Geethanjali College of Engineering and Technology

(UGC AUTONOMOUS)

Cheeryal (V), Keesara (M), Medchal District – 501 301 (T.S)

Internet of Things

LABORATORY WORK BOOK

B.Tech IV Year I Semester(CSE)

DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING

(2021-2022)

Geethanjali College of Engineering and Technology

UGC AUTONOMOUS

Cheeryal (V), Keesara (M), Medchal District – 501 301 (T S)

Internet of Things

Laboratory Work Book

Name:_____

RollNo:_____

Class:_____ Branch: _____

Academic Year: 20___-20_____

Geethanjali College of Engineering and Technology
UGC AUTONOMOUS

Cheeryal (V), Keesara (M), Medchal District – 501 301 (T S)

CERTIFICATE

*This is to certify thatMr./Miss*_____

*has satisfactorily completed*_____*number of experiments in the Internet of ThingsLaboratory.*

RollNo: _____

Branch:_____

AcademicYear: _____

Year:_____

Course Coordinator

Faculty In charge

InternalExaminer

ExternalExaminer

Geethanjali College of Engineering and Technology

(UGC AUTONOMOUS)

Cheeryal (V), Keesara (M), Medchal District – 501 301 (T S)

18CS41L3-Internet of Things Lab

IV Year I Semester (CSE)

L	T	P/D	C
-	-	2	1

Internet of Things Lab Syllabus

Course Objectives

Develop ability to

1. Assess the vision and introduction of IoT and understanding how M2M is connected to internet of things
2. Identify the appropriate Hardware and software components of IoT for communication
3. Gain knowledge on Cloud Storage models, web servers and how to integrate device,data and cloud management framework for IoT.
4. Learn the concepts of various data analytics and operational technology security with IoT.
5. Understand advanced and emerging concepts fog computing and Edge computing-IoT

Course Outcomes

After completion of the course, student would be able to

CO1: Interpret the vision of IoT from a global context, compare and contrast M2M andIoT Technology

CO2: Relate the appropriate Hardware and software components of IoT for providing the communication among the devices

CO3: Implement device, data and cloud management services for IoT applications.

CO4: Explore various data analytical techniques and operational security for IoT applications.

CO5: Comprehend the need of Fog Computing and Edge Computing-IoT

.

List of Experiments:

1. Getting Started with IoT (Arduino).
2. Write an Arduino sketch to blink an LED Light for a particular interval of time.
3. Write an Arduino sketch to measure the distance(in cms) of a certain object.
4. Write an Arduino sketch to

i. Blink an LED and a buzzer if the distance measured is less than a threshold value

ii. Illustrate the working of PIR Sensor with an example.

iii. Illustrate the IR and DHT Sensor

5. Write an Program to send the humidity and temperature data to Cloud (ThingSpeak)
6. Write a program to alert the user through SMS and Email notification if humidity is greater than a threshold value using IFTTT and Thingspeak cloud.
7. Write a Python program that blinks an LED at a rate of 3 second ON, 1 second OFF
8. Connect a PIR sensor to the GPIO pins of the Raspberry Pi. Perform measurements to determine the range of the sensor, i.e., start with a small distance (e.g., a few inches) and see if the motion sensor responds. Repeat these for increasing distances until the sensor stops responding. Report the measured distance.

9. Select at least 1 input sensor (not PIR) and 1 output device and make the RPi control the chosen output device in response to activity by the input device (e.g., a temperature sensor as input and two or more LEDs indicating the current temperature in binary code).
10. Write a python program for client-server based intruder detection system using mqtt application layer protocol
11. Write an Arduino sketch to blink an LED Light for a particular interval of time using wireless communication protocol (LoRa)

Case study:

1. Assume that you are in a college, design and implement a IoT prototype to measure the amount of usage of water at a given location (take the location from user) on a day to day basis and send the information to Cloud.

2. Receive the above information from the sensors/ cloud and apply necessary algorithms to predict the amount of water being wasted at a particular location and also send a notification to the user

VISION OF THE DEPARTMENT – CSE

To produce globally competent and socially responsible computer science engineers contributing to the advancement of engineering and technology which involves creativity and innovation by providing excellent learning environment with world class facilities.

MISSION OF THE DEPARTMENT – CSE

1. To be a centre of excellence in instruction, innovation in research and scholarship, and service to the stake holders, the profession, and thepublic.
2. To prepare graduates to enter a rapidly changing field as a competent computer science engineer.
3. To prepare graduate capable in all phases of software development, possess a firm understanding of hardware technologies, have the strong mathematical background necessary for scientific computing, and be sufficiently well versed in general theory to allow growth within the discipline as itadvances.
4. To prepare graduates to assume leadership roles by possessing good communication skills, the ability to work effectively as team members, and an appreciation for their social and ethical responsibility in a globalsetting.

PEO's, PO's & PSO's

PROGRAM EDUCATIONAL OBJECTIVES (PEO's) - CSE

1. To provide graduates with a good foundation in mathematics, sciences and engineering fundamentals required to solve engineering problems that will facilitate them to find employment in industry and / or to pursue postgraduate studies with an appreciation for lifelong learning.
2. To provide graduates with analytical and problem-solving skills to design algorithms, other hardware / software systems, and inculcate professional ethics, inter-personal skills to work in a multi-cultural team.
3. To facilitate graduates to get familiarized with the art software / hardware tools, imbibing creativity and innovation that would enable them to develop cutting-edge technologies of multi-disciplinary nature for societal development.

PROGRAM OUTCOMES (Common to all branches)

Engineering Graduates would be able to:

PO 1:Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO 2: Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO 3: Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO 4: Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO 5: Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO 6: The engineer and society: Apply reasoning informed by the contextual knowledge to assess

societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO 7: Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO 8: Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO 9: Individual and teamwork: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO 10: Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO 11: Project management and 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 a team, to manage projects and in multidisciplinary environments.

PO 12: Life-long learning: Recognize the need for and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change..

PROGRAM SPECIFIC OUTCOMES (PSO's) - (CSE)

PSO 1: Demonstrate competency in Programming and problem-solving skills and apply these skills in solving real world problems

PSO 2: Select appropriate programming languages, Data structures and algorithms in combination with modern technologies and tools, apply them in developing creative and innovative solutions

PSO 3: Demonstrate adequate knowledge in emerging technologies

Mapping of Lab Course with Programme Educational Objectives

S.No	Course component	Code	Course	Year/Semester	PEO 1	PEO 2	PEO 3
1	Computer Science	**18CS41L3**	**INTERNET OF THINGS LAB**	IV Year I Sem	√	√	

Mapping of Course Outcomes with Program Outcomes & Program Specific Outcomes- CSE:

1:LOW 2:MEDIUM 3:HIGH

Pos	1	2	3	4	5	6	7	8	9	10	11	12	1	2	3
INTERNET OF THINGS LAB															
CO1. Interpret the vision of IoT from a global context, compare and contrast M2M and IoT Technology	**2**	**2**	**3**	**3**	**3**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**3**	**2**	**2**
CO2. Relate the appropriate Hardware and software components of IoT for providingthecommunication among the devices	**3**	**3**	**3**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**3**	**2**	**2**
CO3. Gain knowledge on Cloud Storage models, web servers and how to integrate device,data andcloud management framework for IoT.	**3**	**3**	**3**	**2**	**3**	**3**	**2**	**2**	**3**	**2**	**2**	**3**	**2**	**2**	-
CO4. Explore various data analytical techniques and operational security for IoT applications.	**2**	**3**	**3**	**3**	**3**	**2**	**2**	**2**	**3**	**2**	**2**	**3**	**2**	**2**	-
CO5.Understand advanced and emerging concepts fog computing and Edge computing-IoT	**3**	**3**	**3**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**3**	**2**	**2**	**2**	-

Prerequisites:

- **18CS11L1- Programming for Problem Solving Lab**
  18CS31L2-Computer Networks Lab

INSTRUCTIONS TO THE STUDENTS:

1. Students are required to attend alllabs.
2. Students should be dressed in formals when attending the laboratorysessions.
3. Students will work individually in computerlaboratories.
4. While coming to the lab bring the observation book and Work booketc.
5. Beforecomingtothelab,preparethepre-labquestions.Readthrough thelab experiment to familiarizeyou.
6. Utilize3hourstimeproperlytoperformtheexperimentandnotingdownthe outputs.
7. If the experiment is not completed in the prescribed time, the pending workhas to be done in the leisure hour or extendedhours.
8. Youwillbeexpectedtosubmitthecompletedworkbookaccordingtothe deadlines set up by yourinstructor.

INSTRUCTIONS TO LABORATORY TEACHERS:

1. Observationbookandlabrecordssubmittedforthelabworkaretobechecked and signed before the next labsession.
2. StudentsshouldbeinstructedtoswitchONthepowersupplyafterthe connections are checked by the lab assistant / teacher.
3. Thepromptnessofsubmissionshouldbestrictlyinsistedbyawarding the marks accordingly.
4. Ask viva questions at the end of theexperiment.
5. Do not allow students who come late to the labclass.
6. Encourage the students to do the experimentsinnovatively.
7. Fill continuous Evaluation sheet, on regularbasis.
8. Ensure that the students are dressed informals.

Scheme of Lab Exam Evaluation:

Evaluation of Internal Marks:

a) 15 Marks are awarded for day to daywork

1) Record andObservationbook ---------- 5Marks
2) Attendance and behaviorofstudent ---------- 5 Marks
3) Vivaandperformance ---------- 5 Marks

b) 15 Marks are awarded for conducting laboratory test asfollows:

1) Write upandprogram -------- 5Marks
2) ExecutionofProgram ---------- 5Marks
3) Vivaandperformance ---------- 5Marks

Evaluation of External Marks:

70 Marks are awarded for conducting laboratory test asfollows:

1) Algorithm ---------- 25 Marks.
2) Write upandprogram ---------- 15Marks
3) ExecutionofProgram ---------- 15Marks
4) Viva----------15 Marks

PERFORMANCE INDICATOR

S No.	Name of the Experiment	Page No.	Date of Experiment	Date of Submission	Marks awarded	Faculty Sign.	Remarks
1							
2							
3							
4							
5							
6							
7							
8							
9							
10							
11							
13							
14							
15							
16							
17							
18							
19							

20							
21							
22							
23							
24							
25							
26							
27							
28							
29							
30							
ADDITIONAL EXPERIMENTS							
1							
2							
3							
4							
5							
6							

WEEK1

Arduino Installation **Date:**

Aim: Getting Started with IoT (Arduino) and perform necessary software installation

Arduino:

- Arduino is a platform that makes it easy for you to build projects using electronics.
- IoT is a way of using electronics - to make electronic modules talk to each other remotely and wirelessly (often using a Cloud) to solve problems.
- Now, Arduino can also help you easily build IoT projects in two ways: Using traditional Arduino boards and attaching communication breakout modules (like nRF, Bluetooth,WiFi,LoRA,GSM, etc) to them.

- Arduino is a micro controller that can be connected to one or more sensors and help you capture the data or information and then pass it on to processor. If you know the full stack of IoT then you should also look at Raspberry.

- RaspPi is a microprocessor so the basic difference between Arduino and RasPi is that RaspPi is controller plus processor and Arduino is just a micro controller.

- They suit the need for different use cases. You can easily read online about this both.

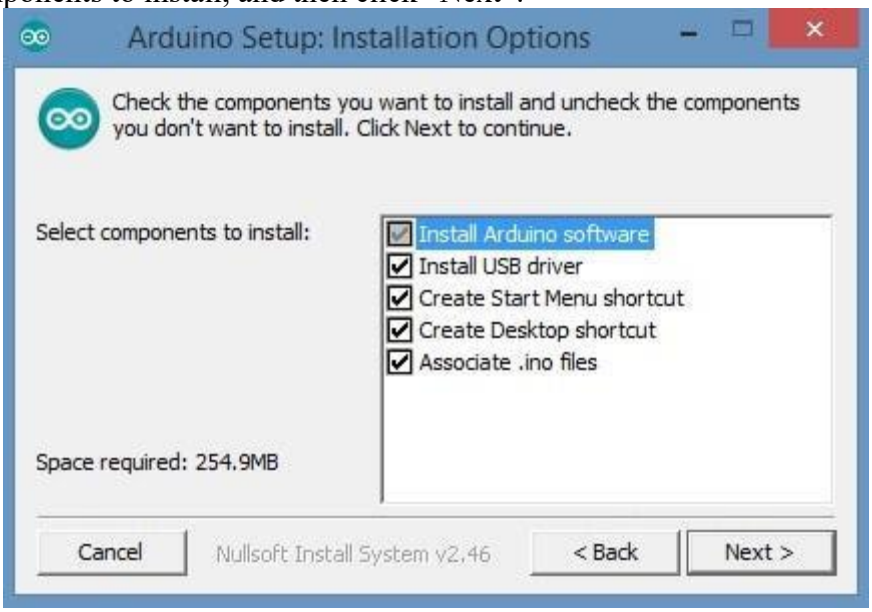
Download and install the Arduino software (Arduino IDE 1.8.15)

- Go to the Arduino website and click the download link to go to the download page.
- After downloading, locate the downloaded file on the computer and extract the folder from the download zipped file. Copy the folder to a suitable place such as your desktop.

