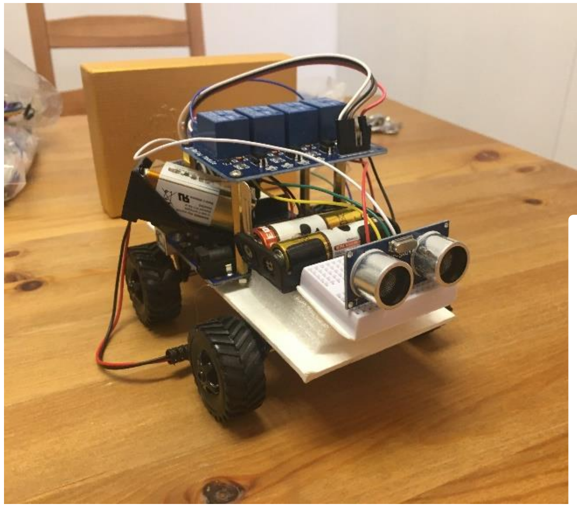
香港教育大學
The Education University
of Hong Kong



FACULTY OF LIBERAL ARTS AND SOCIAL SCIENCES
博文及社會科學學院

科學與環境學系
Department of Science
and Environmental Studies

ARDUINO WORKSHOP

Dr YEUNG Chi Ho Bill

The Department of Science and Environmental Studies
The Education University of Hong Kong

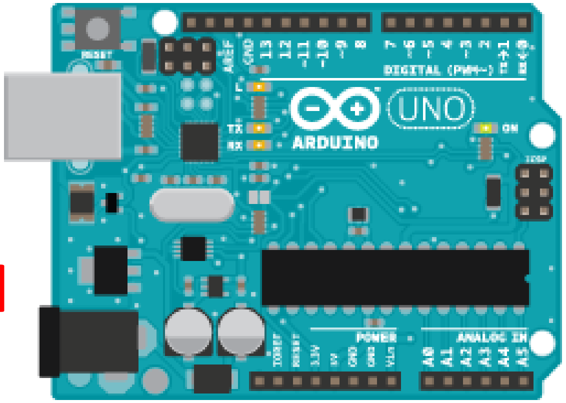
University-School Support Program

Promoting STEM Education using Self-directed Learning as Strategy

Workshop rundown

- **Introduction to Arduino**
- **Operation of Arduino**
- **Part A** – C++ syntax for Arduino coding
- **Part B** – Switch on an LED with simple circuit and coding
  以簡單電路及編碼開啓LED
- **Part C** – Sensors for Arduino, Arduino 感測器
- **Part D** – Controlling motors by Arduino, 以 Arduino 控制馬達
- **Part E** – Arduino as controller by coding with “if”
  以Arduino及”if”編碼作控制器
- **Part F** – Building Smart Devices for the Elderly
  設計長者智能裝置
- **Part G** – Liquid Crystal Display (LCD) as output
  以液晶體顯示屏輸出數據

Arduino microcontroller (微控制器)



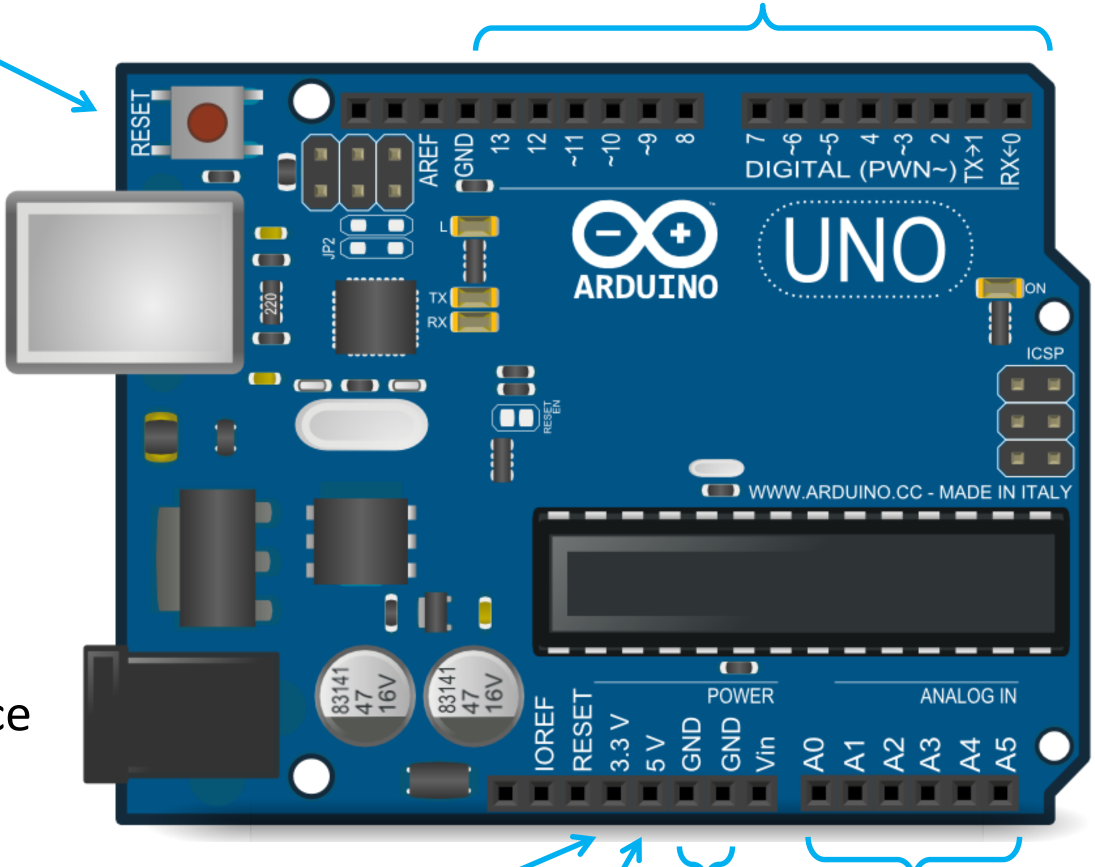
- An **open-source** (開放原始碼) electronics platform based on easy-to-use hardware and software
- A (i) **programmable**, and (ii) **single-board microcontroller** which

