Arduino程式設計

（第一冊）

洪國勝、蔡懷文、王丹君　編著

國立高雄師範大學工業科技教育系
溫嘉榮教授推薦

目錄

第一冊

第二冊

序言

因應教育部科技領域課綱「科技領域之教學，實作活動時數宜占整體課程時數的二分之一至三分之二」，翻閱目前中學生「生活科技」課本，課綱內容非常精彩且豐富，包含科技的本質、機構的繪圖、設計與製作、材料的選擇與加工、木工機具的保養與操作、能源與發電動力機械的原理與操作、電子電路的原理與操作、新興科技的發展與操作等。木工與手工具還停留在 40 年前工藝課的操作，當時我們的工藝課可以製作木工、鐵管小椅子、書架，拿來當作學校週會或家裡用。但現在環境不同了，塑膠一體成形的家具很便宜，所以已經不流行 DIY，連帶家庭常備的木工、鐵工的小機具也慢慢消失；能源與動力設備則太大，無法在每個學校購置；電子電路原是不錯的科技實作選項，但目前已經被單晶控制取代。以筆者資訊教學 35 年的經驗而言，本人大力推廣機電整合的 Arduino 作爲生活科技的實作課程。因爲 Arduino 所占空間最小，設備費最便宜，最適合全班教學。可完成的生活科技產品如字幕機、告白板、叫號器、電子鐘、計時器、溫度計、紅綠燈、霹靂燈、遙控器、電子琴等，都與生活息息相關。做完的產品可以帶回家用，可說是非常眞實的生活科技之產品；且本項操作沒有觸電、刀具運轉等危險性，沒有粉塵、也沒有噪音，又可體驗與學習程式設計等運算思維。其次，我們也開發上課用教具，而使用教具教學的優點如下：

1. 教具可放在學校工場重複使用，這樣老師不用向學生收錢，學生也不會因爲忘了帶材料而影響進度，學生依序探索後，可依自己的興趣，自己購置材料，完成所需專題。
2. Arduino 具有自我保護電路，學生插錯也不會壞。
3. 操作電路電壓僅爲 5V，任意觸摸、接錯電路等都不會有任何危險。

筆者雖然程式教學與著作超過 35 年，快樂與熱情不變。儘管不斷的重複讀稿，力求完善，但仍難免有疏漏及錯誤，尚祈各位先進不吝指正，將不當或錯誤詞句回傳給我，如下表，本人不勝感激。每年會選出與抽出精彩回函，寄發紀念品。本書感謝高師大工教系學妹王丹君同學、學弟蔡懷文老師主動加入共筆創作、完稿於蘆洲國中研習時，也感謝王漢卿老師主動幫忙改正很多錯誤，在此致謝。其次，筆者也會在泉勝出版網站（www.goodbooks.com.tw）刊登本書勘誤表。

洪國勝、蔡懷文、王丹君 謹識

www.goodbooks.com.tw

▶ 讀者意見、內文觀念與詞句修正回函表

Arduino 程式設計			
頁數	行數	錯誤或不當詞句	建議修改為
其他批評與建議：			

以上意見、指正回函表，請郵寄至 aa163677@yahoo.com.tw，每年會選出與抽出精彩回函，寄發紀念品。

推薦序

目前國民小學是採用圖形化介面的 Scratch 當作程式設計工具，高中、大學、業界則採用文字化的程式設計介面，如 C/C++ 與 Python。國民中學就有點尷尬，到底要採用哪一種介面？看過洪國勝老師的『Arduino 程式設計』，我也認同國中、高中生都應該使用文字式程式設計，這樣才能快速連結產業需求。

用Arduino學程式

國中與高一學生因爲數學基礎還太少，老師教程式還要先教數學，眞的很費力，所以我也認同使用 Arduino 的 LED 實物來學習迴圈與陣列，這樣更能激發與維持學生學習程式設計熱度。其次，Arduino 是生活科技、資訊科技最好的補充教材。因爲單晶 Arduino 是目前最新科技主流，簡單的電路與程式就可嵌入目前工業與家用電器，進而簡化控制電路與節省開發的成本，如 ABS 防鎖死剎車、字幕機、電子琴、溫度計、遙控器、叫號器、電子時鐘等。