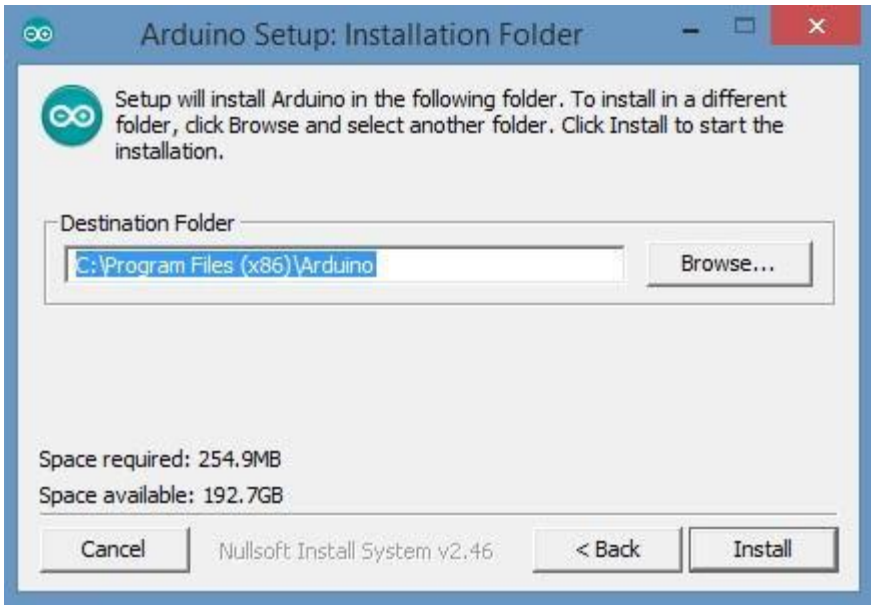
INSTALLING THE ARDUINO IDE ONWindows PCs

1. Visit http://www.arduino.cc/en/main/software to download the latest Arduino IDE version for your computer's operating system. There are versions for Windows, Mac, and Linux systems. At the download page, click on the "Windows Installer" option for the easiest installation.
2. Save the .exe file to your hard drive.
3. Open the .exe file.
4. Click the button to agree to the licensing agreement:

5. Decide which components to install, and then click "Next":

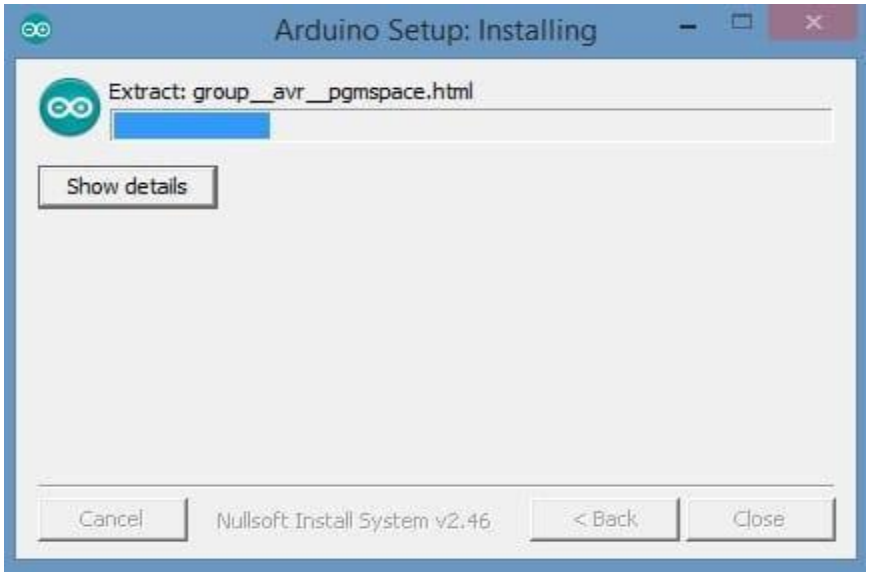


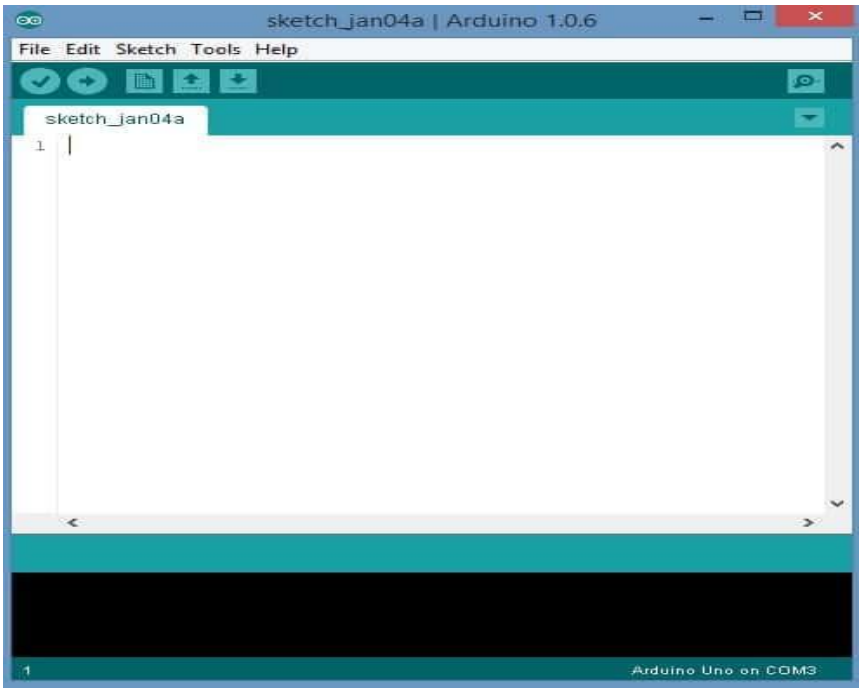
6. Select which folder to install the program to, then click “Install”:



7. Wait for the program to finish installing, and then click “Close”:

8. Now find the Arduino shortcut on your Desktop and click on it. The IDE will open up and you’ll see the code editor:

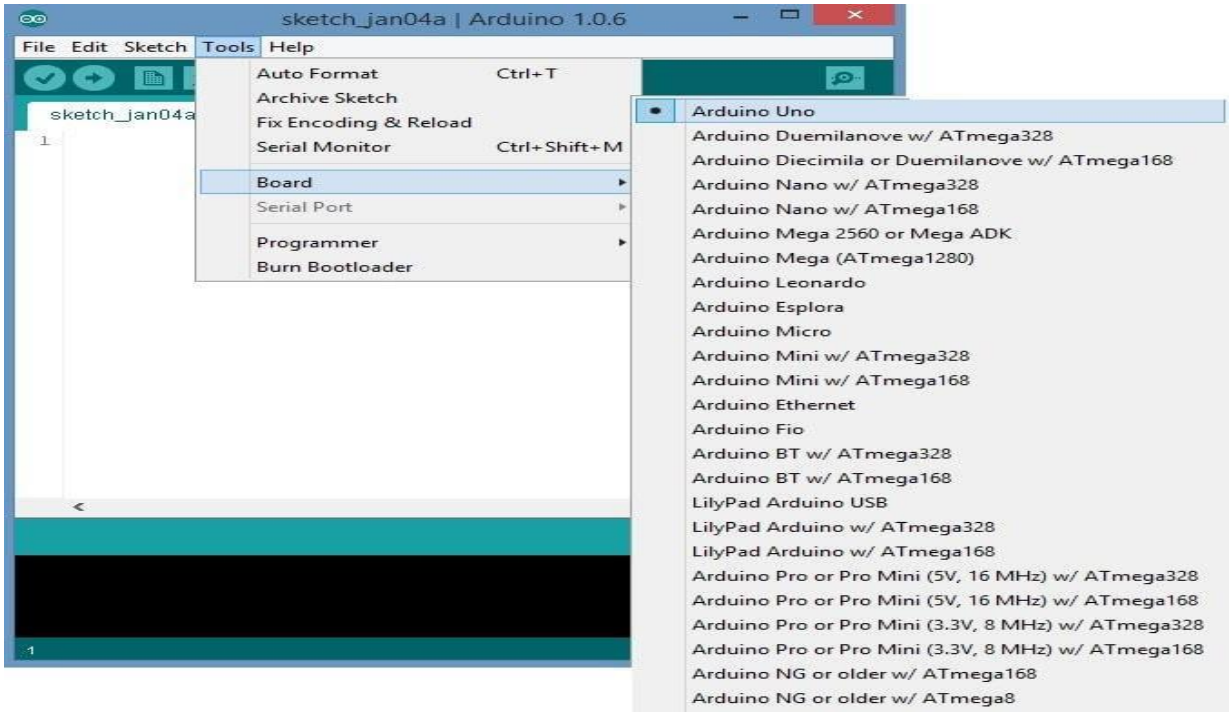




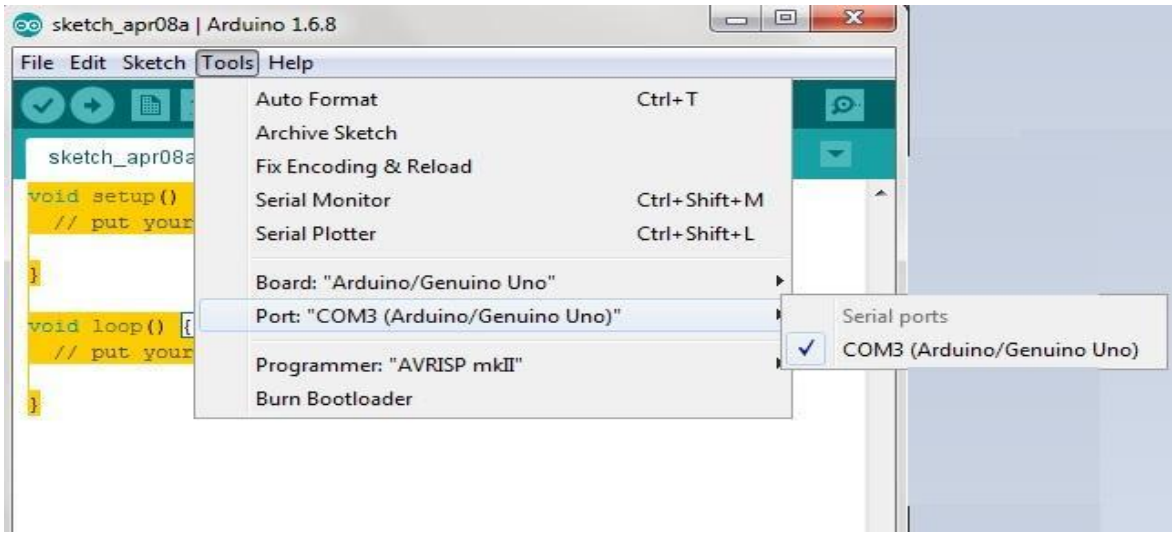
CONFIGURING THE ARDUINO IDE

The next thing to do is to make sure the software is set up for your particular Arduino board. Go to the "Tools" drop-down menu, and find "Board". Another menu will appear where you can select from a list of Arduino models. I have the Arduino Uno R3, so I chose "Arduino Uno".

Selecting arduino board

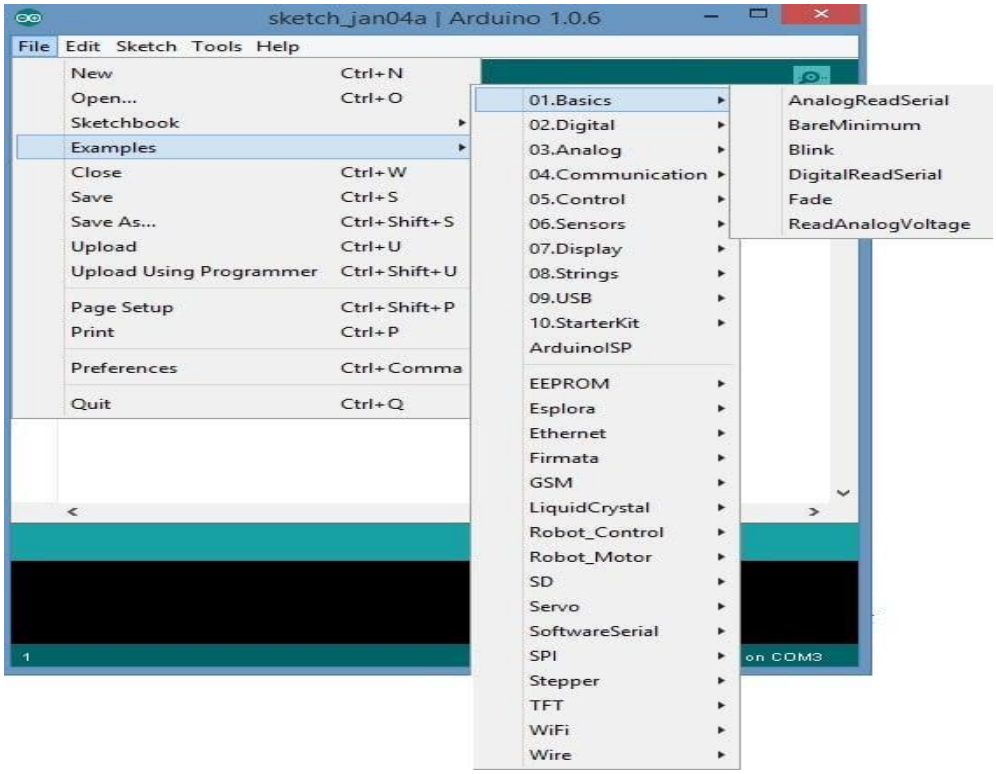


Selecting arduino port



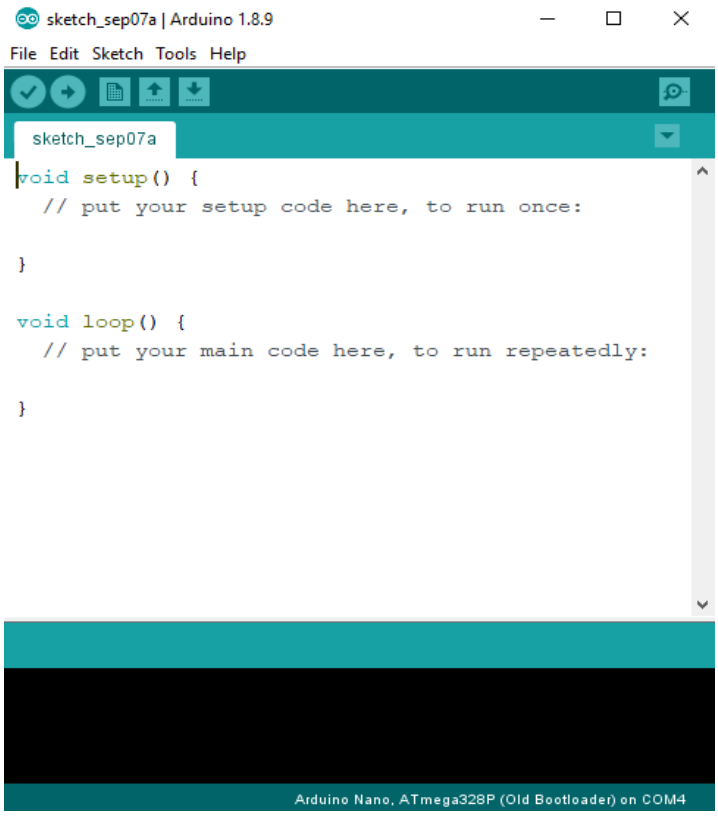
EXPLORING THE ARDUINO IDE

If you want, take a minute to browse through the different menus in the IDE. There is a good variety of example programs that come with the IDE in the "Examples" menu. These will help you get started with your Arduino right away without having to do lots of research:



Running the Arduino IDE Software

This is a display of the Arduino IDE Software. The application is ready to be used to create amazing projects.