1. **reads** inputs, e.g. sensor readings, button states,
2. **processes** the inputs based on instructions, e.g. check with codes and conditions
3. **produces** outputs, e.g. to turn on an LED, activate a motor

- Connection to different hardware - by simple **wiring** (電路)
- Instruction implementation - by the **Arduino Software IDE (**Arduino IDE 軟件**)** and simple programming language

Arduino microcontroller

- The structure on the “**Arduino Uno**” board:



14 digital input and output pins, i.e. high or low

Reset button

USB receptacle

Power source

3V power

5V power

Ground

6 analog inputs
e.g. reading voltage

Photo credit:

Part A - C++ syntax for Arduino Coding

- Arduino IDE software uses **C++** as the coding language
- Nevertheless, there are some functions, e.g. **setup(), loop(),** which are introduced in Arduino IDE but not in other C++ applications
- **Different variable type** in C++:

syntax	Variable types
int	Integer variables e.g. -1, 0, 1
float	Point type variables
double	Point types variables (double precision)
bool	boolean variables (true or false, 1 or 0)
void	a type of function without “return;”

Arduino program structure:

```
void setup() {
        ....
}
```

(initialization)

```
void loop() {
        ....
}
```

(repeating procedures)

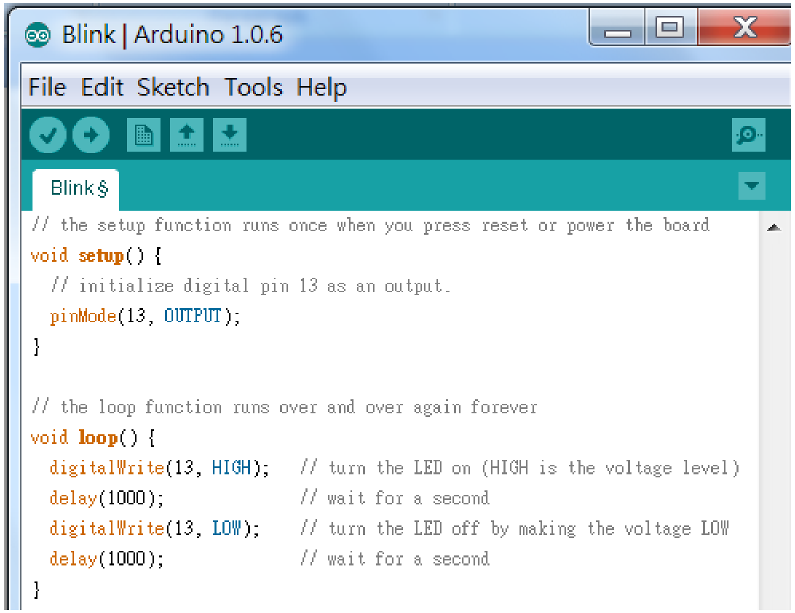
Example to **set up new variables**:

```
int count;
count = 0;
```

Part B - Switching on an LED (開啓LED)

1. Connect the Arduino board to a computer by a USB cable
2. Open “**Device manager**” and **check the port** of the Arduino
3. Open the **Arduino software (IDE)** in “Tools” then “Serial port”, and select the correct port
4. In “Tools” and then “Board”, select the correct board (we are now using “Arduino Uno”)
5. Go to “File” → “Examples” → “Basics” → “Blink” to open examplar file “blink.ino”

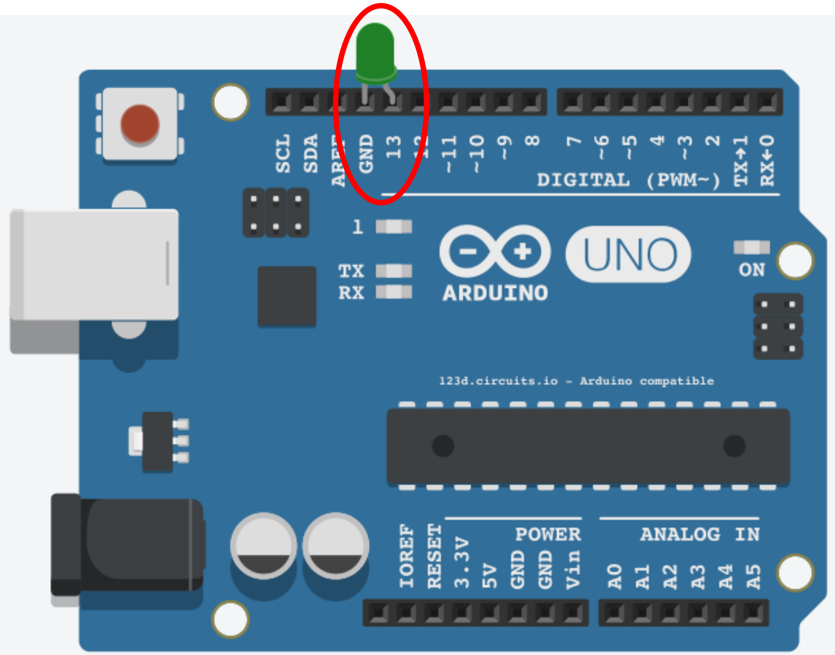
```
void setup() {
        pinMode(13, OUTPUT);
}
void loop() {
        digitalWrite(13, HIGH);
        delay(1000);
        digitalWrite(13, LOW);
        delay(1000);
}
```



```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin 13 as an output.
  pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(13, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);              // wait for a second
  digitalWrite(13, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);              // wait for a second
}
```

Switching an LED (2)