目前校園常用的單晶控制晶片有 Macro:Bit、Arduino，Macro:Bit 比較著重在圖形開發介面，主要是來讓小學生玩簡單的聲光控制遊戲；Arduino 則是文字形式的開發介面，也是眞實工業的嵌入式晶片，可眞實改善許多生活與工業控制，適合中學生以上與社會人士學習嵌入式系統控制。Arduino 之所以能異軍突起，我認爲是因爲它主張開源，且軟硬體都是開放的，使用者可以站在巨人的肩膀，繼續接力開發新產品。其次，Arduino 輸出電流變大、腳位也變多，這樣可以簡化生活電器的四位數七段

與點陣 LED 顯示。既然單晶電路已經簡化、成本也降低，那麼 Arduino 程式設計也就更適合拿來當作國高中的生活科技課程的實作教材，讓國中與高中生活科技也能與世界潮流平行接軌。學生學的都是目前新興科技，而且都是簡單的接線與簡單的程式，就可改善與實作目前生活科技產品，這樣的學習可說非常生動與實用，當然可以激發學生自造更多創新科技產品。

洪國勝老師是我 76 年高師大工教系畢業高徒，國高中雲縣與省賽工藝競賽科展優勝，所以立志讀工教系。大二全校程式設計第一名、大四全國大學含研究所『微電腦應用創作』優勝第一名，畢業分發高雄市立海青工商資訊科任教。退休後教學熱情不變，自力成立『泉勝出版有限公司』，自費推廣國高中的科技與程式教育，詳見泉勝出版網站。每本著作都是洪國勝老師累積超過 40 年實作與教學心得。所以，本人不斷推薦其 C/C++、Python、APCS 與 Arduino 等程式設計著作，也在此『Arduino 程式設計』寫公開序言推薦。

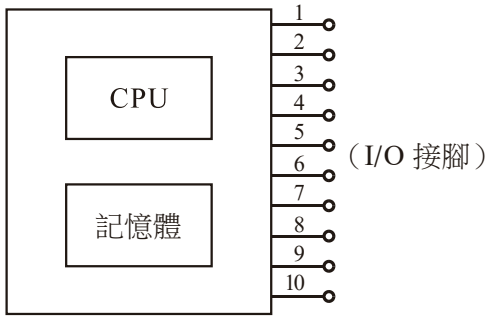
國立高雄師範大學工業科技教育系
溫嘉榮教授
大力推薦
2022/5/12

第1單元

準備工作

單晶片

單晶片是將 CPU、記憶單元整合在一個 IC 裡面，然後預留若干 I/O 接腳如下圖。這些 I/O 接腳可以讓使用者，以軟體程式的方式，指派爲低電位或高電位，進而控制所有負載的 ON 與 OFF。這動作看起來很簡單，但因爲單晶速度快，速度快就能有變化，就協助改善很多共業控制系統。其次，其體積很小，體積小就能嵌入原有的工業產品，所以又稱爲嵌入式控制。例如家電的冷氣、洗衣機、風扇、汽車的防鎖死煞車、胎壓偵測、自動駕駛等都已經使用單晶片協助控制，因爲都是透過軟體簡化硬體設備，這樣當然可縮短產品開發流程與降低硬體設備成本，所以學習單晶片已經是時代趨勢。其次，單晶片又稱單板電腦，因爲一小塊電路板，內部就含 CPU 與記憶單元，其功能就如同小型的電腦，已具有輸出入、決策、迴圈、陣列、輸出入等電腦所具有的功能，方塊示意圖如右：