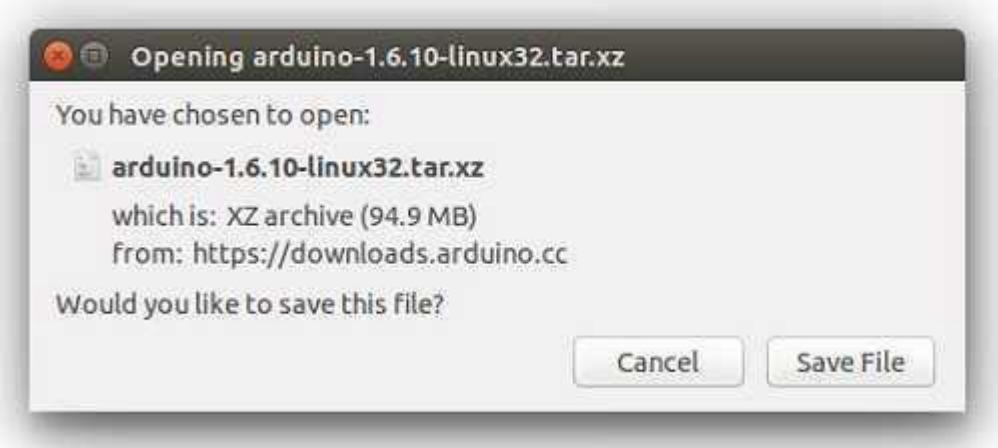


Install the Arduino Software (IDE) on Linux

The Linux build of the Arduino Software (IDE) comes in different packages depending on your system architecture. There are no specific instructions needed for the different distributions of Linux (e.g. Ubuntu).

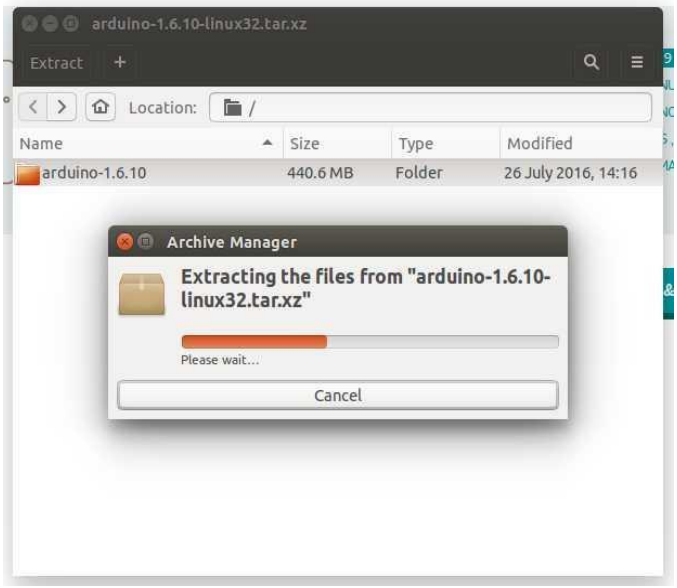
Download the Arduino Software (IDE)

Get the latest version from the download page. You can choose between the 32, 64 and ARM versions. It is very important that you choose the right version for your Linux distro. Clicking on the chosen version brings you to the donation page and then you can either open or save the file. Please save it on your computer.



Extract the package

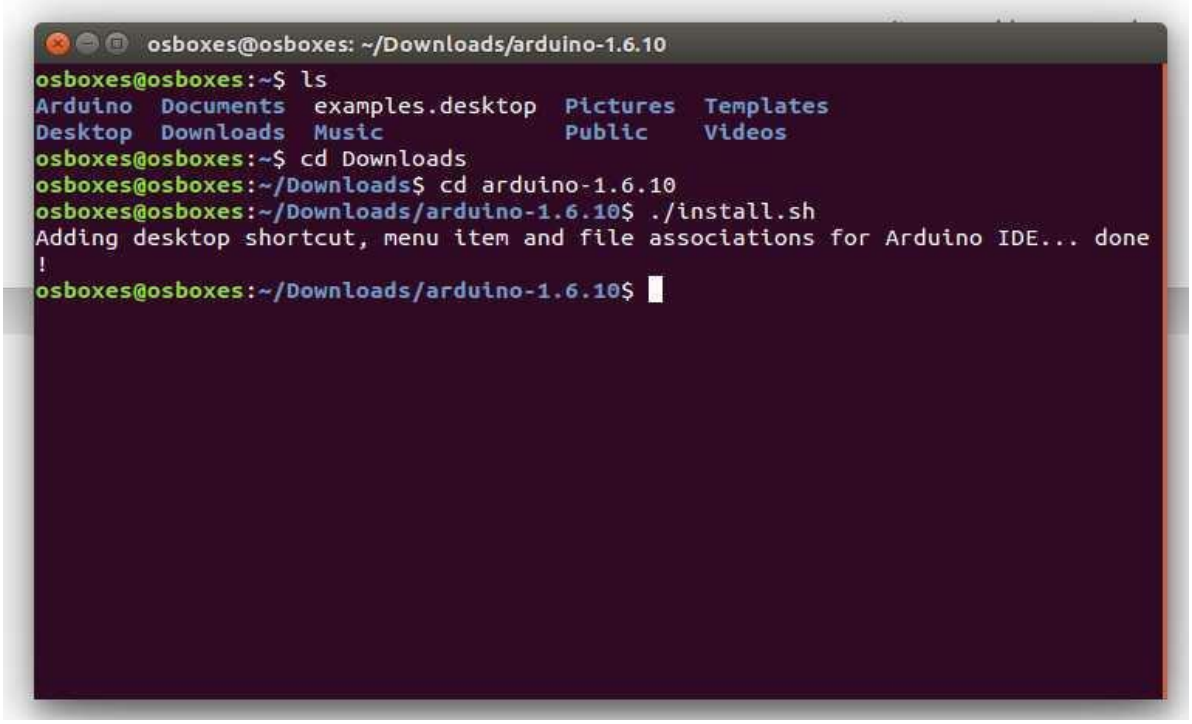
The file is compressed and you have to extract it in a suitable folder, remembering that it will be executed from there.



Run the install script

Open the **arduino-1.6.x** folder just created by the extraction process and spot the **install.sh** file. Right click on it and choose **Run in Terminal** from the contextual menu. The installation process will quickly end and you should find a new icon on your desktop.

If you don't find the option to run the script from the contextual menu, you have to open a Terminal window and move into the **arduino-1.6.x** folder. Type the command **./install.sh** and wait for the process to finish. You should find a new icon on your desktop.



Proceed with board-specific instructions

When the Arduino Software (IDE) is properly installed you can go back to the Getting Started Home and choose your board from the list on the right of the page.

Please Read...

It might happen that when you upload a sketch - after you have selected your board and the serial port -, you get an error *Error opening serial port* ... If you get this error, you need to set serial port permission.

Open Terminal and type:

ls -l /dev/ttyACM*

you will get something like:

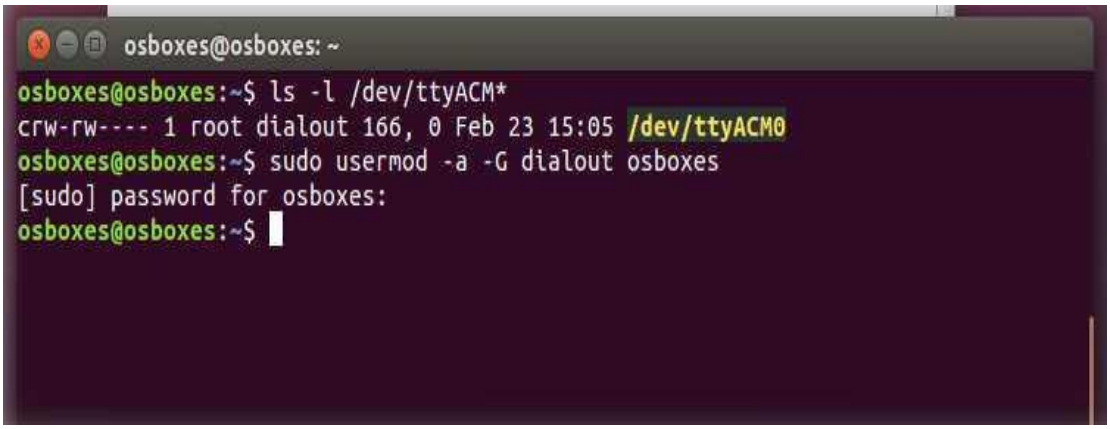
crw-rw---- 1 root dialout 188, 0 5 apr 23.01 ttyACM0

The "0" at the end of ACM might be a different number, or multiple entries might be returned. The data we need is "dialout" (is the group owner of the file).

Now we just need to add our user to the group:

sudousermod -a -G dialout<username>

where <username> is your Linux user name. **You will need to log out and log in again for this change to take effect.**



This is the procedure to access the serial port from the Arduino Software (IDE) if you get an error

After this procedure, you should be able to proceed normally and upload the sketch to your board or use the Serial Monitor.

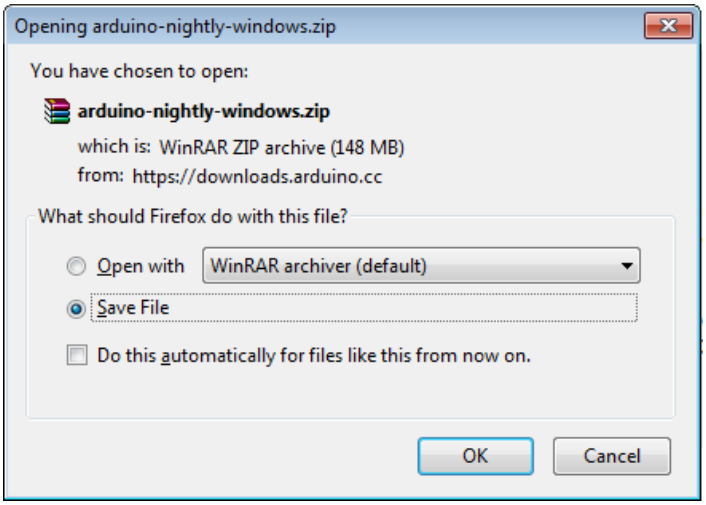
Steps to connect Arduino board:

Step 1 − First you must have your Arduino board (you can choose your favorite board) and a USB cable. In case you use Arduino UNO, Arduino Duemilanove, Nano, Arduino Mega 2560, or Diecimila, you will need a standard USB cable (A plug to B plug), the kind you would connect to a USB printer as shown in the following image.

In case you use Arduino Nano, you will need an A to Mini-B cable instead as shown in the following image.

Step 2 − Download Arduino IDE Software.

You can get different versions of Arduino IDE from the Download page on the Arduino Official website. You must select your software, which is compatible with your operating system (Windows, IOS, or Linux). After your file download is complete, unzip the file.



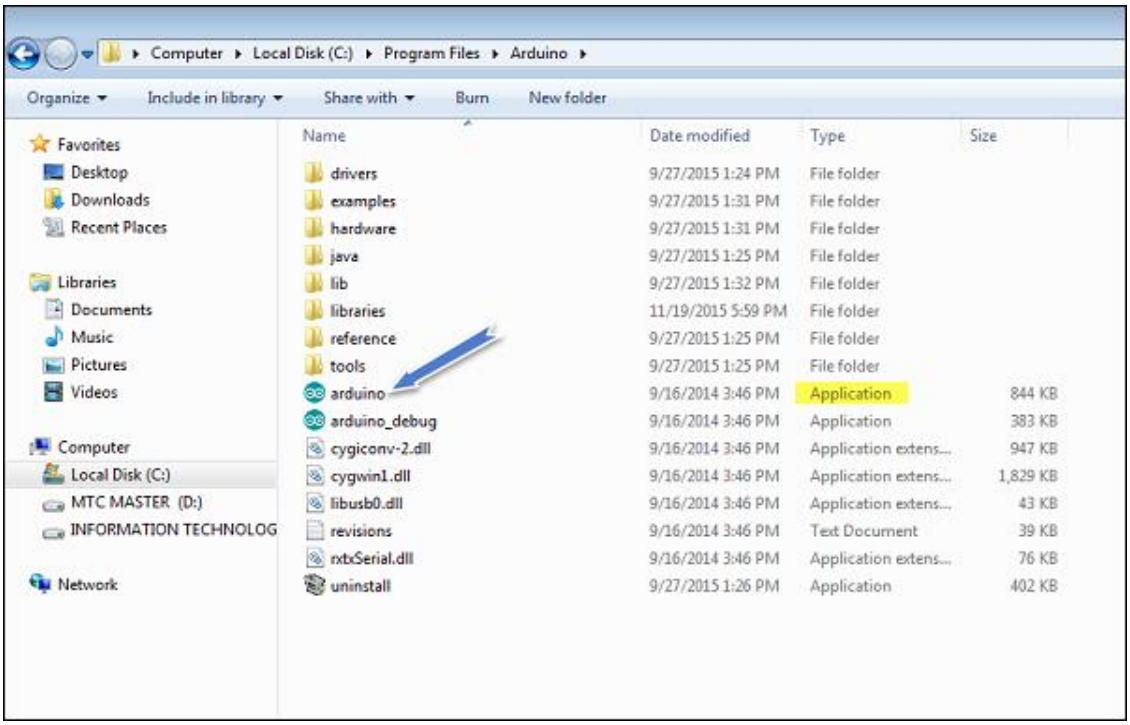
Step 3 − Power up your board.

The Arduino Uno, Mega, Duemilanove and Arduino Nano automatically draw power from either, the USB connection to the computer or an external power supply. If you are using an Arduino Diecimila, you have to make sure that the board is configured to draw power from the USB connection. The power source is selected with a jumper, a small piece of plastic that fits onto two of the three pins between the USB and power jacks. Check that it is on the two pins closest to the USB port.

Connect the Arduino board to your computer using the USB cable. The green power LED (labeled PWR) should glow.

Step 4 − Launch Arduino IDE.

After your Arduino IDE software is downloaded, you need to unzip the folder. Inside the folder, you can find the application icon with an infinity label (application.exe). Double-click the icon to start the IDE.

