));
          return;
}
Serial.print((int)temp);
Serial.println("°c");
Serial.print((int)humid);
Serial.println("H");
If((int)temp>32){
     digitalWrite(13,HIGH);
 }else{
      digitalWrite(12,low);
}
```

Output

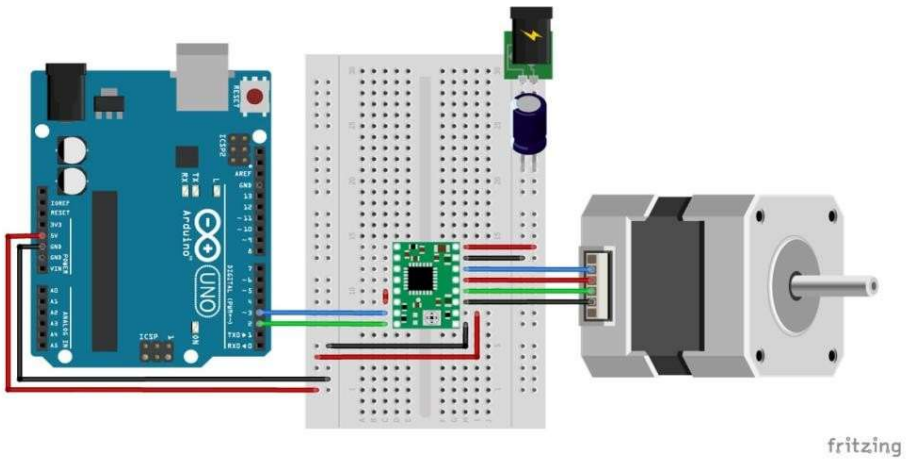


7. Arduino program to drive a DC motor and a stepper motor.

```
Int motor1pin1=2;
Int motor1pin2=3;
Const int stepsPerRevolution=200;
Stepper mystepper(stepsPerRevolution, 8, 9, 10, 11);
Int stepcount=0;

Void setup() {
    pinMode(motor1pin1,OUTPUT);
    pinMode(motor1pin2,OUTPUT);
}
Void loop() {
    digitalWrite(motor1pin1,HIGH);
    digitalWrite(motor1pin2,LOW);
    Delay(1000);
    digitalWrite(motor1pin2,HIGH);
    digitalWrite(motor1pin1,LOW);
    Delay(1000);
    Int sensorvalue=analogRead(A0);
    Int motorSpeed=map(sensorvalue, 0,1023,0,100);
    If(motorSpeed>0){
          Mystepper.setSpeed(motorSpeed);
         Mystepper.step(stepsPerRevolution/100);
}
```

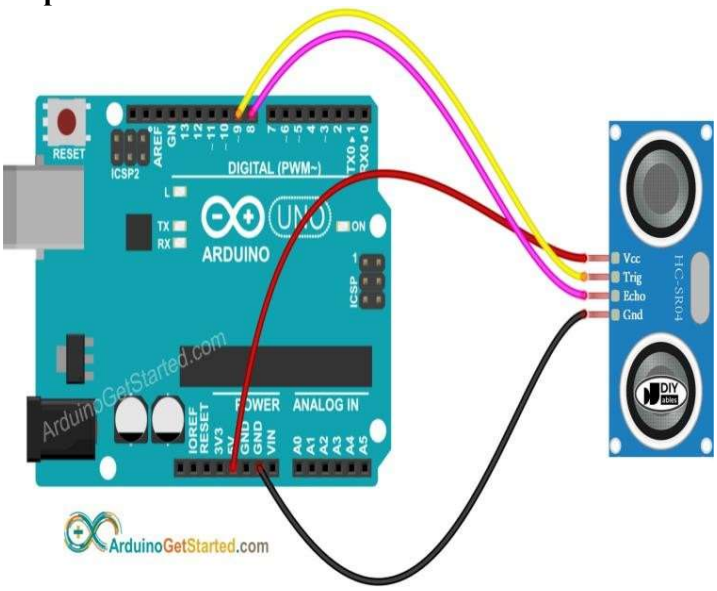
Output



8. Arduino program to implement an ultrasonic sensor to measure distance to an obstacle and "buzz" when too close to object.

```
Int trigpin=9;
Int echopin =8;
Void setup() {
        Serial.begin(9600);
        pinMode(trigpin,OUTPUT);
        pinMode(echopin,OUTPUT);
}
Void loop() {
        Long duration, distance ;
        digitalWrite(trigpin,LOW);
        delayMicroseconds(2);
        digitalWrite(trigpin,HIGH)l
        digitalMicroseconds(10);
        Duration=pulseIn(echopin,HIGH);
        Distance =(duration/2)/29.1;
        Serial.print('distance);
        Delay(100);
}
```

output

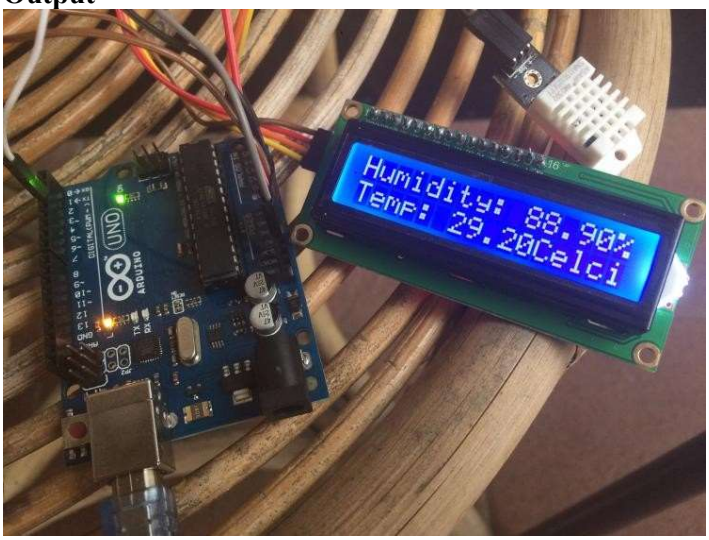


9. Arduino program to implement a 16x2 LCD alphanumeric display and display temperature and current date and time.