單晶片模組

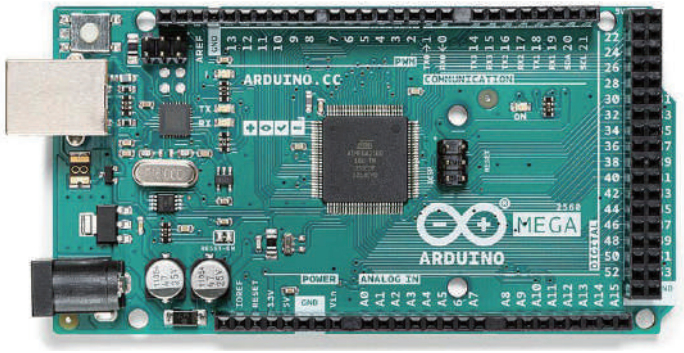
Arduino Maga 2560

目前常聽到的 Macro:Bit、8051、Arduino 等都是單晶片，Macro:Bit 比較著重在圖形開發介面，主要是來讓小學生玩簡單的聲光控制遊戲；8051 與 Arduino 都是使用文字式的開發工具，是眞實的嵌入式工業晶片，可眞實改善許多生活與工業控制，適合中學生以上與社會人士學習嵌入式系統控制。但 8051 已經過時，漸漸被 Arduino 取代，Arduino 官網是 www.arduino.cc。Arduino 之所以能異軍突起，我是認爲它主張開源，它的軟硬體都是開放（甚至使用者用 Arduino 所開發的程式也僅能公開），使用者可以站在巨人的肩膀，繼續接力開發新產品。其次，Arduino 輸出電流變大，可以直接驅動 LED，使的產品的電路也變的簡單。還有，Arduino 腳位也變多，這樣可以直接驅動一些需要重複掃描輸出的元件，例如，本書要介紹的四位數七段顯示器與點陣 LED，與傳統 8051 單晶相較，也變簡單了。Arduino 依照使用者不同的需求，開發很多版本的微控板，其中型號 Mega 2560 的 I/O 腳位共有 70 隻，因爲腳位多、電流也夠大，可直接驅動本書介紹的四位數七段與點陣 LED，這樣用來實作這些生活科技常用電路，其電路與軟體最爲省事，本書就選用 Mega 2560 介紹生活科技產品，下表是其技術規格（摘自 Arduino 官網）。

OVERVIEW | TECH SPECS | DOCUMENTATION

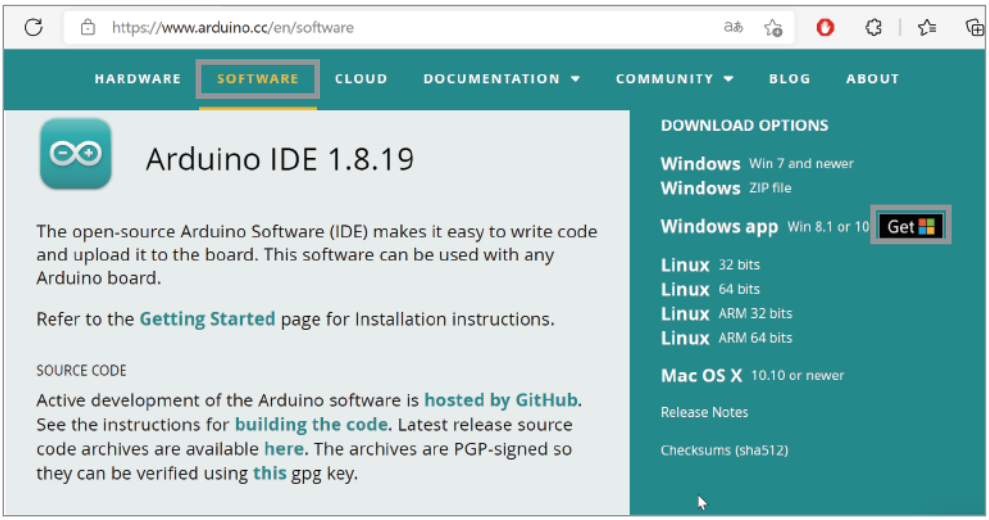
Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz
LED_BUILTIN	13
Length	101.52 mm

下圖是其微控板實體照片（摘自 Arduino 官網），中間黑色正方形 IC 就是 MEGA2560 單晶片，共有 70 隻 I/O 腳位，這些接腳都可以使用軟體的設定，分別指派當作數位輸出、數位輸入、或有上拉電阻 INPUT_PULL UP 的輸入等 3 種功能，這會在本書陸續介紹。