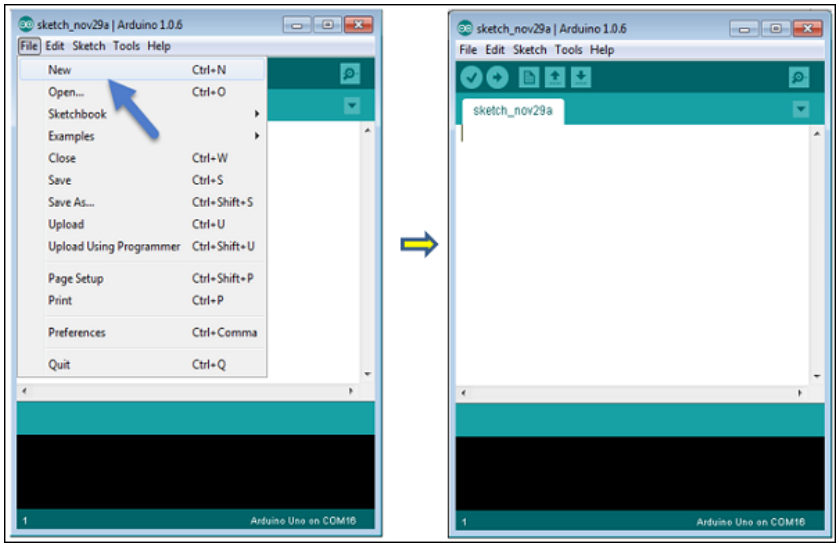


Step 5 − Open your first project.

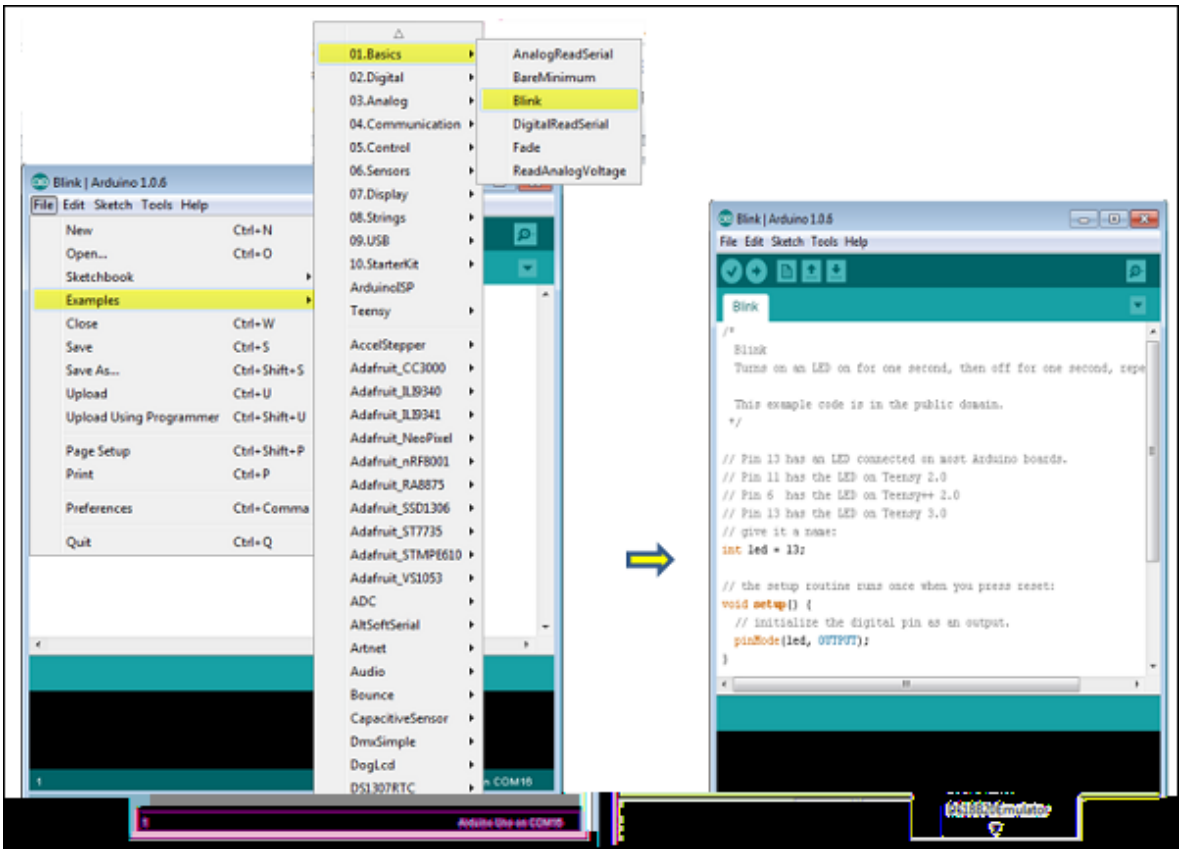
Once the software starts, you have two options −

- Create a new project.
- Open an existing project example.

To create a new project, select File → **New**.



To open an existing project example, select File → Example → Basics → Blink.

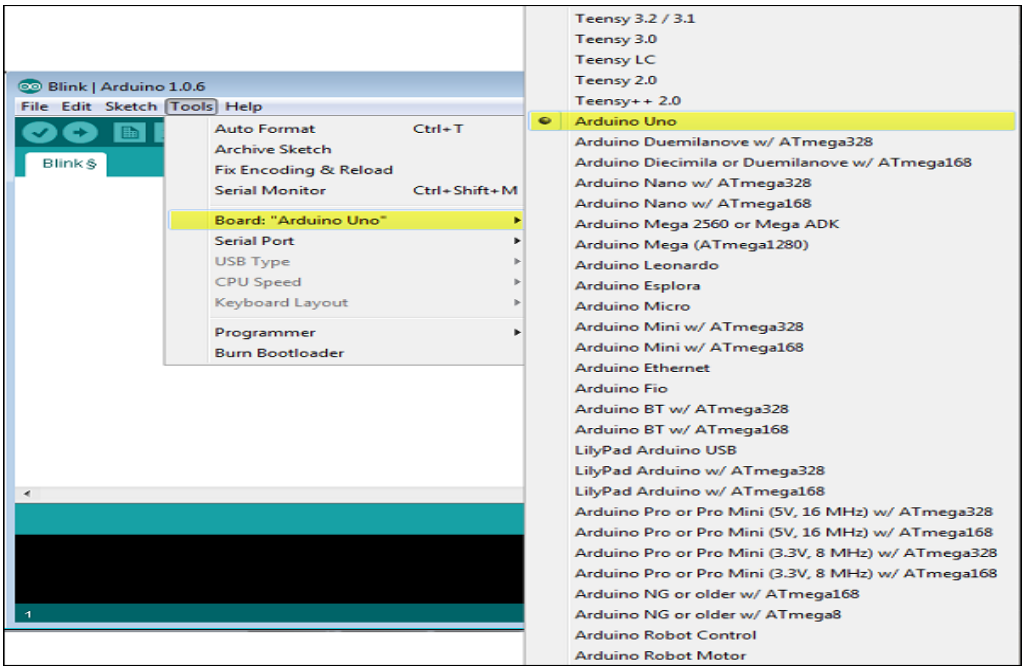


Here, we are selecting just one of the examples with the name Blink. It turns the LED on and off with some time delay. You can select any other example from the list.

Step 6 − Select your Arduino board.

To avoid any error while uploading your program to the board, you must select the correct Arduino board name, which matches with the board connected to your computer.

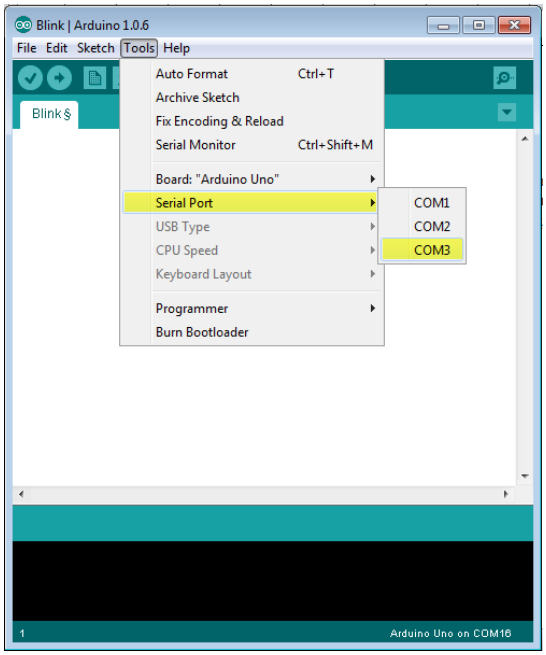
Go to Tools → Board and select your board.



Here, we have selected Arduino Uno board according to our tutorial, but you must select the name matching the board that you are using.

Step 7 − Select your serial port.

Select the serial device of the Arduino board. Go to Tools → Serial Port menu. This is likely to be COM3 or higher (COM1 and COM2 are usually reserved for hardware serial ports). To find out, you can disconnect your Arduino board and re-open the menu, the entry that disappears should be of the Arduino board. Reconnect the board and select that serial port.



Step 8 − Upload the program to your board.

Before explaining how we can upload our program to the board, we must demonstrate the function of each symbol appearing in the Arduino IDE toolbar.

A − Used to check if there is any compilation error.

B − Used to upload a program to the Arduino board.

C − Shortcut used to create a new sketch.

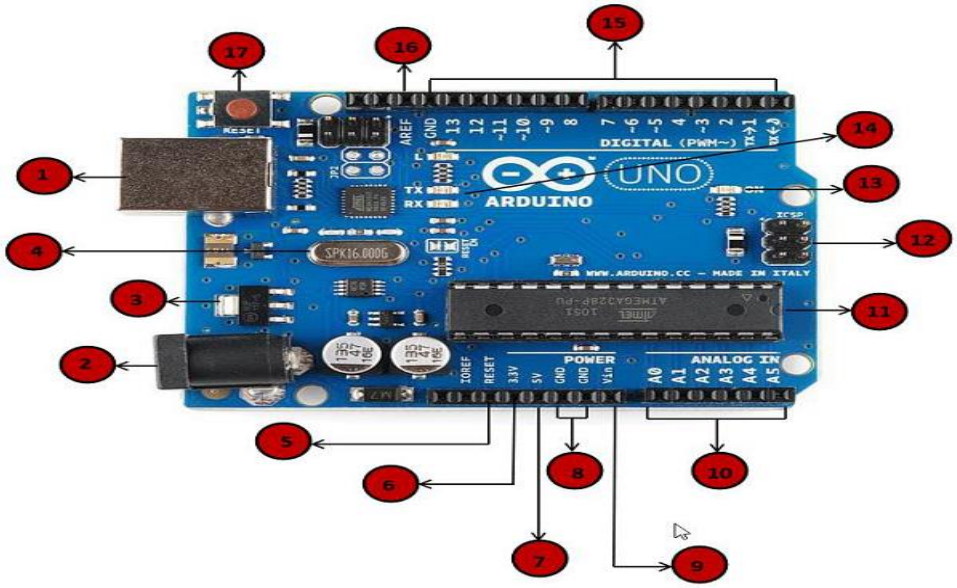
D − Used directly open one of the example sketch.

E − Used to save your sketch.

F − Serial monitor used to receive serial data from the board and send the serial data to the board.

Now, simply click the "Upload" button in the environment. Wait a few seconds; you will see the RX and TX LEDs on the board, flashing. If the upload is successful, the message "Done uploading" will appear in the status bar.

Arduino uno Board



1.Power USB

Arduino board can be powered by using the USB cable from your computer. All you need to do is connect the USB cable to the USB connection (1).

2. Power (Barrel Jack)

Arduino boards can be powered directly from the AC mains power supply by connecting it to the Barrel Jack (2).

3. Voltage Regulator

The function of the voltage regulator is to control the voltage given to the Arduino board and stabilize the DC voltages used by the processor and other elements.

4. Crystal Oscillator

The crystal oscillator helps Arduino in dealing with time issues. How does Arduino calculate time? The answer is, by using the crystal oscillator. The number printed on top of the Arduino crystal is 16.000H9H. It tells us that the frequency is 16,000,000 Hertz or 16 MHz.

5, 17- Arduino Reset

You can reset your Arduino board, i.e., start your program from the beginning. You can reset the UNO board in two ways. First, by using the reset button (17) on the board. Second, you can connect an external reset button to the Arduino pin labelled RESET (5).

6,7,8,9: Pins (3.3, 5, GND, Vin)

- 3.3V (6) − Supply 3.3 output volt
  5V (7) − Supply 5 output volt
- Most of the components used with Arduino board works fine with 3.3 volt and 5 volt.
- GND (8)(Ground) − There are several GND pins on the Arduino, any of which can be used to groundyour circuit.
- Vin (9) − This pin also can be used to power the Arduino board from an external power source, like AC mains power supply.

10. Analog pins

The Arduino UNO board has six analog input pins A0 through A5. These pins can read the signal from an analog sensor like the humidity sensor or temperature sensor and convert it into a digital value that can be read by the microprocessor.

11.Main microcontroller

Each Arduino board has its own microcontroller (11). You can assume it as the brain of your board. The main IC (integrated circuit) on the Arduino is slightly different from board to board. The microcontrollers are usually of the ATMEL Company. You must know what IC your board has before loading up a new program from the Arduino IDE. This information is available on the top of the IC. For more details about the IC construction and functions, you can refer to the data sheet.

12. ICSP pin

Mostly, ICSP (12) is an AVR, a tiny programming header for the Arduino consisting of MOSI, MISO, SCK, RESET, VCC, and GND. It is often referred to as an SPI (Serial Peripheral Interface), which could be considered as an "expansion" of the output. Actually, you are slaving the output device to the master of the SPI bus.

13. Power LED indicator

This LED should light up when you plug your Arduino into a power source to indicate that your board is powered up correctly. If this light does not turn on, then there is something wrong with the connection.

14. TX and RX LEDs

On your board, you will find two labels: TX (transmit) and RX (receive). They appear in two places on the Arduino UNO board. First, at the digital pins 0 and 1, to indicate the pinsresponsible for serial communication. Second, the TX and RX led (13). The TX led flashes with different speed while sending the serial data. The speed of flashing depends on the baud rate used by the board. RX flashes during the receiving process.

15. Digital I/O

The Arduino UNO board has 14 digital I/O pins (15) (of which 6 provide PWM (Pulse Width Modulation) output. These pins can be configured to work as input digital pins to read logic values (0 or 1) or as digital output pins to drive different modules like LEDs, relays, etc. The pins labeled "~" can be used to generate PWM.

16. AREF

AREF stands for Analog Reference. It is sometimes, used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins.

Sketch − The first new terminology is the Arduino program called "**sketch**".

Structure

Arduino programs can be divided in three main parts: **Structure, Values** (variables and constants), and **Functions**. In this tutorial, we will learn about the Arduino software program, step by step, and how we can write the program without any syntax or compilation error.