```
#include <DS323.h>
#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
DS3231 RTC(SDA,SCL);

void setup () {
RTC.begin();
lcd.begin(16, 2);
RTC.SetDow(SUNDAY);
RTC.SetTime(18,50,35);
RTC.SetDATe(16,12,2018);
}
void loop () {
lcd.setCursor(0, 0);
lcd.print("Real Time Clock");
lcd.setCursor(0, 1);
lcd.print(“Time:”);
lcd.print('rtc.getTimeStr()');
Delay(3000);
lcd.setCursor(0, 1);
lcd.print(0,1);
lcd.print(“Date:”);
lcd.print(rtc.getDateStr());
Delay(3000);
lcd.setCursor(0, 1);
lcd.print(“Day:”)
lcd.print(rtc.getDOWStr());
Delay(3000);
lcd.setCursor(0, 1);
lcd.print(“Temp:”);
lcd.print(rtc.getTemp);
lcd.print(“C”);
lcd.print(“ “);
Delay(3000);
```

Output



10. Arduino program to implement a GSM module and send SMS using some carrier to a cellphone number.

```
#include<SoftwareSerial.h>
#include “Adafruit-FONA.h”
#define FONA_RX 12
#define FONA_TX 13
#define FONA_R!_INTERRUPT 11
Char sendto[21]=”70xxxxxxxxx”;
Char meg[141]=”welcome”;
String t;
SoftwareSerial fonass=softwareSerial(FONA_TX,FONA_RX);
Adafruit_FONA fona=Adafruit_FONA(FONA_RST);

Void setup (){
    Serial.begin(115200);
    Serial.println(F(“FONA incoming call”));
     Delay(5000);
     Fonass.begin(4800);
    If(!fona.begin(fonass)){
    Serial.println(F(“FONA is ok”));
    Fona.print(“AT+(SMP =17,167,0,0\r”);
   Fona.sendSMS(sendto,message);
   Delay(100);
}
```

Output

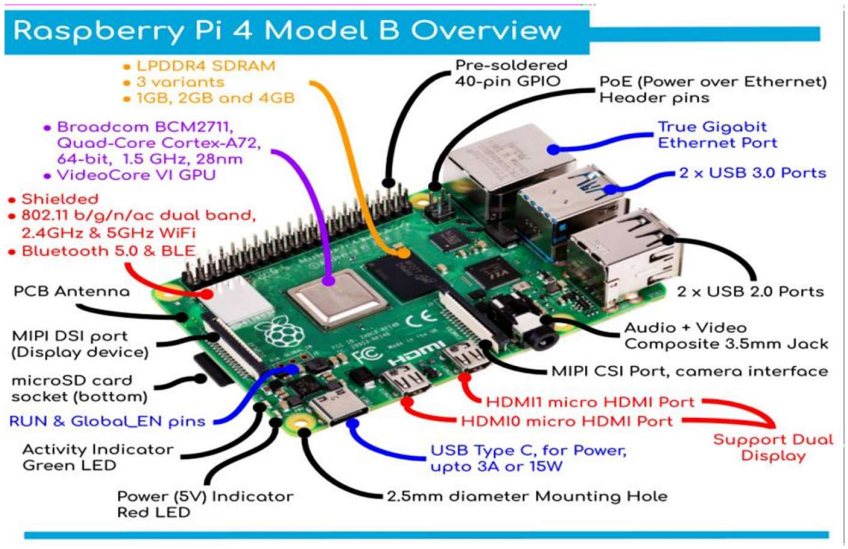
11. **Learn the basics of Raspberry Pi, features, pinout and configuration**.

Raspberry Pi is a series of small Single-Board Computers(SBC)'s developed in the UK by Raspberry Pi Foundation in asssociation with Broadcam.
Specifications also known as Raspi, RPI

Release date	24feb 2012
Operating system	RaspberryPiOS, Free BSD, Linux, Windows 10 IoT, Net BSD, RISCOS
CPU	Pi 3A +:, 1.4 GHZ quad-core A53 64bit Pi 4B: 1.5 GHZ quad-core A72 64bit
Memory	Pi 3A+:512MB, Pi4B :2,4, or 8GB
Storage-	Micro SDHC slot, UniversalSerialBusMassStorage
Graphics	Pi 3A+, Pi4B

The Raspberry Pi is a low cost, credit-card sized computer that plugs into a computer monitor ono TV & uses a standard keyboard and mouse. Raspberry Pi has the ability to interact with the outside world, & has been used in a wide array of digital maker projects. It can be used as replacement for digital PC server.

Output



12. Program to implement MQTT protocol and publish some data.

```
Import paho.mqtt client as mgtt
From random import randrange, uniform
Import time
Mqttb=”mqtt.eclipseprojects.io”
Client=mqtt.client(“Temp-out”)
Client.connect(mqttb)
While True:
    randNumber=randrange(0,10)
    Client.publish(“Temperature”,randNumber)
    Print(“Just published “+str(randNumber)+”to topic Temperature”)
Time.sleep(2)
```

Output

```
Just published  3 to topic Temperature
Just published  5 to topic Temperature
Just published  1 to topic Temperature
Just published  2 to topic Temperature
```