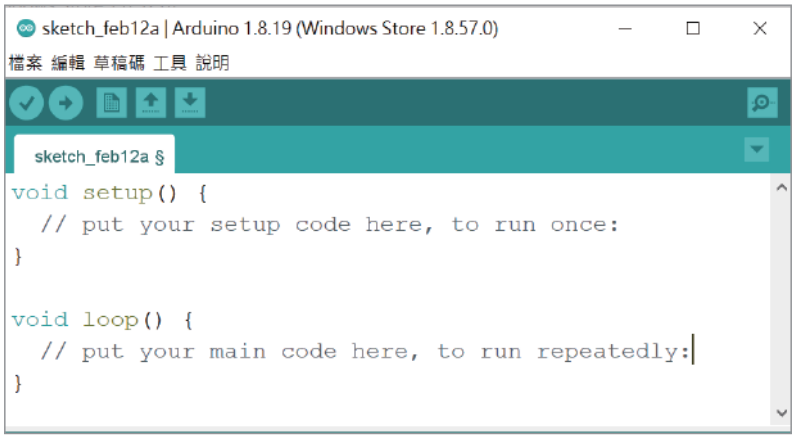
Arduino軟體下載與安裝

所有傳統的單晶片，都要自備類似記事本的編輯器，編寫程式、存檔、離開，然後使用其所提供的編譯程式，若有錯誤則要繼續開啓記事本修改，再編譯，直到完全正確。然後還要購買萬元燒錄器，將程式燒到單晶片。但是 Arduino 就神了，竟然有免費整合編輯視窗，整合以上編輯、編譯、燒錄（上傳）於單一視窗，此稱爲整合編輯程式（IDE, Integrated Development Environment 的縮寫）。請於 Arduino 官網點選『SOFTWARE/DOWNLOADS』，畫面如下，請點選『Windows app win8.1 or 10』，即可下載最新安裝執行檔（*.exe）。接著，請開啓檔案總管，至下載區按兩下該執行檔，即可安裝。



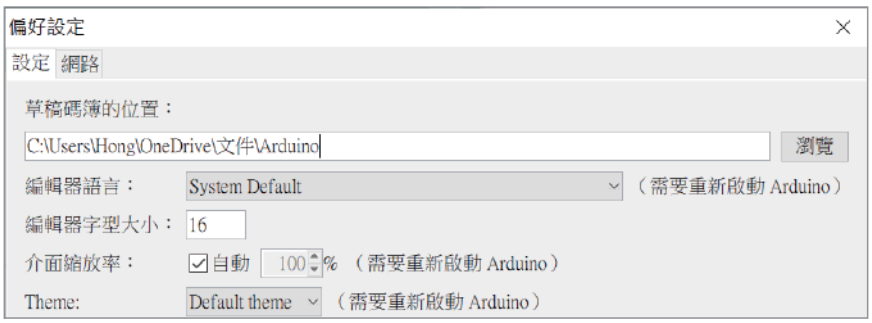
IDE畫面

下圖是開啓 Arduino IDE 畫面。



功能表的語言設定

理論上安裝後，功能表的語言會依據作業系統的語言自動完成設定，但若不是您所要的語言，請於功能表點選『檔案 / 偏好設定』，如下圖，即可在此畫面點選您要的語言，還有編輯器文字的大小等等。



單晶微控板使用步驟

Arduino 的單晶微控板使用步驟分別是插入微控板、點選開發板型號、點選通訊埠編號，分別說明如下：

1. **插入微控板**

   請依照指示，將微控板 USB 插頭插入電腦 USB 插座。請留意微控板右上角電源指示燈是否亮起。

2. **點選開發板型號**

   請點選功能表的『工具 / 開發板』即可點選您所使用的微控板型號。(請依照您的微控板型號點選，筆者是點選『Arduino/Genuino Mega or Mega 2560』，若您是 UNO 版，也是在此點選，因爲不同版本腳位數量不一樣，選錯了就無法正確編譯程式)

3. **點選通訊埠編號**

   請點選功能表的『工具 / 序列埠』即可點選您所使用的序列埠編號（備註：系統會出現可用編號 com1 或 com2,3,4,5 等等，讓您點選，有時候會同時出現很多個編號，請按照順序點選，直到可上傳 (或稱燒錄) 爲止。其次，有些電腦不會自動抓到通訊埠 (com1、com2…)，請到網路搜尋與下載『CH34x-install-Windows』，並安裝，直到出現通訊埠。)

4. **取得開發版資訊**

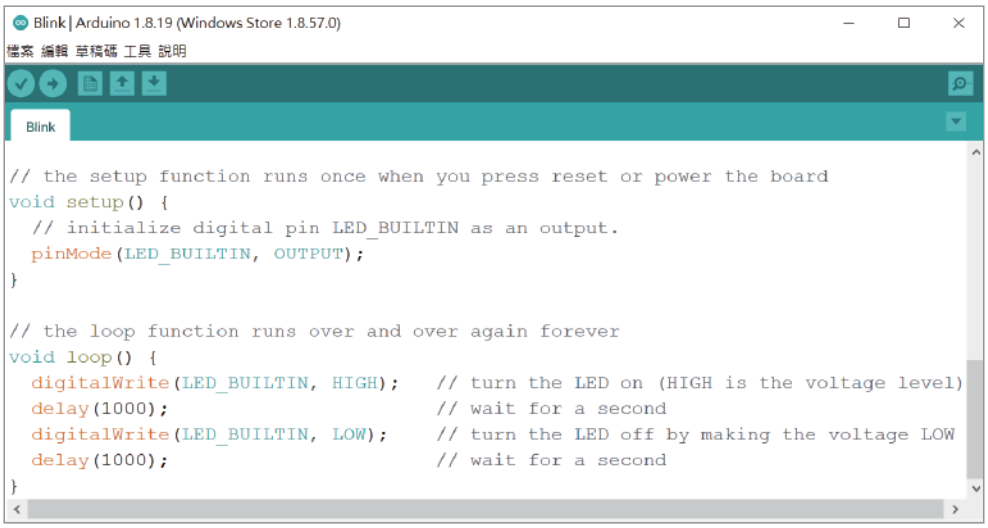
   請點選功能表的『工具 / 取得開發版資訊』即可取得您的微控板資訊。若出現下圖，才表示以上設定就緒，才能上傳程式。(實驗的中途，若改插入別人的微控板，那也要重覆以上步驟，直到看到下圖，才表示有抓到此微控板，才能上傳程式。其次，下圖左是原廠的微控板，若不是原廠，那可能就像下圖右，顯示『未知的開發板』。)