Let us start with the **Structure**. Software structure consists of two main functions −

- Setup( ) function
- Loop( ) function

```
Void setup ( ) {

}
```

- **PURPOSE** − The **setup()** function is called when a sketch starts. Use it to initialize the variables, pin modes, start using libraries, etc. The setup function will only run once, after each power up or reset of the Arduino board.
- **INPUT** − -
- **OUTPUT** − -
- **RETURN** − -

```
Void Loop ( ) {
                    }
```

- **PURPOSE** − After creating a **setup()** function, which initializes and sets the initial values, the **loop()** function does precisely what its name suggests, and loops consecutively, allowing your program to change and respond. Use it to actively control the Arduino board.
- **INPUT** − -
- **OUTPUT** − -
- **RETURN** − -

Viva Questions:

1. What is the name of program written with the Arduino IDE?

2. What does p refers to in ATmega328p?

3. What are the two functions in Arduino Sketch?

4. How many digital pins are there in the Arduino Uno board?

5. What is the Stable version of Arduino software?

WEEK-2 **Date:**

AIM: To interface LED/Buzzer with Arduino/Raspberry Pi and write a program to turn ON LED for 1 sec after every 2 seconds.

Hardware Requirements:

1. 1x Breadboard
2. 1x Arduino Uno R3
3. 1x RGB LED
4.2x Jumper Wires

Procedure:

1. Connect the Arduino board to your computer using the USB cable.
2. Set the pin-mode as LED output
3. Set the pin-mode as Buzzer output.
4. Set the delay time for output
5. Set the digital pin-mode on.
6. Set the digital pin-mode off.

Source Code:

```
int led=13;
int buzzer=11;
void setup() {
pinMode(13, OUTPUT);
pinMode(11, OUTPUT);

}
void loop() {
digitalWrite(13, HIGH);
digitalWrite(11, HIGH);
delay(1000);
digitalWrite(13, LOW);
digitalWrite(11, LOW);
delay(2000);
}
```

OUTPUT:

Viva Questions

1. **What is the use of pinMode() in Arduino IDE?**

2. **In which language Arduino software was written?**

3. **What are the functions of time in Arduino?**

4. **How many times does the setup() function run on every startup of the Arduino System?**

5. **How many times does the loop() function run on every startup of the Arduino System?**

WEEK-3: **Date:**

AIM: Write an Arduino sketch to measure the distance (in cm's) of a certainobject.

Ultrasonic Sensor:

The HC-SR04 ultrasonic sensor uses SONAR to determine the distance of an object just like the bats do. It offers excellent non-contact range detection with high accuracy and stable readings in an easy-to-use package from 2 cm to 400 cm or 1" to 13 feet.

The operation is not affected by sunlight or black material, although acoustically, soft materials like cloth can be difficult to detect. It comes complete with ultrasonic transmitter and receiver module.

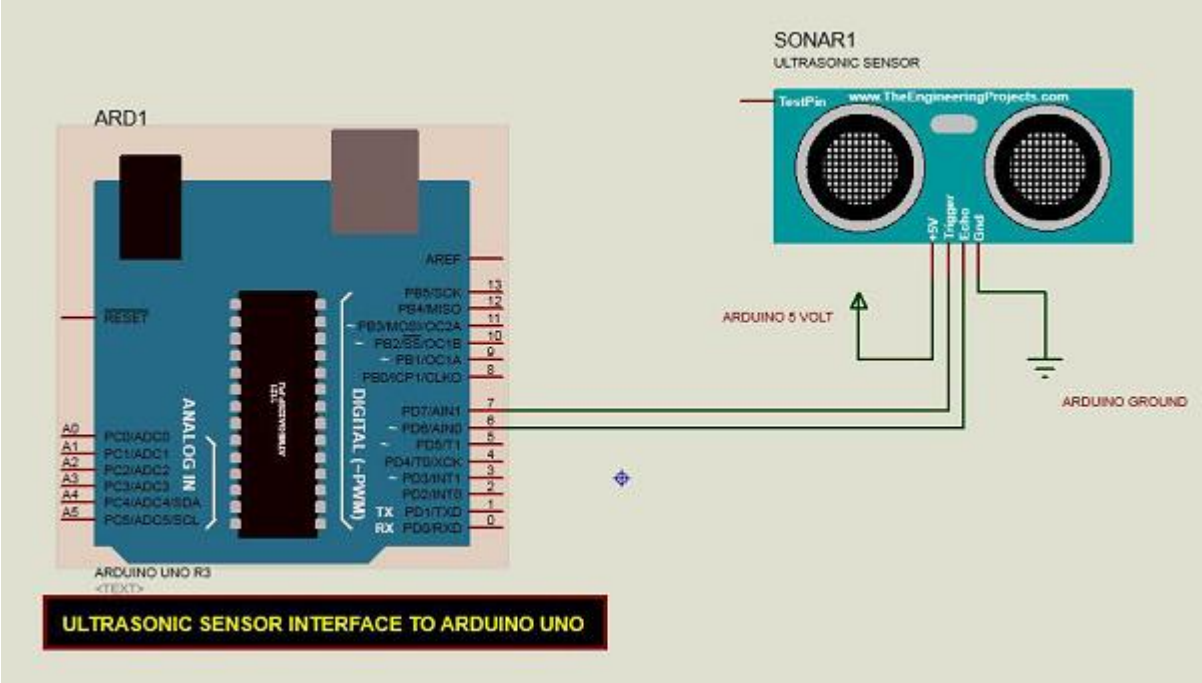
Ultrasonic Sensor

Hardware Requirements:

1. 4 x jumping wires(male to female)
2. 1 x Arduino Uno R3
3. 1 x ULTRASONIC Sensor (HC-SR04)

Procedure:

Follow the circuit diagram and make the connections as shown in the image given below.



The Ultrasonic sensor has four terminals - +5V, Trigger, Echo, and GND connected as follows −

1. Connect the +5V pinto +5v on your Arduino board.
2. Connect Trigger to digital pin 7 on your Arduino board.
3. Connect Echo to digital pin 6 on your Arduino board.
4. Connect GND with GND on Arduino.

Source code:

```
const int pingPin = 7; // Trigger Pin of Ultrasonic Sensor
const int echoPin = 6; // Echo Pin of Ultrasonic Sensor
void setup() {
Serial.begin(9600); // Starting Serial Terminal
}
void loop() {
  long duration, inches, cm;
pinMode(pingPin, OUTPUT);
digitalWrite(pingPin, LOW);
delayMicroseconds(2);
digitalWrite(pingPin, HIGH);
delayMicroseconds(10);
digitalWrite(pingPin, LOW);
pinMode(echoPin, INPUT);
  duration = pulseIn(echoPin, HIGH);
  inches = microsecondsToInches(duration);
  cm = microsecondsToCentimeters(duration);
Serial.print(inches);
Serial.print("in, ");
Serial.print(cm);
Serial.print("cm");
Serial.println();
delay(100); }
```

```
long microsecondsToInches(long microseconds) {
  return microseconds / 74 / 2;
}
long microsecondsToCentimeters(long microseconds) {
  return microseconds / 29 / 2;
}
```

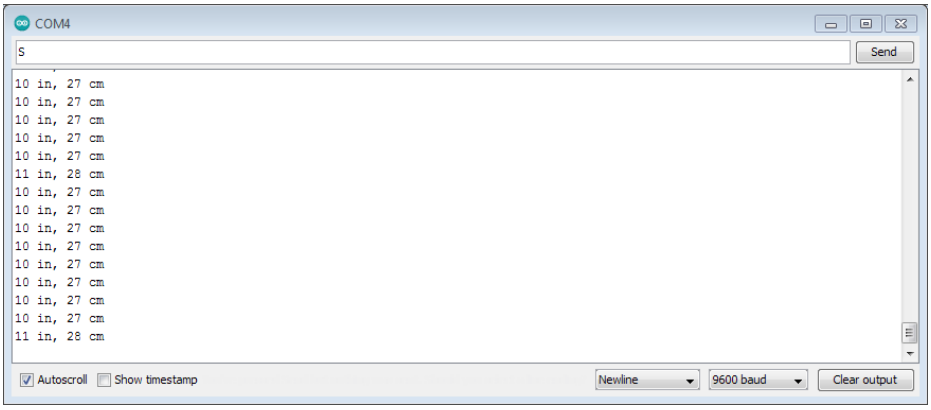
Procedure to see output:

1. Upload code.
2. Open serial monitor under tools menu.
3. Output will be shown.

OUTPUT:

To see output:

Go to tools -> Serial Monitor



Viva Questions:

1. What is the unit of delay() in Arduino?

2. What is the Use of Ultrasonic Sensor?

3. How many pins are present in the Ultrasonic Sensor?

4. What mode should we put the Arduino pin to , in order for object detection to work with the Ultrasonic Sensor?

5. What kind of waves does the Ultrasonic Sensor works on?

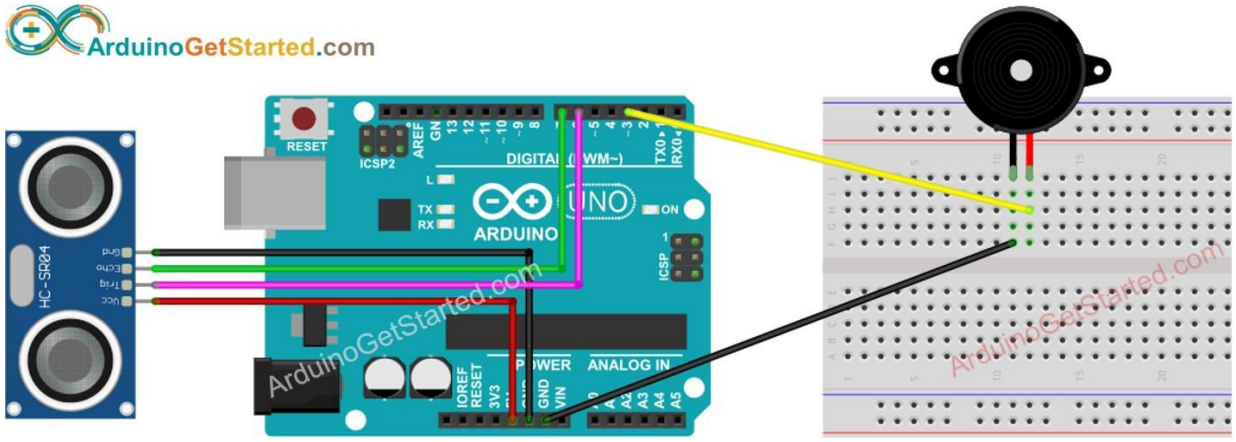
WEEK-4: **Date:**

i. **AIM** :Write an Arduino sketch to blink an LED and a buzzer if the distance measured is less than a threshold value.

Hardware Requirements:

1. 1x Breadboard
2. 1x Arduino Uno R3
3. 1x Buzzer
4. 6x Jumper Wires
5. 1 x ULTRASONIC Sensor (HC-SR04)

Pin diagram:



Procedure:

1. Arduino GND to Ultrasonic sensor GND.
2. Vcc of Ultrasonic to +5V of Arduino.
3. Echo of Ultrasonic to pin 7 of Arduino.
4. Trig of Ultrasonic to pin 6 of Arduino.
5. Cathode of Buzzer to pin 3 of Arduino.
6. Anode of Buzzer to GND of Arduino.

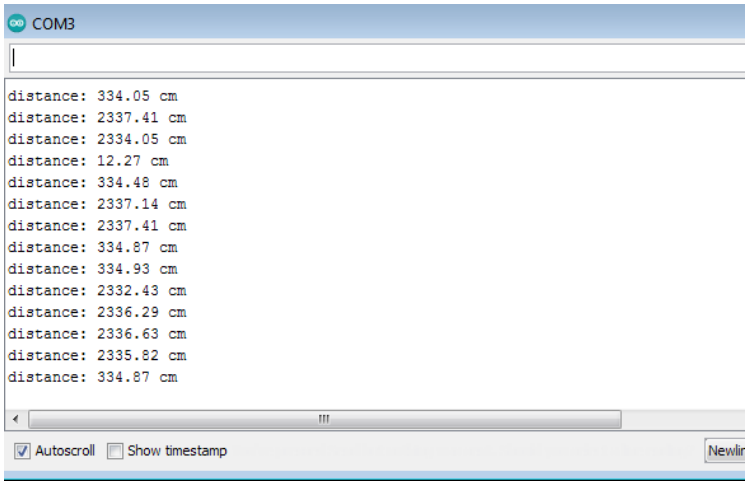
Source Code:

```
const int TRIG_PIN   = 6; // Arduino pin connected to Ultrasonic Sensor's TRIG pin
const int ECHO_PIN   = 7; // Arduino pin connected to Ultrasonic Sensor's ECHO pin
const int BUZZER_PIN = 3; // Arduino pin connected to Piezo Buzzer's pin
const int DISTANCE_THRESHOLD = 50; // centimeters
const int LED_PIN = 13;

// variables will change:
float duration_us, distance_cm;

void setup() {
Serial.begin (9600);        // initialize serial port
pinMode(TRIG_PIN, OUTPUT);   // set arduino pin to output mode
pinMode(ECHO_PIN, INPUT);    // set arduino pin to input mode
pinMode(BUZZER_PIN, OUTPUT);
pinMode(LED_PIN, OUTPUT);
     // set arduino pin to output mode
}

void loop() {
  // generate 10-microsecond pulse to TRIG pin
digitalWrite(TRIG_PIN, HIGH);
delayMicroseconds(10);
digitalWrite(TRIG_PIN, LOW);

  // measure duration of pulse from ECHO pin
duration_us = pulseIn(ECHO_PIN, HIGH);
  // calculate the distance
distance_cm = 0.017 * duration_us;

if(distance_cm< DISTANCE_THRESHOLD){
digitalWrite(BUZZER_PIN, HIGH);
digitalWrite(LED_PIN, HIGH);
     // turn on Piezo Buzzer
     }
else{
digitalWrite(BUZZER_PIN, LOW);  // turn off Piezo Buzzer
digitalWrite(LED_PIN, LOW);
  }
  // print the value to Serial Monitor
Serial.print("distance: ");
Serial.print(distance_cm);
Serial.println(" cm");

delay(500);
}
```

OUTPUT:

To see output:

Go to tools -> Serial Monitor

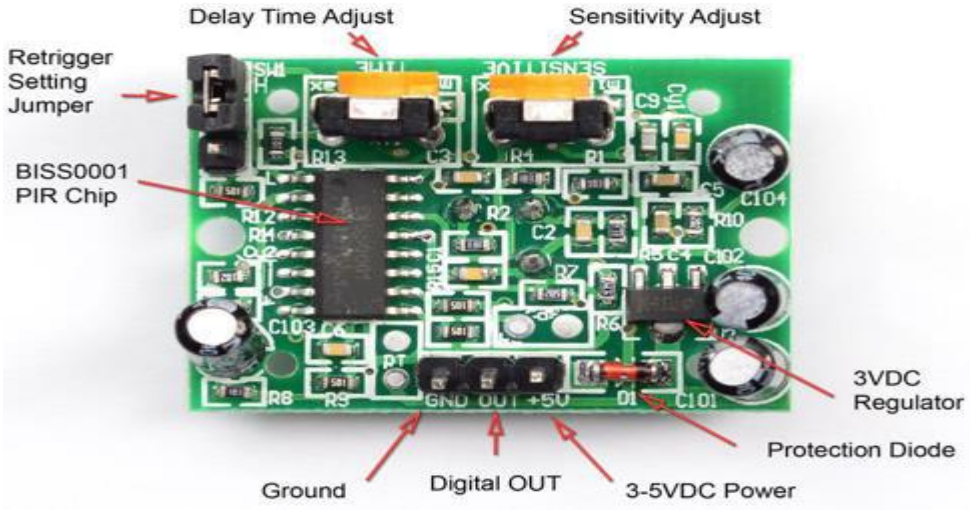


ii. AIM :Write an Arduino sketch to Illustrate the working of PIR Sensor with an example.