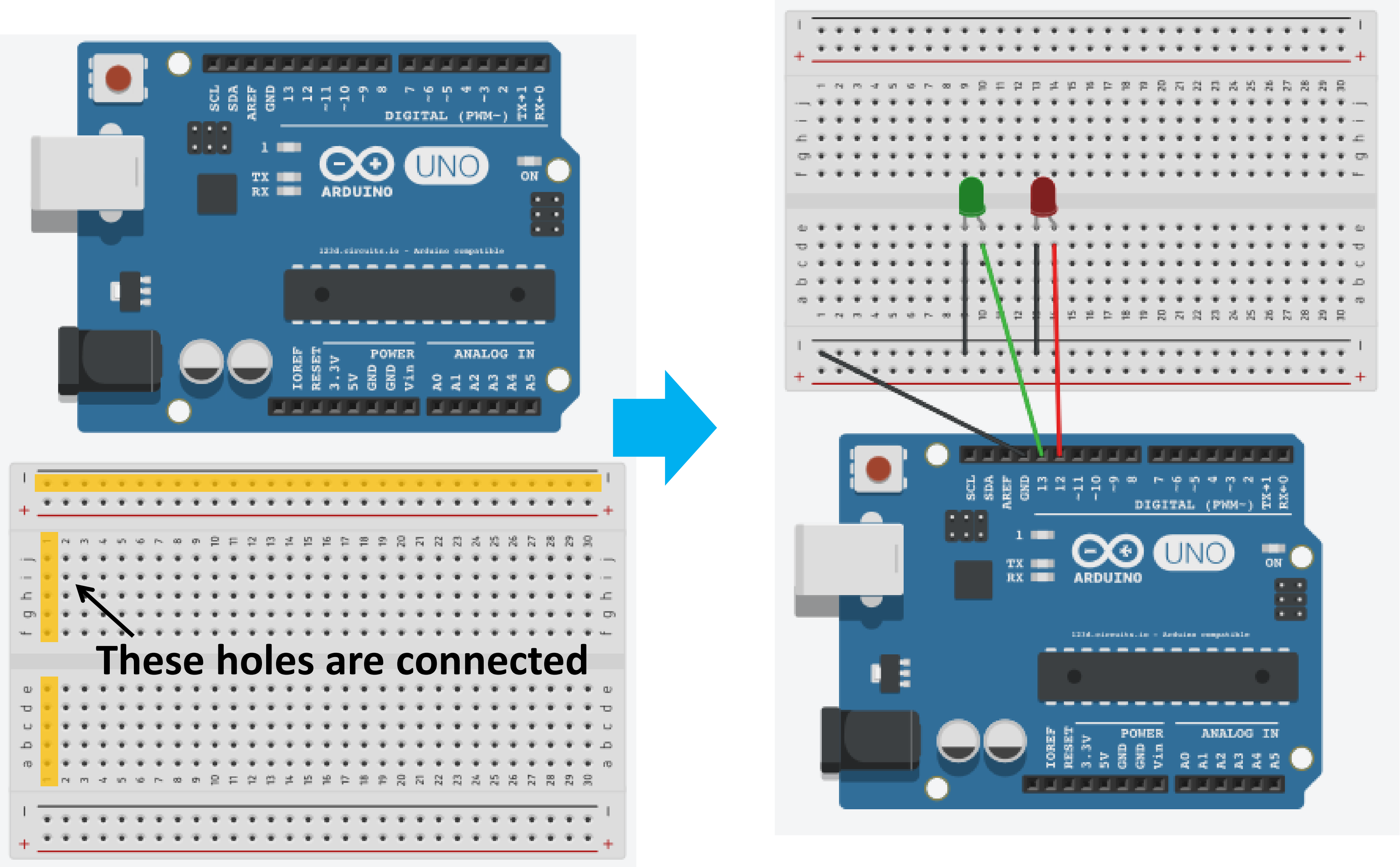
6. Click the **tick** ✓ , the program is checked (compiled)
7. If there is no problem, click the **arrow** ➔ , the program is uploaded to the broad
8. What do you observe for the on-board orange LED?
9. Connect the positive and the negative terminal of a **Green LED** to pin 13 and the ground pin respectively, what do you observe?



Exercise

1. Extend the on-state of the Green LED light to 3s
2. Construct a simple system which simulate the traffic light for pedestrian

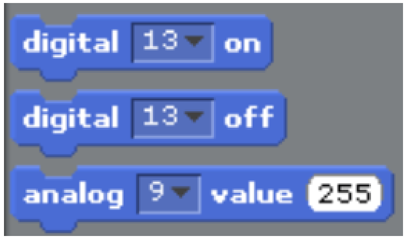
Breadboard (麵包板)



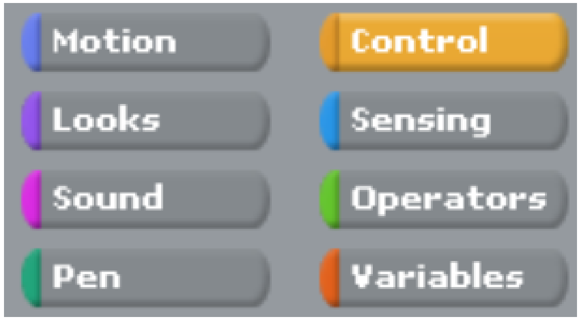
These holes are connected

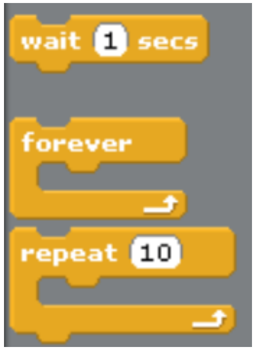
S4A and ArduBlock

- **S4A** is a platform modified from **Scratch** - a graphical programming language developed by MIT which is suitable for youngsters to learn programming
- Another other common block-programming software for Arduino is called **ArduBlock**
- Although script coding is most fundamental, many codes can also be written in S4A and ArduBlock