自我測試

Arduino 微控板有內植一顆 LED，腳位編號是 13，別名是 LED_BUILTIN。其次，整合開發環境安裝後，內部也含一個自我測試程式，這樣就可以測試此微控板是否已經安裝完成。進行單晶片控制都要這樣一步一步來，這樣當問題發生時，才能一步一步除錯，逐漸縮小錯誤範圍，並排除故障。以下說明如何自我測試：

1. 開啓自我測試程式

下圖是開啓測試程式 Blink（請點選功能表的『檔案 / 範例 /01.Basics/Blink』），其功能是直接使用微控板預植的 LED（不管什麼板 UNO、MEGA…等），都是腳位 13，以常數『LED_BUILTIN』表示），並令其明滅閃爍各一秒。



2. 驗證程式

按一下工具列的『驗證』按鈕  ，即可編譯程式。

3. 上傳程式

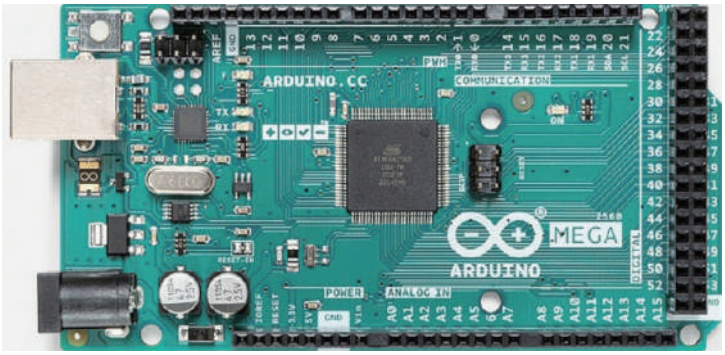
按一下工具列的『上傳』按鈕  ，即可上傳程式到微控板（上傳又稱爲燒錄）。上傳結束將會自動執行程式，請觀察微控板上所預植 LED 是否明滅（此顆 LED 就在腳位 13 旁）。

補充說明

1. setup() 函式僅在程式一開始時執行一次。所以通常放置執行程式的初始設定、或程式執行後只執行一次的指令。
2. pinMode(LED_BUILTIN, OUTPUT); 是指派編號 13 這隻腳的功能是輸出。
3. loop() 函式則會重複不斷的被執行，所以就放置一些需要反覆執行的指令。
4. digitalWrite(LED_BUILTIN, HIGH); 是指派腳位 13 爲高電位，請用三用電表量其電壓，將會有 5V，這樣就可以讓 LED 發光。
5. delay() 函式是程式延遲程式，單位是毫秒 ms，千分之一秒。因爲指派 LED 亮，總要停留一點時間，使用者才有機會看到。請自行調整時間，並觀察執行結果。
6. digitalWrite(LED_BUILTIN, LOW); 是指派腳位 13 爲低電位，請用三用電表量其電壓，將會有 0V，這樣就可以讓 LED 熄滅。
7. 雙斜線『//』稱爲註解，此爲單列註解，僅給人看，電腦不予編譯，沒有鍵入也沒關係。其次『/*』與『*/』之間的文字，也都是註解，此稱爲多列註解，這些註解都是給人看的，電腦不會編譯。

I/O腳位探索

Arduino Mega 2560 有 70 個 I/O 接腳，如下圖。



這些接腳都可單獨使用，那就使用腳位編號（例如，0,1,2,⋯ A1,A2⋯）。也可 8 個 1 組，使用暫存器名稱。例如，22,23,24,25, 26,27,28,29 這 8 隻腳暫存器名稱爲 PORTA，亦可以使用暫存器名稱 PORTA。其餘 PORT 名稱，如下表所示：