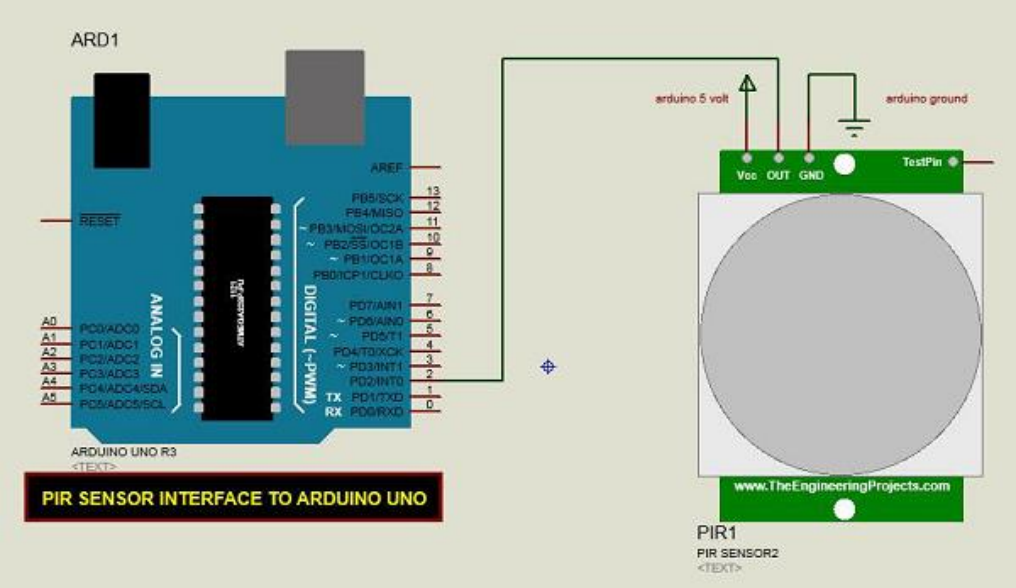
Hardware Requirements:

1. 1x Arduino Uno R3
2. 3x Jumper Wires
3. 1 x PIR Sensor

PIR Sensor:



Pin Diagram:



Procedure:

1. Arduino GND to PIR sensor GND.
2. Vcc of PIR to +5V of Arduino.
3. Out of PIR to pin 2 of Arduino.

Source Code:

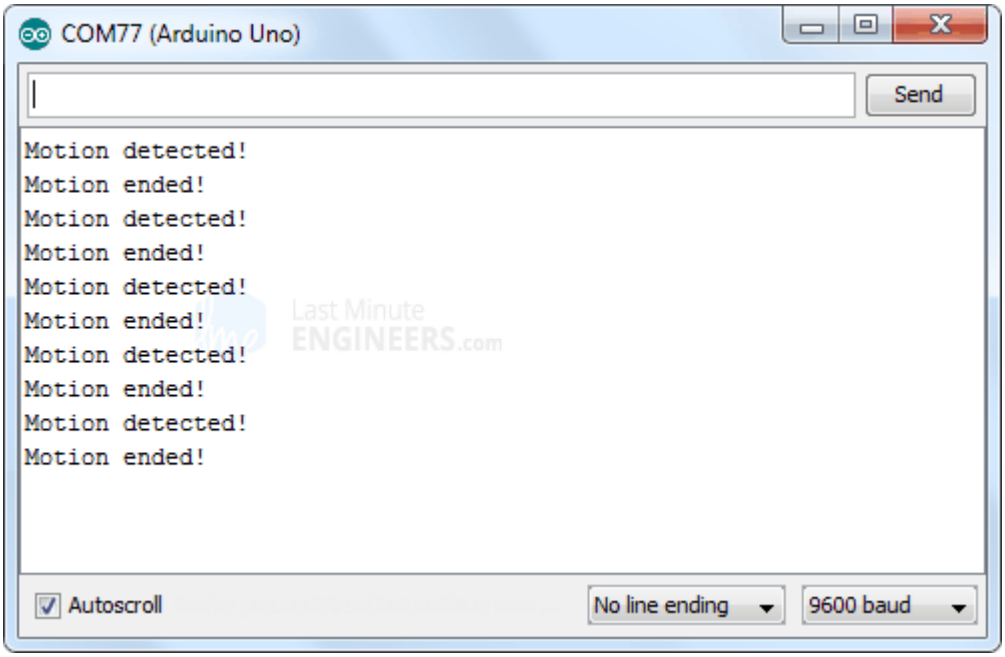
```
#definepirPin 2
int calibrationTime = 30;
long unsigned int lowIn;
long unsigned int pause = 5000;
booleanlockLow = true;
booleantakeLowTime;
int PIRValue = 0;

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

void loop() {
PIRSensor();
}
```

```
void PIRSensor() {
  if(digitalRead(pirPin) == HIGH) {
    if(lockLow) {
PIRValue = 1;
lockLow = false;
Serial.println("Motion detected.");
delay(1);
    }
takeLowTime = true;
  }
  if(digitalRead(pirPin) == LOW) {
    if(takeLowTime){
lowIn = millis();takeLowTime = false;
    }
if(!lockLow&&millis() - lowIn> pause) {
PIRValue = 0;
lockLow = true;
Serial.println("Motion ended.");
delay(1);
    }
  }
}
```

OUTPUT:

To see output:

Go to tools -> Serial Monitor



iii. AIM : Write an Arduino sketch to Illustrate the IR and DHT Sensor

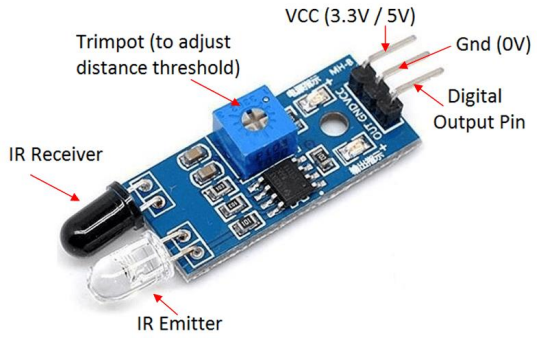
IR Sensor:

Hardware Requirements:

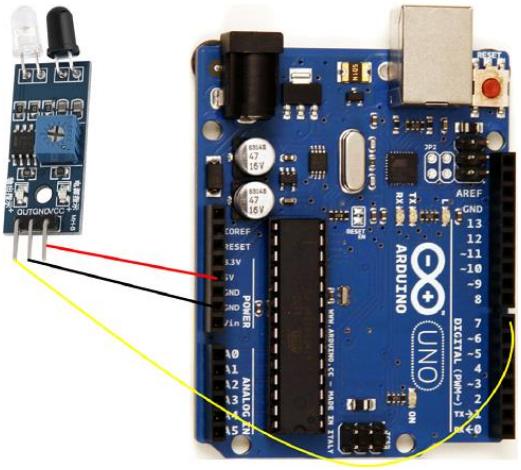
1. 1x Arduino Uno R3
2. 3x Jumper Wires
3. 1 x IR Sensor
4. 1 x LED

Procedure:

1. Arduino GND to IR sensor GND.
2. Vcc of IR to +5V of Arduino.
3. Out of IR to pin 7 of Arduino.
4. LED to Pin 13 of Arduino.



Pin diagram



Source Code:

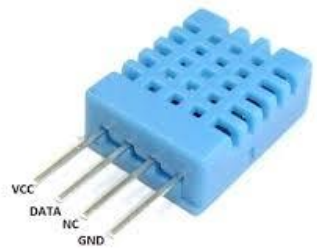
```
void setup() {
pinMode(7,INPUT);
Serial.begin(9600);
pinMode(13,OUTPUT);
}

void loop() {
Serial.print("IRSensorip  ");
Serial.println(digitalRead(7));
if(digitalRead(7)==0)
{
digitalWrite(13,HIGH);
  }
else{
digitalWrite(13,LOW);
    }

}
```

OUTPUT:

If there is any obstacle ahead the LED will glow.

DHT Sensor

	DHT11
	
Temperature range	0 to 50 °C +/-2 °C
Humidity range	20 to 90% +/-5%
Resolution	Humidity: 1% Temperature: 1°C
Operating voltage	3 – 5.5 V DC
Current supply	0.5 – 2.5 mA
Sampling period	1 second



Hardware Requirements:

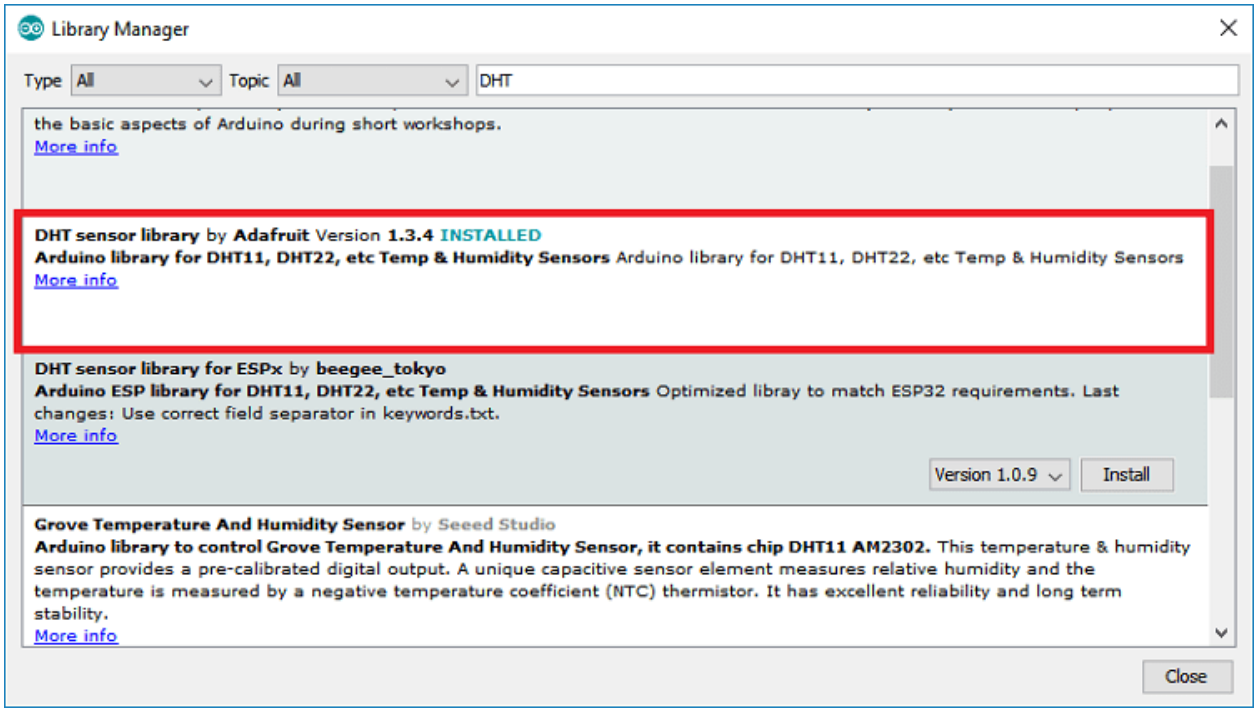
1. 1x Arduino Uno R3
2. 3x Jumper Wires
3. 1 x DHT Sensor

Procedure:

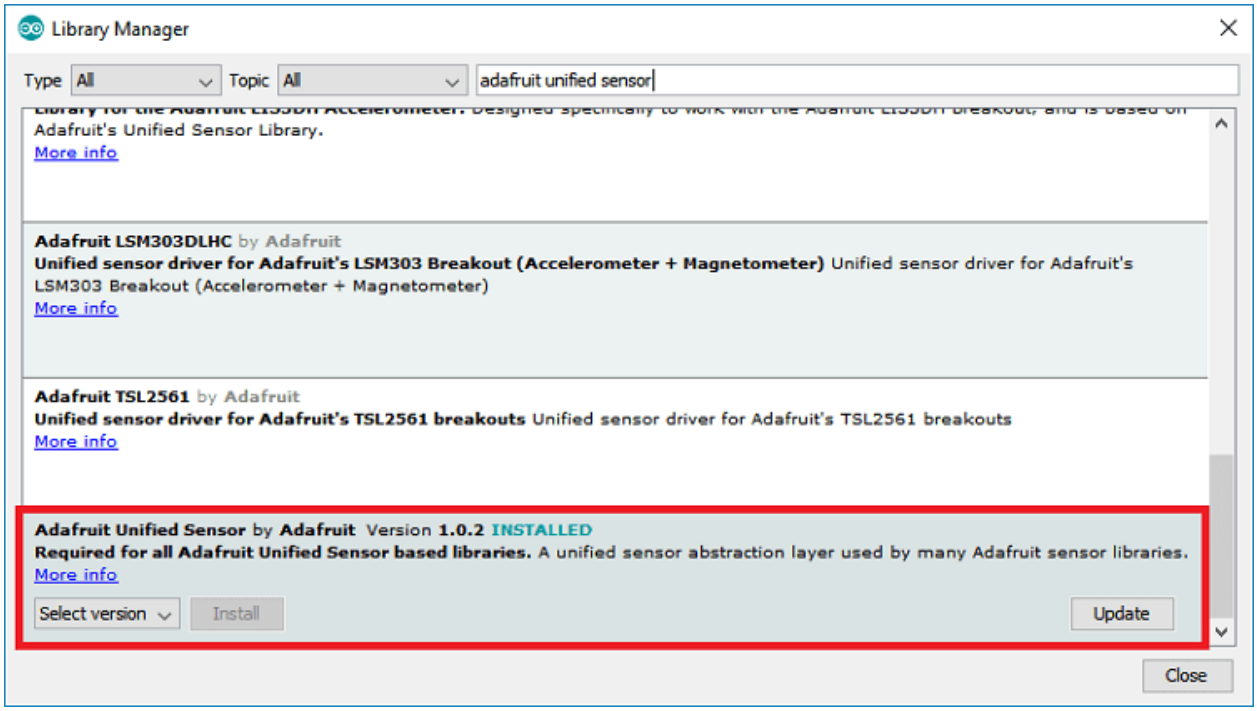
1. Arduino GND to DHT sensor GND.
2. Vcc of IR to +5V of Arduino.
3. DATA of IR to pin 2 of Arduino.