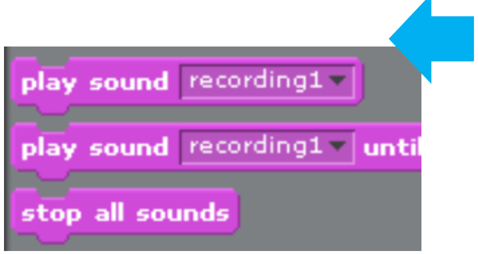


digital 13 on
digital 13 off
analog 9 value 255

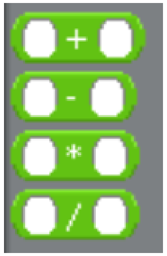




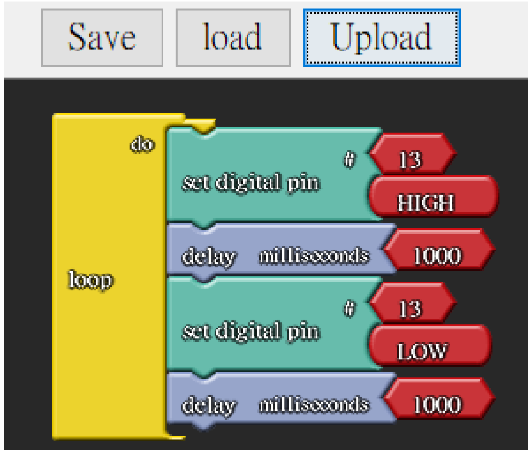
wait 1 secs
forever
repeat 10



play sound recording1
play sound recording1 until
stop all sounds



To upload codes to Arduino, one can use ArduBlock

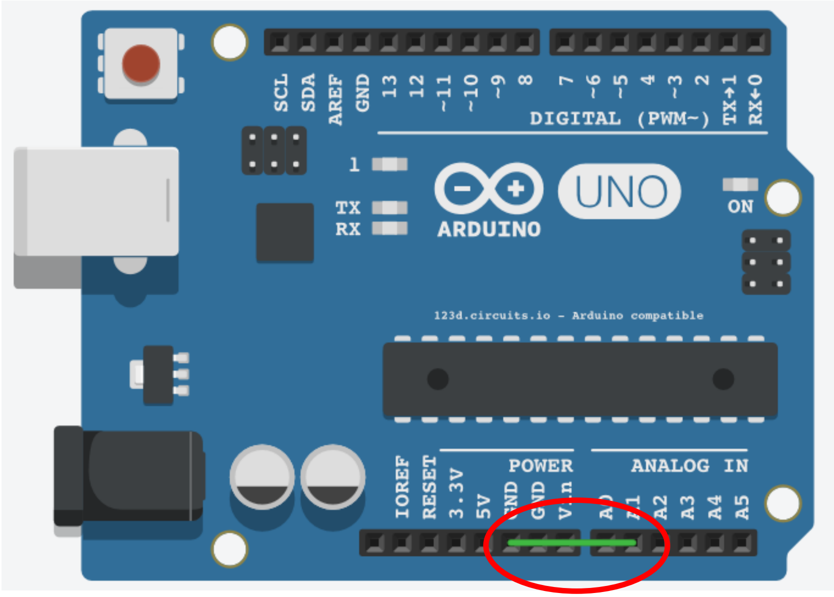


loop do
set digital pin # 13 HIGH
delay milliseconds 1000
set digital pin # 13 LOW
delay milliseconds 1000

Reading data (輸出數據)

- Next, we will use the following code to read data from the Arduino

```
void setup(){
  Serial.begin(9600);
}

void loop(){
   double reading = analogRead(1);
   Serial.print(reading);
   Serial.print("\n");
   delay(2000);
}
```

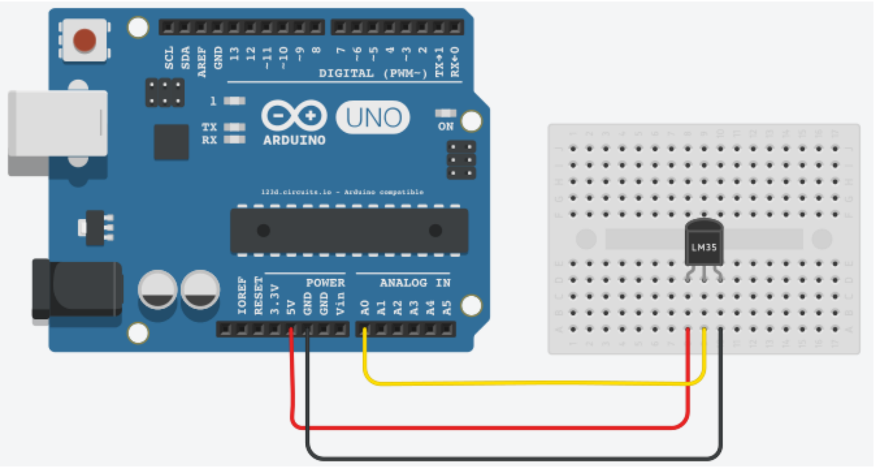


Analog input is a high impedance input, almost draw no current

- **Connect analog pin 1 to the GND pin**, as shown below
- Open “Tools” → “Serial monitor”; observation: ____
- How about **connecting analog pin 1 to the 5V pin**? _______

Part C – Sensor 1: Temperature by LM35

- To measure **temperature,** we can use sensor **LM35**
- Connect the circuit on the left:



- This code displays the sensed temperature in the serial monitor (“Tools”→“Serial Monitor”)

```
void setup(){
                pinMode(0, INPUT);
                Serial.begin(9600);
}
void loop(){
                float temperature =  (analogRead(0)*500.0) / 1024;
                Serial.println(temperature);
                delay(1000);            // output data every second (i.e. 1000ms)
}
```

Sensor 2: Temperature, Humidity by DHT 11

- To measure **temperature and humidity**, we can use **DHT11**
- To use DHT11 sensor, a "**DHT11 library**" has to be installed
- A "**library**" usually include one **.cpp** and one **.h** (header) file, which are useful functions of the sensor
- **Example**: (1) download dht11.cpp and dht11.h from the hyperlink below, (2) go to "My document/Arduino/libraries" (3) create a folder "dht11" and put the .cpp and .h files into the folder