暫存器\位元	7	6	5	4	3	2	1	0
PORTA	29	28	27	26	25	24	23	22
PORTB	13	12	11	10	50	51	52	53
PORTC	30	31	32	33	34	35	36	37
PORTD	38				18	19	20	21
PORTE			3	2	5		1	0
PORTF	A7	A6	A5	A4	A3	A2	A1	A0
PORTG			4			39	40	41
PORTH		9	8	7	6		16	17
PORTI								
PORTJ						15	14	
PORTK	A15	A14	A13	A12	A11	A10	A9	A8
PORTL	42	43	44	45	46	47	48	49

以上這些接腳都可以使用軟體的設定，指派腳位功能。其次，腳位功能有三種，分別是數位輸出、數位輸入、或有上拉電阻 INPUT_PULL UP 的輸入等 3 種功能，分別說明如下：

指派腳位功能

Arduino 指派腳位功能有兩種方式，分別是單隻腳位的 pinMode 指派與 8 隻腳一起指派的 DDRA 指令（Data Direction of Port A 的縮寫）。例如，以下程式可使用 pinMode 指派腳位 13 作爲數位輸出。

```
pinMode(13, OUTPUT);
```

以下程式可指派腳位 29,28,27,26,25,24,23,22 作爲數位輸出。

```
    pinMode(29, OUTPUT);
    pinMode(28, OUTPUT);
    pinMode(27, OUTPUT);
    pinMode(26, OUTPUT);
    pinMode(25, OUTPUT);
    pinMode(24, OUTPUT);
    pinMode(23, OUTPUT);
    pinMode(22, OUTPUT);
```

因爲以上腳位 29 ～ 22 剛好是 PORTA 暫存器，所以以上程式亦可簡化爲

```
DDRA=B11111111;// 1是輸出，指派PORTA爲輸出
```

B 表示後續數字代表二進位，1 表示輸出，0 表示輸入。同理，若是

```
DDRA=B00000000;// 0是輸入，指派PORTA爲輸入
```

則指派 PORTA 爲輸入。也就是 pinMode 是配合腳位名稱，一次指派一個腳位的功能；DDRA 是配合暫存器名稱，一次指派 8 個腳位的功能，同理 PORTB、PORTC 就用 DDRB、DDRC 等指派其功能。

數位輸出

Arduino 腳位若當作數位輸出，使用手冊的說明如下：

> Properties of Pins Configured as OUTPUT
>
> Pins configured as OUTPUT with pinMode() are said to be in a low-impedance state. This means that they can provide a substantial amount of current to other circuits. Atmega pins can source (provide positive current) or sink (provide negative current) up to 40 mA (milliamps) of current to other devices/circuits. This is enough current to brightly light up an LED (don't forget the series resistor), or run many sensors, for example, but not enough current to run most relays, solenoids, or motors.
>
> Short circuits on Arduino pins, or attempting to run high current devices from them, can damage or destroy the output transistors in the pin, or damage the entire Atmega chip. Often this will result in a "dead" pin in the microcontroller but the remaining chip will still function adequately. For this reason it is a good idea to connect OUTPUT pins to other devices with 470Ω or 1k resistors, unless maximum current draw from the pins is required for a particular application.

大意是說，當輸出時，若設定爲 HIGH，則電壓有 5V（請留意有些微控板是 3.3V），且每隻腳位高電位的最大輸出電流是 20mA；設定爲 LOW 時，可承受或稱流入 40mA 的電流。其次，因爲 LED 只要 10mA(0.01A) 就很亮，所以直接使用高電位驅動 LED 可說綽綽有餘，但是驅動電流若太大，LED 也會燒毀，所以要加上限流電阻如下：

$$\frac{5-1.7}{0.01} = 330\,\Omega$$

以上 1.7(V) 稱爲 LED 的順向切入電壓，10mA=0.01A，這樣計算時，單位才一致。

指派電位

單晶片的優點是您可直接下指令，指派任何接腳爲 HIGH 或 LOW。指派的方式有兩種，分別是單隻腳位的 digitalWrite 指派與八位元的暫存器名稱 (如 PORTA) 整體指派。例如，以下程式，您可指派接腳 22 爲 HIGH。

```
digitalWrite(22, HIGH);
```

以下程式，您可指派其爲 LOW。

```
digitalWrite(22, LOW);
```

所有腳位都有一個暫存器名稱，還可使用暫存器名稱一起指派 8 隻腳的電位，例如，以下程式可快速指派 PORTA 的 8 個位元全爲 HIGH，PORTA 是暫存器名稱。

```
PORTA=B11111111;
```