Installing Libraries:

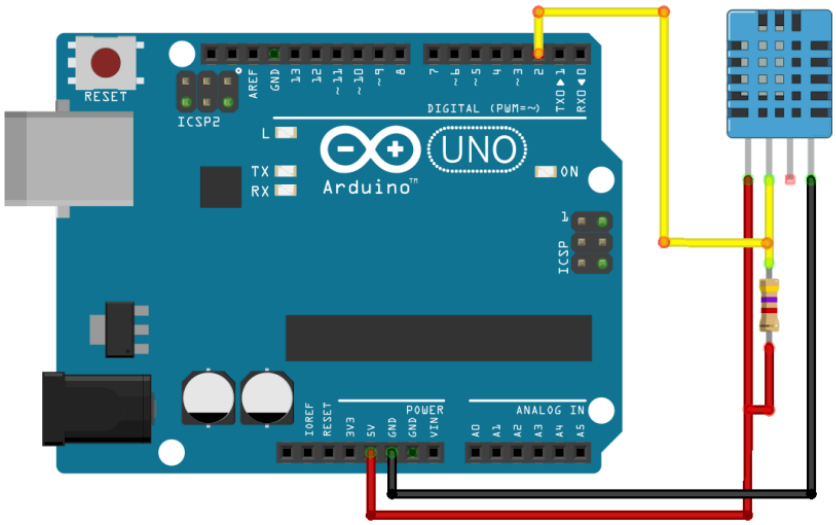
1. Open your Arduino IDE and go to **Sketch** > **Include Library** > **Manage Libraries**.
2. The Library Manager should open.
3. Search for "**DHT**" on the Search box and install the DHT library from Adafruit



4. After installing the DHT library from Adafruit, type "**Adafruit Unified Sensor**" in the search box. Scroll all the way down to find the library and install it.



Pin Diagram:



Procedure:

1. Vcc of DHT11 to +5V of Arduino.
2. GND of DHT11 to GND.
3. DATA of DHT11 to Pin 2 of Arduino.

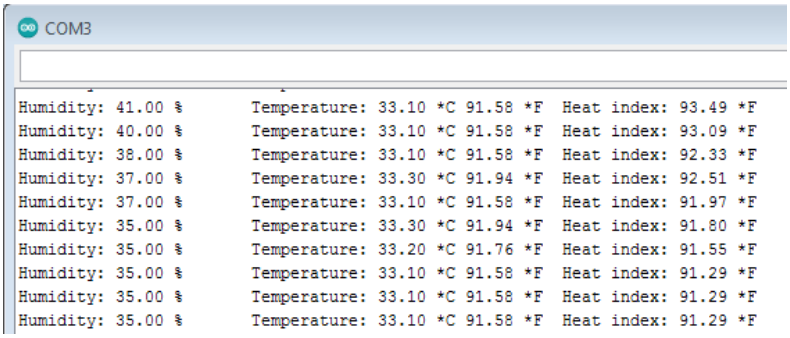
Source Code:

```
#include "DHT.h"
#define DHTPIN 2     // what pin we're connected to
#define DHTTYPE DHT11   // DHT 11
// Initialize DHT sensor for normal 16mhz Arduino
DHT dht(DHTPIN, DHTTYPE);
void setup() {
Serial.begin(9600);
Serial.println("DHTxx test!");
dht.begin();
}
void loop() {
  // Wait a few seconds between measurements.
delay(2000);
  // Reading temperature or humidity takes about 250 milliseconds!
  // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
  float h = dht.readHumidity();
  // Read temperature as Celsius
  float t = dht.readTemperature();
  // Read temperature as Fahrenheit
  float f = dht.readTemperature(true);

  // Check if any reads failed and exit early (to try again).
  if (isnan(h) || isnan(t) || isnan(f)) {
Serial.println("Failed to read from DHT sensor!");
    return;
  }
  // Compute heat index
  // Must send in temp in Fahrenheit!
  float hi = dht.computeHeatIndex(f, h);
Serial.print("Humidity: ");
Serial.print(h);
Serial.print(" %\t");
Serial.print("Temperature: ");
Serial.print(t);
Serial.print(" *C ");
Serial.print(f);
Serial.print(" *F\t");
Serial.print("Heat index: ");
Serial.print(hi);
Serial.println(" *F");
}
```

OUTPUT:

To see output:

Go to tools -> Serial Monitor



Viva Questions:

1. What is the Use of IR Sensor?

2. How many pins are present in the IR Sensor?

3. What is the full form of PIR Sensor?

4. Which type of DHT11 sensor is?

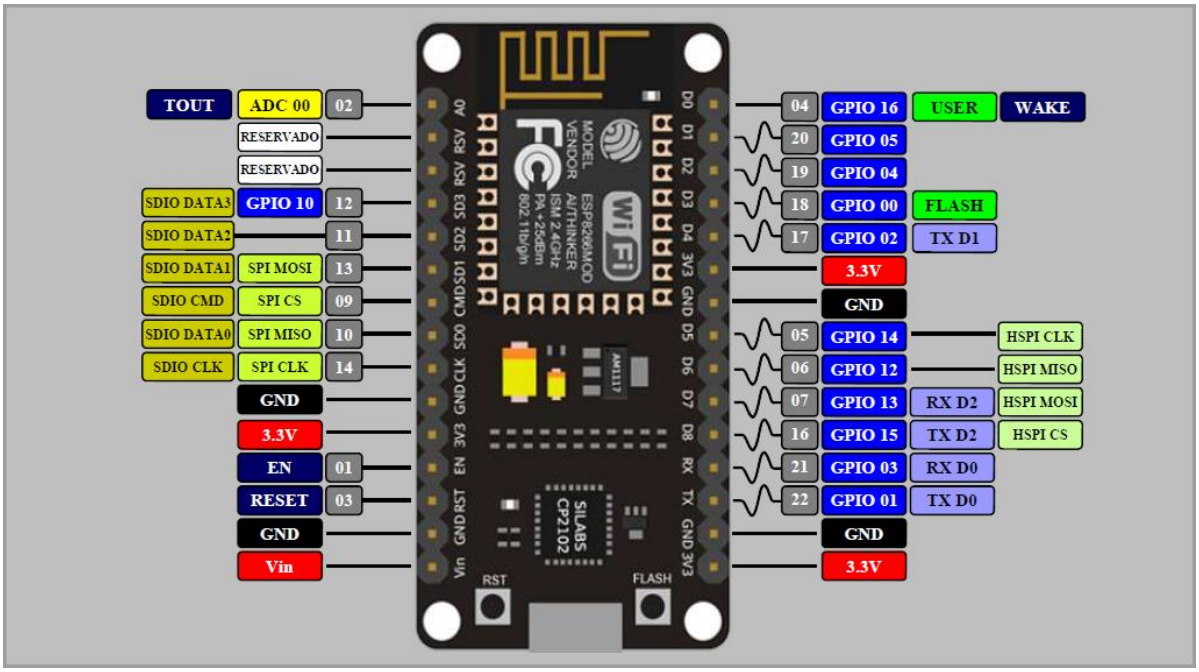
5. How many pins have does temperature sensor have?

WEEK-5 **Date:**

AIM: Write an Program to send the humidity and temperature data to Cloud (ThingSpeak).

Hardware Requirements:

1. 3 x jumping wires(female to female)
2. 1 xNodeMCU
3. 1 x DHT11 Sensor



NodeMCU Pin diagram

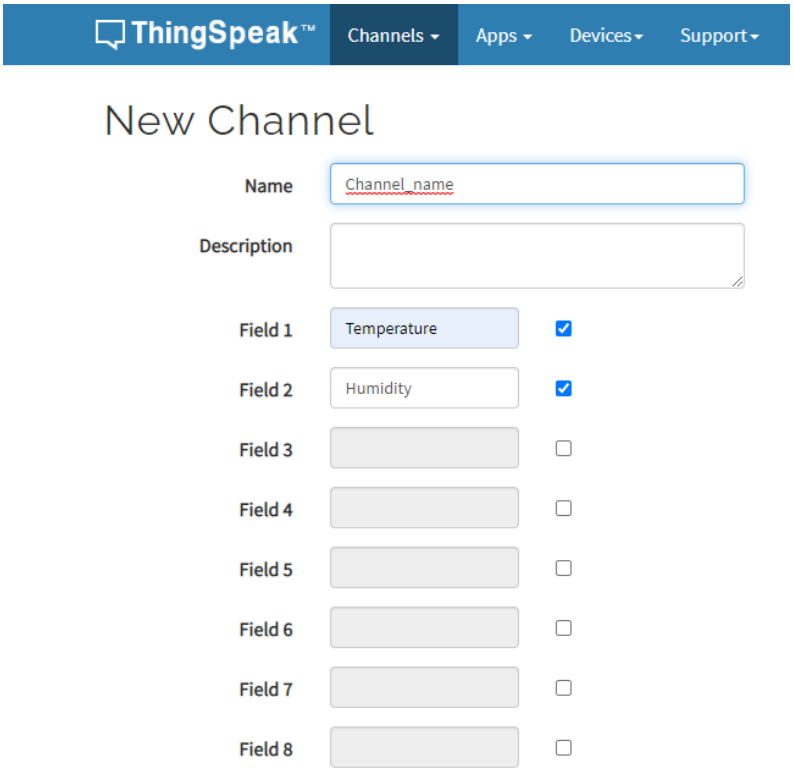
Procedure:

1. Create account on ThingSpeak cloud using official college mail I'd.
2. Setting up Arduino IDE to read NodeMCU.
3. Upload data to cloud using NodeMCU and DHT11.

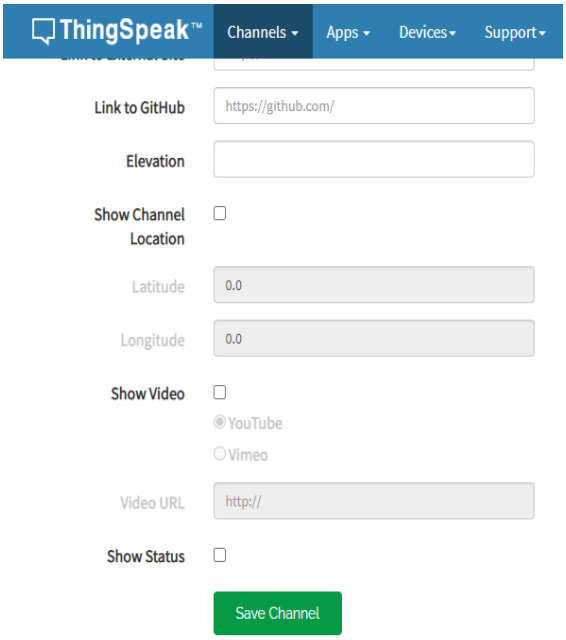
Creating account on ThingSpeak cloud:

Step1: Go to https://thingspeak.com/ and create account using official college mail I'd.

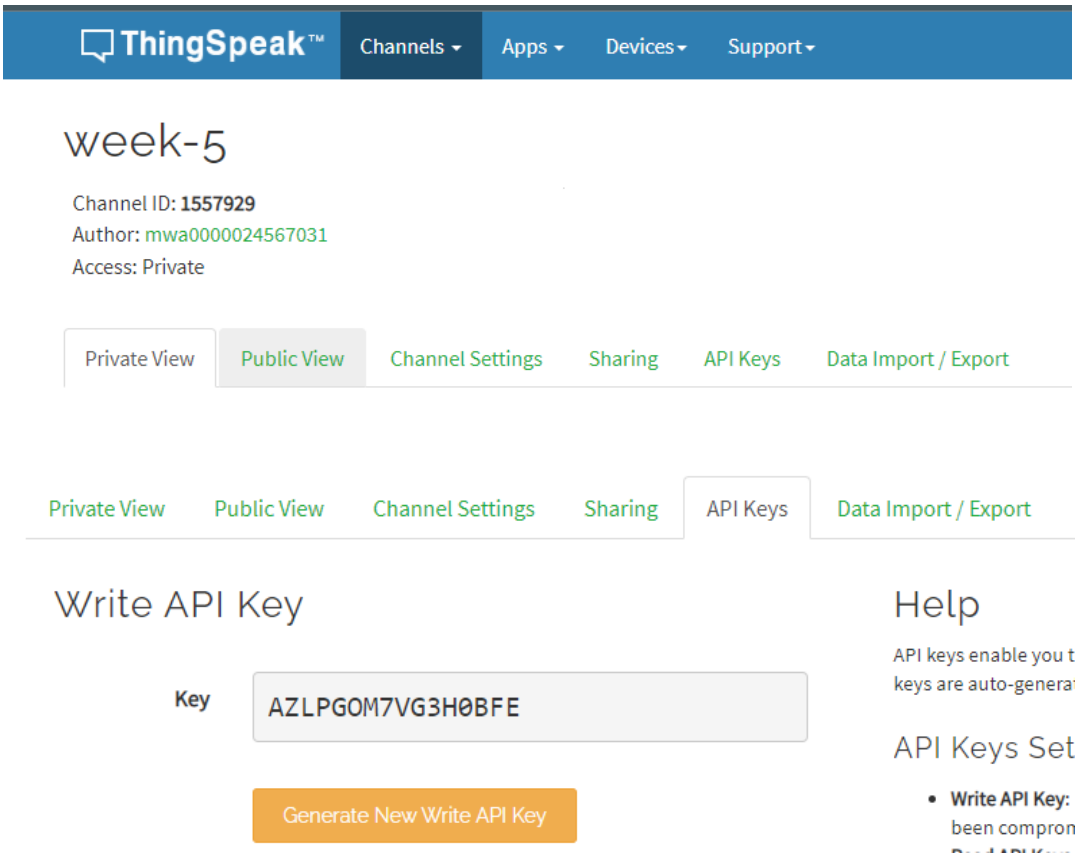
Step2: Create a new channel and add two fields namely “Temperature” and “Humidity”.