DHT11 library:
http://playground.arduino.cc/Main/DHT11Lib

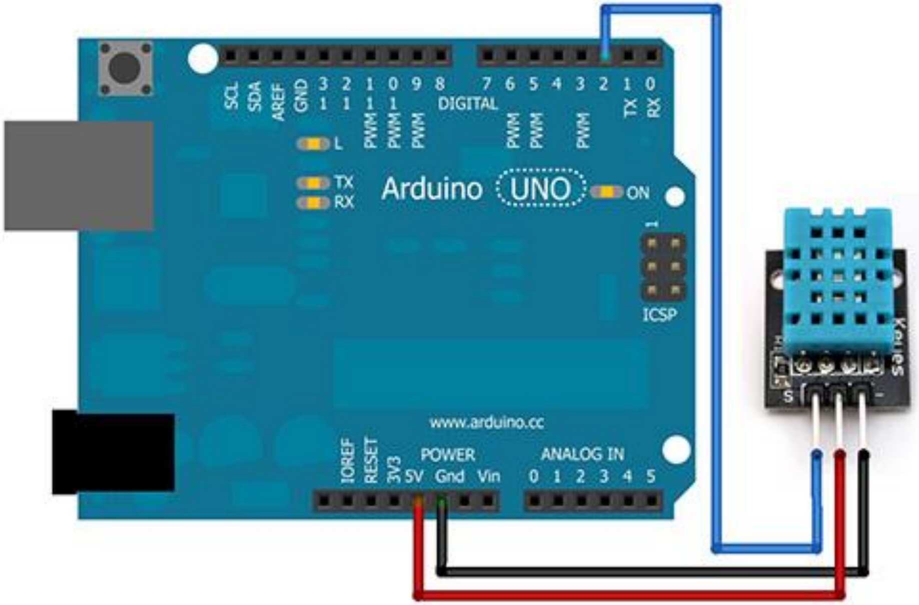


Figure: chioszrobots.com

Temperature and humidity (2)

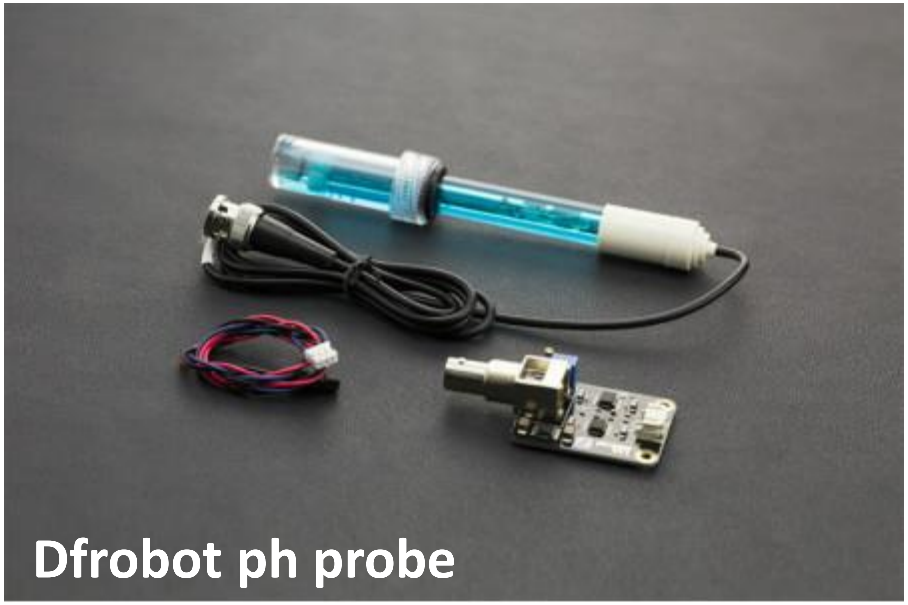
- The following code measures **temperature and humidity by DHT11**

```
#include <dht11.h>
dht11 DHT11;

void setup(){
  Serial.begin(9600);
  DHT11.read(2);  // input at pin 2
}

void loop(){
  Serial.print("Humidity (%): ");
  Serial.print((float)DHT11.humidity, 2);  // to 2 decimal places

  Serial.print("\t Temperature (Deg.): ");
  Serial.println((float)DHT11.temperature, 2);
  delay(2000);
}
```

STEM Challenges: Building a Ph meter

- We will use a **dfrobot ph probe, Arduino**, and **linear equations** in mathematics to construct a Ph meter
- The dfrobot ph probe produces **a roughly linear response** of output voltage as a function of the ph value of the solution
- Connect the ph probe to the Arduino board as follow:

GND → GND, VCC → 5V, Signal → any analog input



Dfrobot ph probe

Calibration

- The code below output the reading of the ph probe on the serial monitor at every 2s interval:

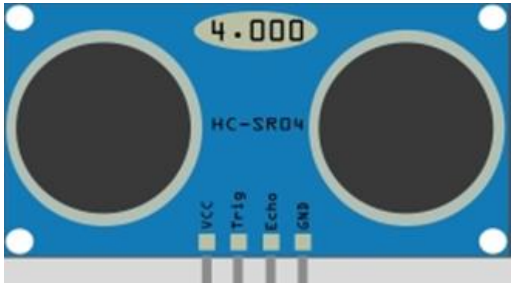
```
int tempin = 3;
void setup() {
  Serial.begin(9600);
  pinMode(tempin, INPUT);
}

void loop() {
  Serial.println(analogRead(tempin));
  delay(2000);
}
```

Challenges

1. Use two solutions with known ph values to calibrate the probe, and convert the reading of the probe into ph values (the formula of **linear equation** has to be applied)
2. Use the probe to measure the ph value of an unknown solution

Sensor 3 - Measuring distance (1)

- To measure **distance**, we can use **ultrasonic sensor HC-SR04 (超聲波距離感測器)**, which consists of one emitter and one receiver of ultrasound
- The **4 pins** of HC-SR04 are as follows:
- Connect the following circuit:



GND – ground pin

Echo – output the signal received at reception

Trigger – trigger the pulse emission

VCC – voltage supply

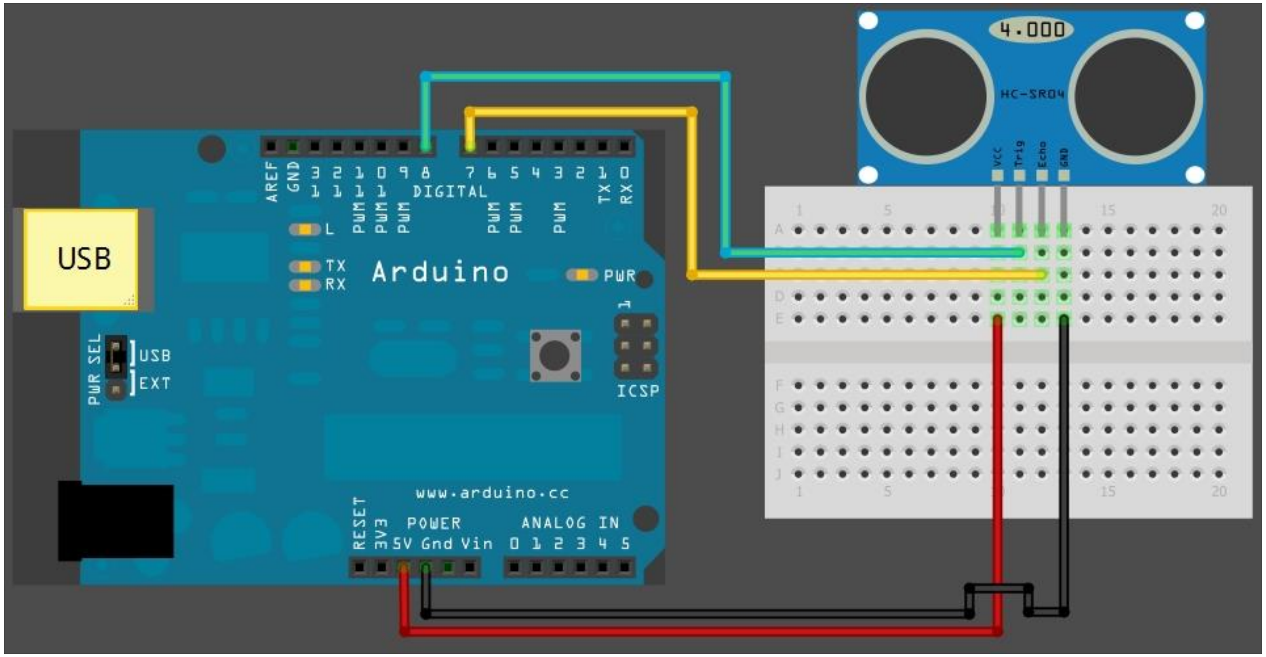


Figure: http://arduinobasics.blogspot.hk/2012/11/arduinobasics-hc-sr04-ultrasonic-sensor.html

Measuring distance (2)

- The following codes **measure distance** of an object by SR04:

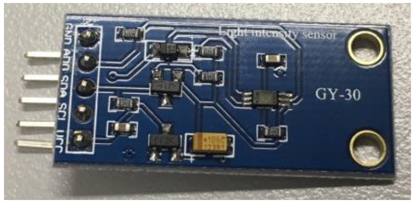
```
int trigPin = 8;
int echoPin = 7;
long duration, distance;

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

```
void loop() {
digitalWrite(trigPin, LOW);
 delayMicroseconds(2);

 digitalWrite(trigPin, HIGH);
 delayMicroseconds(10);

 digitalWrite(trigPin, LOW);
 duration = pulseIn(echoPin, HIGH);

 //Calculate the distance (in cm) based on sound speed
 distance = duration/58.2;
 Serial.println(distance);

 //Delay 0.5s before next reading.
 delay(500);
}
```

Exercise: Output the speed of a moving object using SR04

Sensor 4 – Light intensity sensor

- To measure **light intensity**, we can use **GY30**, which measures light intensity in **LUX level**



- To use GY30, the library **BH1750.h** is required
- GY30 uses **I^2C communication mode** on Arduino, where the following connection has to be made

GND → GND, VCC → 5V
SDA (dataline), SDL (clockline):

Board	**I2C / TWI pins**
Uno, Ethernet	SDA → A4 SCL → A5
Mega2560	SDA → A20 SCL → A21
Leonardo	SDA → A2 SCL → A3

This codes output the LUX level sensed by GY30:

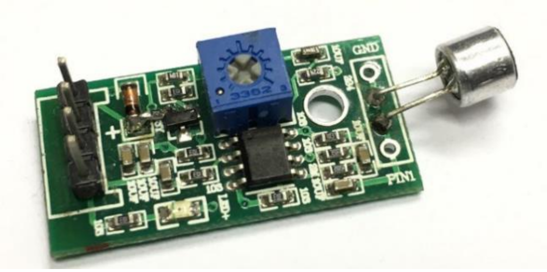
```
#include <Wire.h>
#include <BH1750.h>
BH1750 lightMeter;

void setup(){
  Serial.begin(9600);
  lightMeter.begin();
}

void loop() {
  uint16_t lux = lightMeter.readLightLevel();
  Serial.println(lux);
  delay(1000);
}
```

Other sensors

- **Motion sensor** (by passive infrared) – output “1” when there is a person nearby, and otherwise “0”:

```
void setup(){            // connect output to D10
        pinMode(10, INPUT);
        Serial.begin(9600);
}
void loop(){
        Serial.println((digitalRead(10));
        delay(1000);
}
```

- **Sound sensor** – output either a value of sound level, or output “1” when the sound level is larger than a threshold:

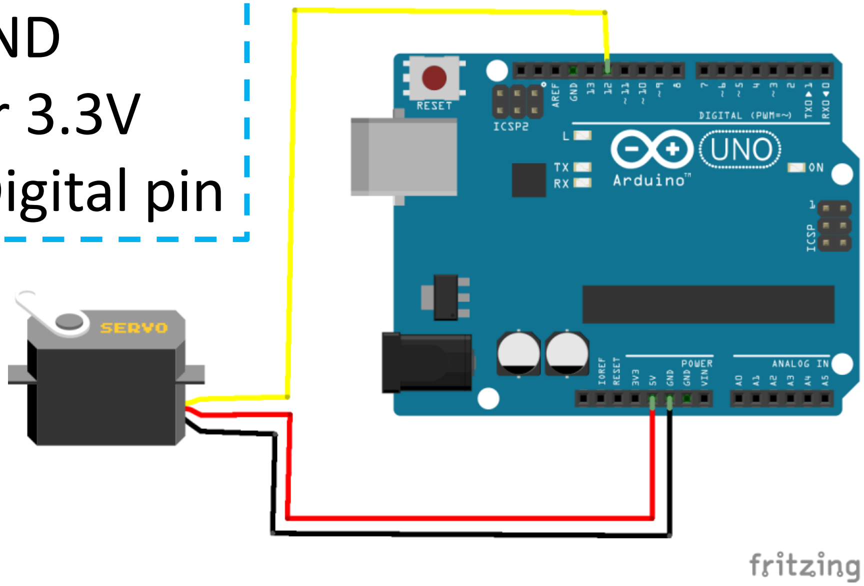


```
void setup(){            // connect output to A0
        pinMode(0, INPUT);
        Serial.begin(9600);
}
void loop(){
        Serial.println((analogRead(0));
        delay(1000);
}
```

Part D - Controlling Servomotors

- **Servomotors** are devices which can rotate to an **precise angle** according to an input signal
- To connect the servomotor to the Arduino board, connect the following circuit:

Brown → GND
Red → 5V or 3.3V
Orange → Digital pin



- To control servomotor, one can download and install the library "**Servo.h**"

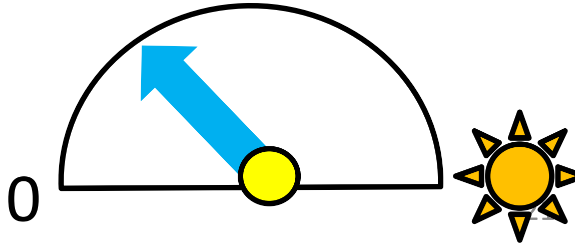
Controlling servomotors (2)

- Use these codes to **control the servomotor** and observe:

```
#include<Servo.h>
Servo servomotor1;
void setup(){           // connect the servomotor to D12
            servomotor1.attach(12);
}
void loop(){
            servomotor1.write(0);
            delay(2000);
            servomotor1.write(90);
            delay(2000);
            servomotor1.write(180);
            delay(2000);
}
```

Exercise

1. Simulate **an automatic door** which opens when it senses a person approaches
2. Construct a device to **display light intensity level** with a rotary indicator



Part E – Arduino as controller by coding with “if” 以Arduino及”if”編碼作控制器

- **Our goal**: a device which alerts a too low or a too high temperature around the classroom

- **Suggestion**: a device which

1. switches on a **green light** when temperature T < 24°
2. switches on a **red light** when temperature T > 30°

- These devices can be distributed around the classroom
- You can have **your own design**!

Build your own device!

- Code for conditional actions:

```
If(temperature < 24)
   Your action;
```

Part F - Smart Devices for the Elderly

- Use "**if … then …** " and various **sensors** and **actuators** to build the following smart devices for the elderly:

1. **Smart cane -** alerts the elderly when the cane is close an obstacle

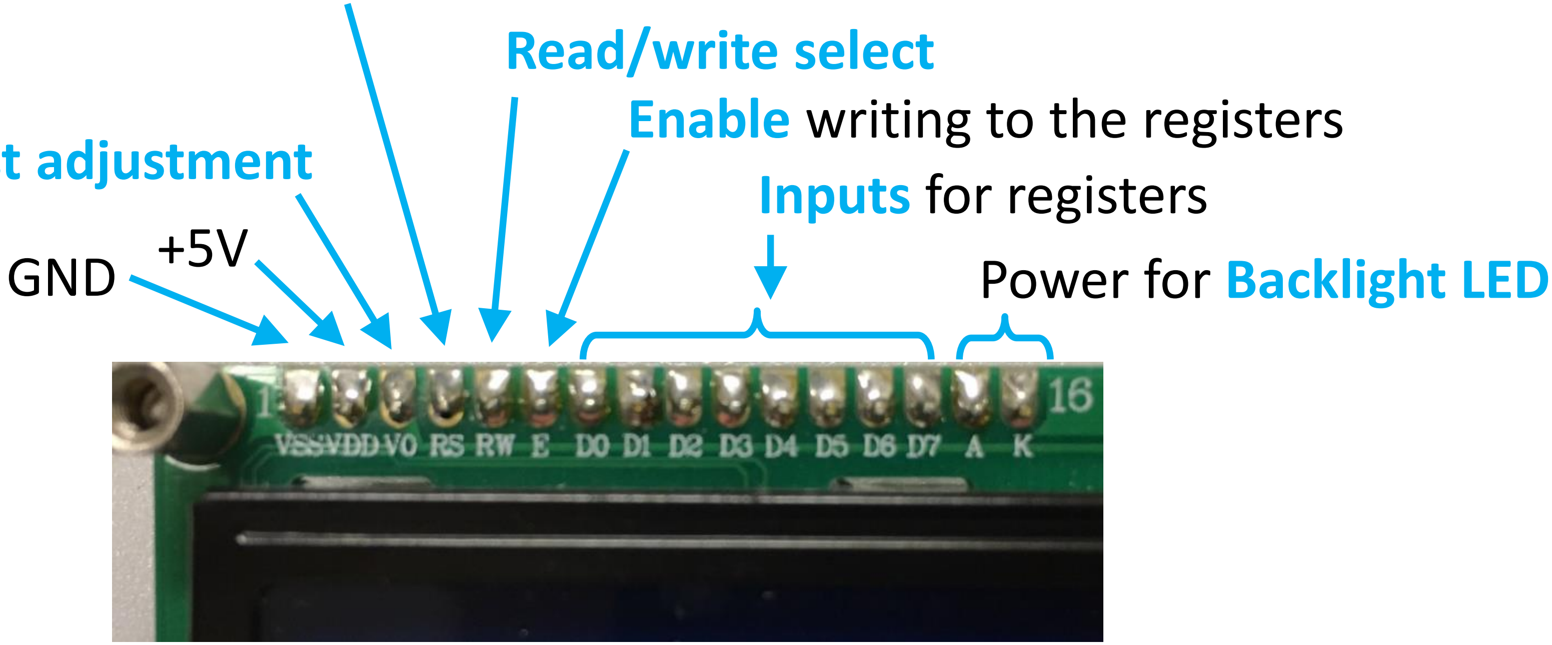
2. **Smart medicine box -** alerts the elderly and open automatically when it is the time for taking medicine

3. **Passenger counters** – counting the number of passengers on the upper deck

Part E - Liquid crystal display (液晶體顯示屏)

- After reading different physical quantities, we will learn to display them using a 1602A **liquid crystal display (LCD)**
- The screen has **2 rows**, each row can display **16 characters**, and there are 16 input pins

Register select (RS) – which can select between data registers (for screen display), or instruction register, which LCD follows instructions



Read/write select

Enable writing to the registers

Contrast adjustment

Inputs for registers

+5V

GND

Power for **Backlight LED**

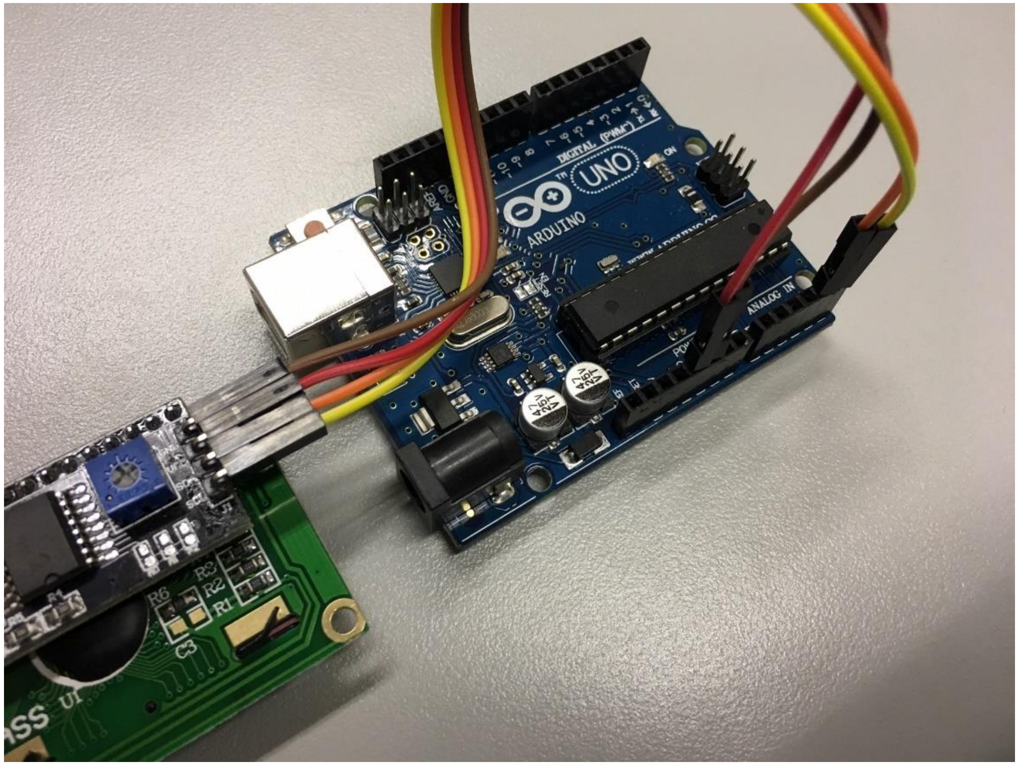
Reference: https://www.arduino.cc/en/Tutorial/HelloWorld

Liquid crystal display (2) & I^{2}C

- We will use a **communication mode** called **I^{2}C** to communicate with the LCD display; library "**Wire.h**" has to be called
- Connect the circuit below:

GND → GND
VCC → 5V

SDA (dataline):
SDL (clockline):



Board	**I2C / TWI pins**
Uno, Ethernet	SDA → A4 SCL → A5
Mega2560	SDA → A20 SCL → A21
Leonardo	SDA → A2 SCL → A3

Reference: https://www.arduino.cc/en/Reference/Wire

Liquid crystal display (3)

- The following codes display some text on the LCD display

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
// Refence: https://bitbucket.org/fmalpartida/new-liquidcrystal/downloads
LiquidCrystal_I2C lcd(0x3F, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);  // Try either 0x27 or 0x3F

void setup() {
 lcd.begin(16, 2)     ;              // set up the LCD's number of columns and rows
 lcd.print("hello, world!");      // Print a message to the LCD.
}

void loop() {
 lcd.setCursor(0, 1);                // set the cursor to column 0, line 1
 lcd.print(millis() / 1000);       // print the number of seconds since reset:
}
```

Exercise

- Measure the temperature by LM35 and display the temperature on the screen every 1s (NOTE: lcd.print((char)223) prints the symbol °)

Summary

- We have introduced

1. The **hardware** structure of the Arduino board
2. The use of **Arduino software IDE** to incorporate simple computer programs onto the Arduino board

- Examples **Arduino-based activities** – automated traffic light, smart devices, etc.
- Several different **sensors** are introduced:

1. **LM35** – temperature sensor
2. **DHT11** – temperature and humidity sensor
3. **HC-SR04** – ultrasound distance sensor
4. **GY30** – light intensity sensor

- Others: **display data on LCD**, **relay, remote controller**

Q&A

Thank
you!!