以下程式可快速指派 PORTF 的 8 個位元輸出全爲 LOW。

```
PORTF=0;//0就0，當然不用再指派任何進位。
```

範例 1a

PORTA 腳位探索實習。（PORTA 共有 8 隻腳位，本書往後簡稱 PA）

1. 請鍵入以下程式，驗證、上傳。

```
void setup() {
  // put your setup code here, to run once:
    DDRA=B11111111;// 指派PA爲輸出
    PORTA=B11111111;//指派PA爲高電位
}void loop() {}//此函式雖然沒用到，但不能刪除
```

2. 請用三用電表，檢驗微控板的腳位 22, 23, 24, 25, 26, 27, 28, 29 是否爲高電位 5V。
3. 若沒三用電表，請依照附錄 A，焊接一個 LED 電位筆，然後負端先插入 Gnd，正端就可當作電位探測筆，LED 亮就代表有電壓 5V。
4. 鍵入以下程式，重新驗證、上傳，再檢驗微控板的腳位 22, 23, 24, 25, 26, 27, 28, 29 是否爲低電位 0V。

```
void setup() {
    DDRA=B11111111;// 指派PORTA爲輸出
    PORTA=0;//指派PORTA輸出低電位
}void loop() {}
```

自我練習

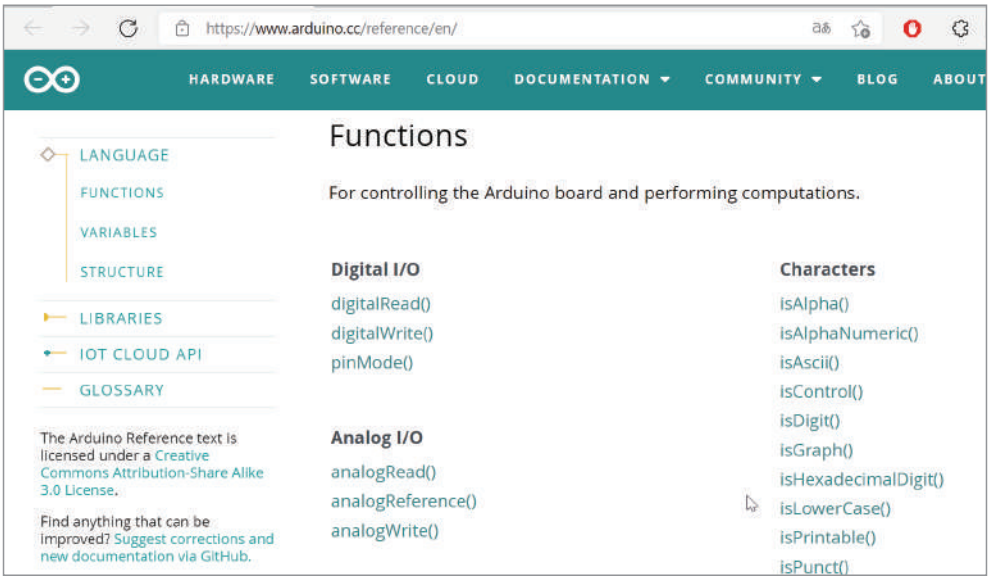
1. 請鍵入以下程式，並驗證 PORTA、PORTB、PORTC 對應腳位是否爲高電位。

```
void setup() {
  // put your setup code here, to run once:
    DDRA=B11111111;//指派PA爲輸出
    PORTA=B11111111;//設定PA0～7均爲高電位
    DDRB=B11111111;//指派PB爲輸出
    PORTB=B11111111;//設定PB0～7均爲高電位
    DDRC=B11111111;
    PORTC=B11111111;
} void loop() {}
```