Step3: Scroll down and click “save”.



Step4: Make a note of Channel number and API Key.



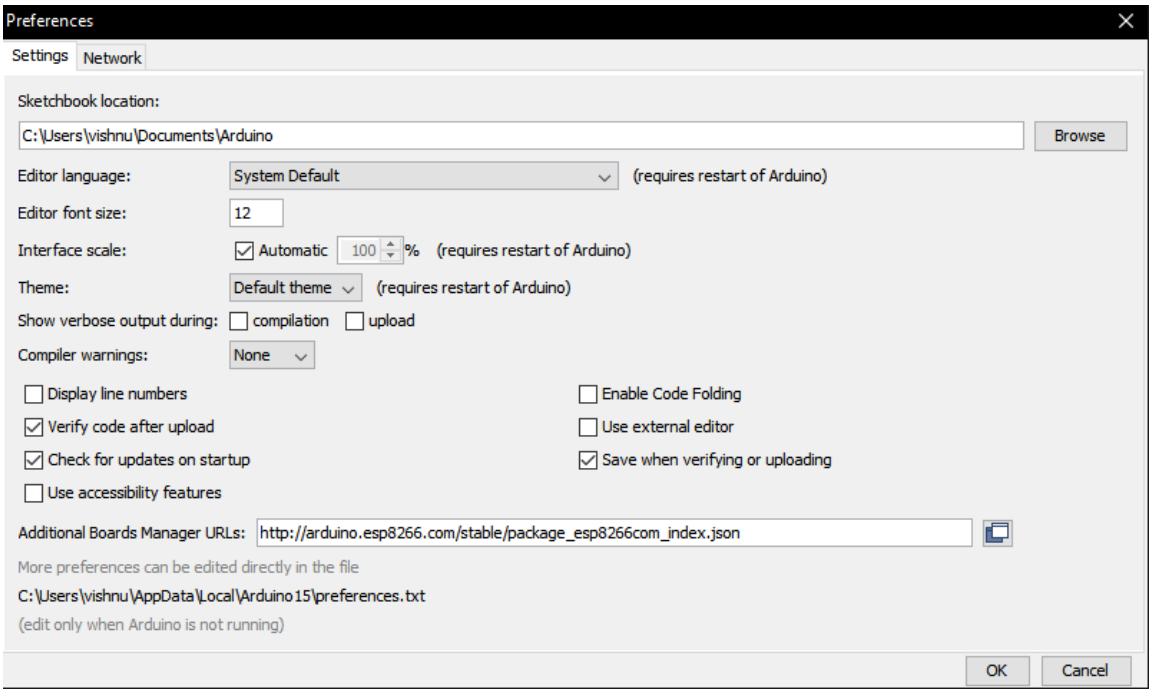
Setting Arduino IDE to read NodeMCU:

Step1: Open Arduino IDE.

Step2: Open file->Preferences.

Step3: In the Additional Board Managers URL field enter the below link

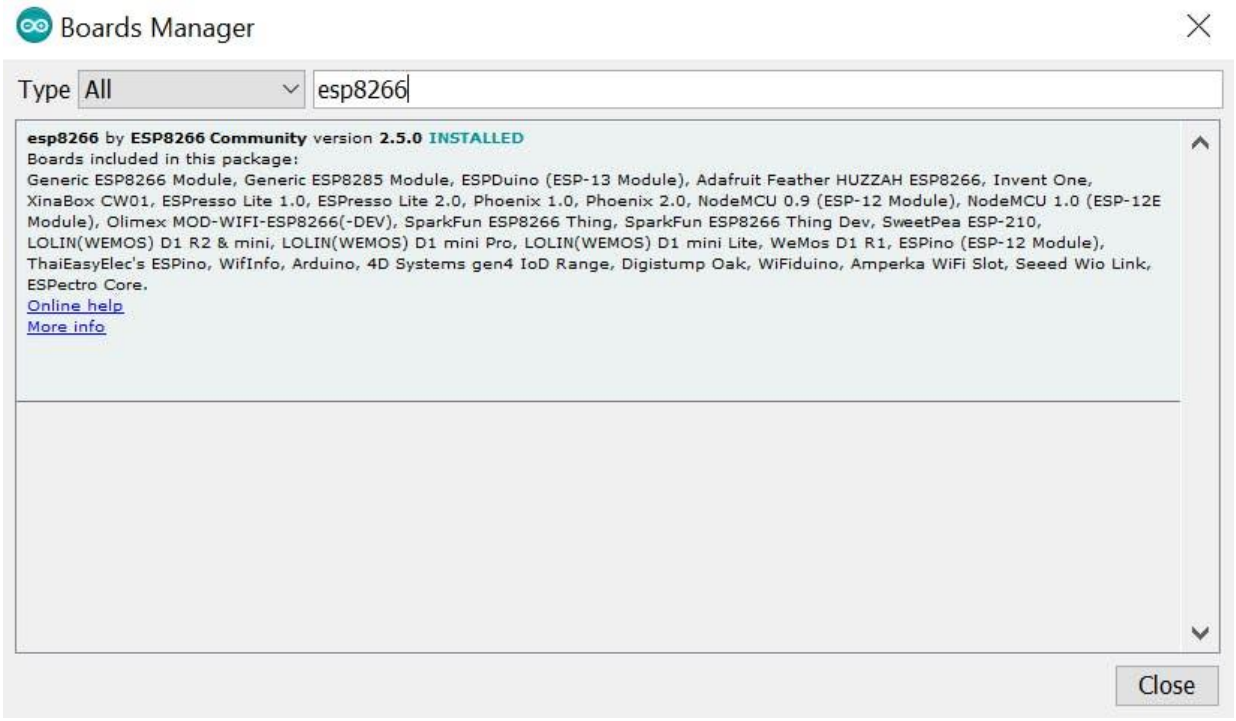
http://arduino.esp8266.com/stable/package_esp8266com_index.json



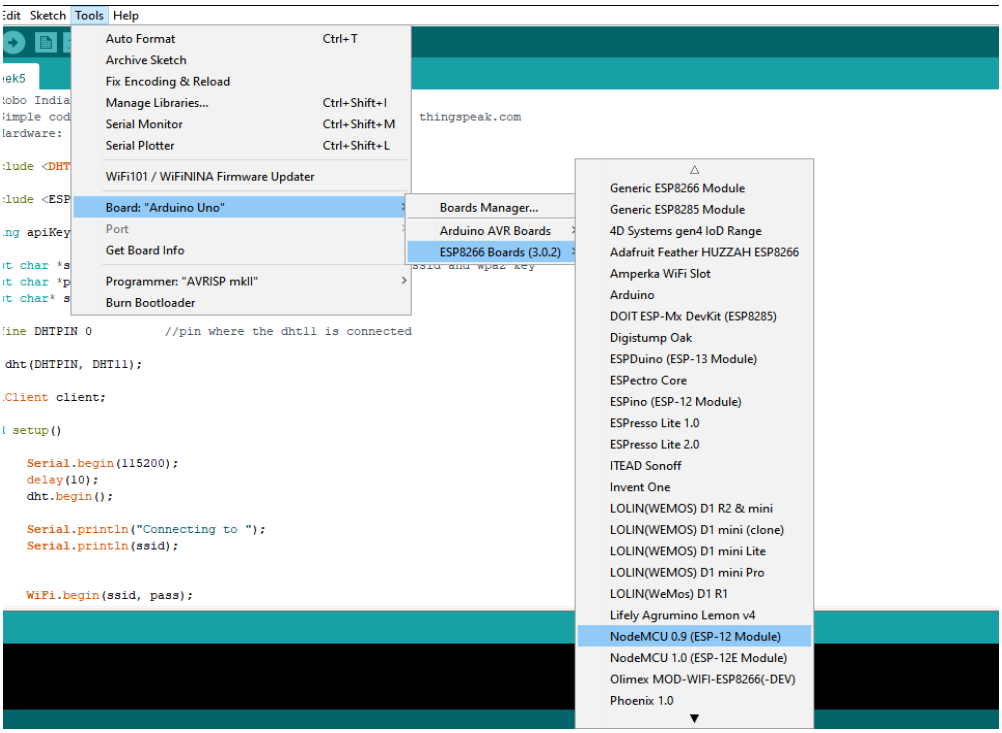
Click Ok .

Step4:

Search the word ESP8266 in Boards>boards manager from Tools menu. Then install ESP8266 boards. After complete installation, you will see the INSTALLED label on ESP8266 boards.



Step5: Go to tools and Select the board and we can see ESP8266 and select NodeMCU 0.9 from the list.



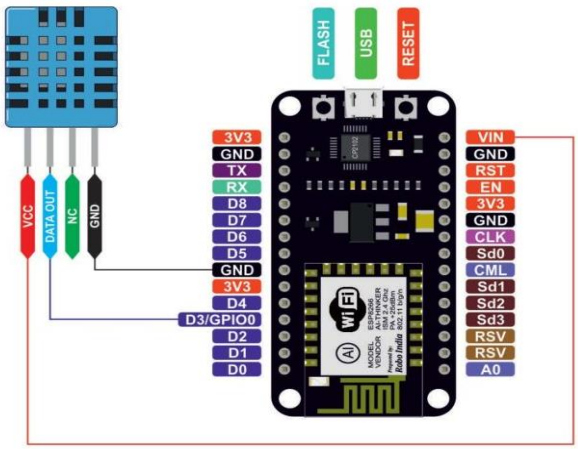
Uploading data to cloud:

Write the code in Arduino IDE and Verify and upload the code.

The Code gets uploaded to cloud this takes some time.

The output graphs can be seen in the channel created on ThingSpeak Cloud.

<u>Pin Diagram:</u>

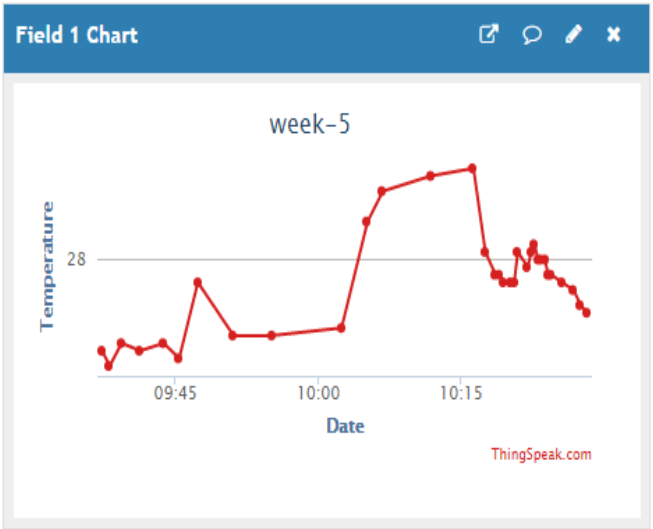


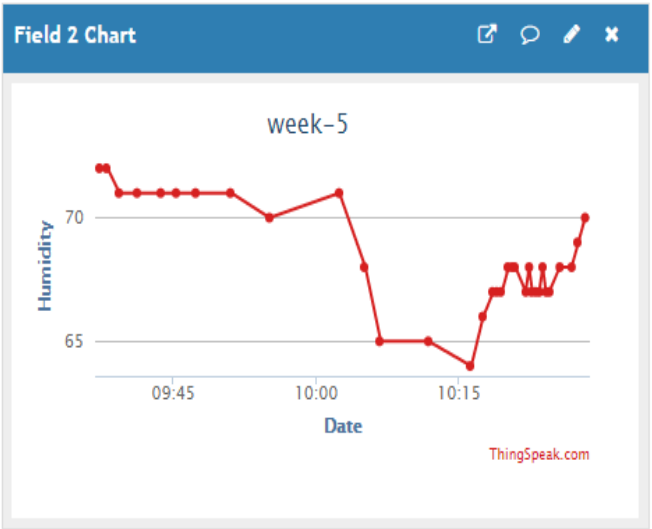
Source Code:

```
#include <DHT.h>  // Including library for dht
#include <ESP8266WiFi.h>

String apiKey = "AZLPGOM7VG3H0BFE";     //  Enter your Write API key from ThingSpeak
const char *ssid=  "wifi network name ";     // replace with your wifissid and wpa2 key
const char *pass =  "password of wifi network";
const char* server = "api.thingspeak.com";

#define DHTPIN 0          //pin where the dht11 is connected

DHT dht(DHTPIN, DHT11);

WiFiClient client;

void setup()
{
Serial.begin(115200);
delay(10);
dht.begin();

Serial.println("Connecting to ");
Serial.println(ssid);


WiFi.begin(ssid, pass);

     while (WiFi.status() != WL_CONNECTED)
    {
delay(500);
Serial.print(".");
    }
Serial.println("");
Serial.println("WiFi connected");

}

void loop()
{
     float h = dht.readHumidity();
     float t = dht.readTemperature();
           if (isnan(h) || isnan(t))
              {
Serial.println("Failed to read from DHT sensor!");
                   return;
              }
```

```
    if (client.connect(server,80))   //   "184.106.153.149" or api.thingspeak.com
                {

                        String postStr = apiKey;
postStr +="&field1=";
postStr += String(t);
postStr +="&field2=";
postStr += String(h);
postStr += "\r\n\r\n";

client.print("POST /update HTTP/1.1\n");
client.print("Host: api.thingspeak.com\n");
client.print("Connection: close\n");
client.print("X-THINGSPEAKAPIKEY: "+apiKey+"\n");
client.print("Content-Type: application/x-www-form-urlencoded\n");
client.print("Content-Length: ");
client.print(postStr.length());
client.print("\n\n");
client.print(postStr);

Serial.print("Temperature: ");
Serial.print(t);
Serial.print(" degrees Celcius, Humidity: ");
Serial.print(h);
Serial.println("%. Send to Thingspeak.");
                    }
client.stop();

Serial.println("Waiting...");

  // thingspeak needs minimum 15 sec delay between updates, i've set it to 30 seconds
delay(10000);
}
```

OUTPUT:





Viva Questions:

1. What is use of ESP8266 ?

2. How many pins are present in the ESP8266 wifi module?

3. What is the use of TX pin in ESP8266?

4. What is the use of ThingSpeak Cloud?

5. How to submit data to the ThingSpeak Cloud?