2. 請探索 PORTF、PORTK、PORTL 腳位在哪裡？並寫程式檢驗這些腳位電壓是否能隨著您的程式而變化。

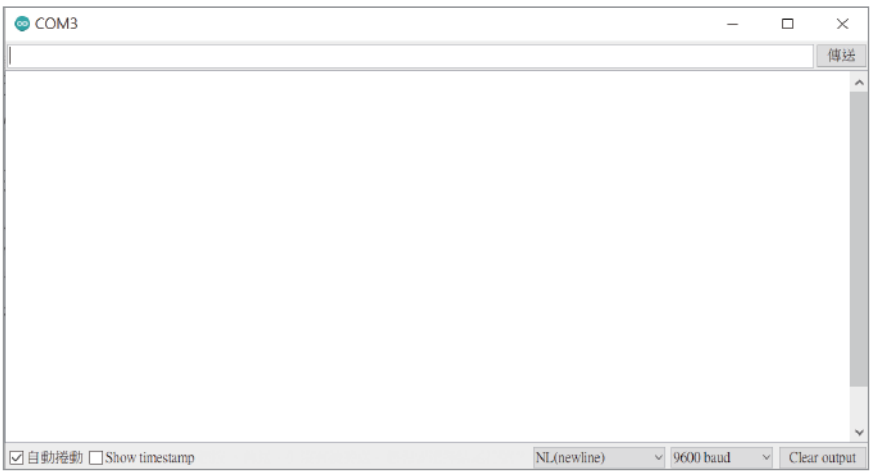
線上使用手冊

點選 Arduino IDE 功能表的『說明 / 參考文件』畫面如下圖（電腦請先連線），這就是 Arduino 所提供的軟體指令手冊，不論指令分類、指令解說、範例等都很詳細，請讀者自行探索。



程式的輸出入與序列埠監控視窗

人類號稱萬物之靈，眼睛、鼻子等五官與手腳是人類的輸出入設備，所有程式語言都是幫助人類處理事情的工具，當然也有輸出入設備。輸出入也是所有程式設計的第一步，本單元先介紹 Arduino 的序列埠監控視窗。序列埠監控視窗如下圖所示：(點選『工具』/『序列埠視窗』)



它可以讓我們用電腦的鍵盤與螢幕和微控板雙向溝通。此一功能是以往其他單晶片所沒有的功能，因爲透過此序列埠監控視窗，不僅可以學習 Arduino 語言，學習 Arduino 所能完成的所有程式設計工作。例如，以下程式可以學習 Arduino 的運算子與數學函式，因爲程式執行後，可在序列埠視窗出現執行結果。

```
void setup() {
    Serial.begin(9600);
    Serial.println(4+2);
    Serial.println(4>=2);
    Serial.println(pow(2,3));
} void loop() {}
```

其次，於開發 Arduino 程式中，從感測器所讀到的值、或運算結果要從輸出元件輸出前，都可將結果先行輸出，先行測試此段程式邏輯與硬體接線是否正確。例如，以下程式，可以將指撥開關的輸入值，先行輸出於電腦螢幕的序列埠視窗，這樣可以確認此部分硬體接線是否正確。

```
a=digitalRead(37);//讀取腳位37的電壓值
Serial.println(a);//於序列埠視窗輸出電壓值
```

以下程式，可以先將運算結果先行輸出，這樣可以檢查此軟體運算是否正確，然後再輸出於 PORTF。

```
b=a+3//處理
Serial.println(b);//於序列埠視窗輸出電壓值
PORTF=b;
```

Serial物件

要讓微控板與電腦溝通，那就要透過 Serial 物件，請開啓 Arduino 參考文件 (點選『說明』/『參考文件』/『Serial』)。Serial 的 Function 如下圖所示。這些方法可以讓我們啓用序列

埠，然後進行輸出文數字、輸入字元、輸入字串與數值，分別說明如下。(補充說明：以下這些函式，在舊式函式導向程式設計稱爲『函式』，但在目前物件導向的程式設計領域，函式已經改稱爲『方法』)

Functions

If (Serial)
available()
availableForWrite()
begin()
end()
find()
findUntil()
flush()
parseFloat()
parseInt()
peek()
print()
println()
read()
readBytes()
readBytesUntil()
setTimeout()
write()
serialEvent()

啓用序列埠

要使用序列埠，首先要使用 begin() 方法啓用序列埠，先規定雙方傳輸速度，本例規定 9600，程式如下：

```
Serial.begin(9600);
```

然後請點選『工具 / 序列埠監控視窗』進入『序列埠監控視窗』點選相同的速度，如下圖：

輸出

要在序列埠輸出文數字，要用 print() 或 println() 方法，兩者的差別是 println() 輸出後換行歸位，print() 則沒有。例如，以下程式，可輸出『GwoshengGwosheng』，如下右圖。

`Serial.print("Gwosheng");` `Serial.print("Gwosheng");`	GwoshengGwosheng

以下程式可輸出兩列 Gwosheng，如下圖右。

`Serial.println("Gwosheng");` `Serial.println("Gwosheng");`	Gwosheng Gwosheng

若是 print 或 println 方法內接變數，則會轉印出此變數的內容。例如，以下程式，可將變數所代表的內容輸出至電腦螢幕。

```
int a=3;//規定資料的儲存空間，宣告a爲整數變數
String b="Gwosheng";//規定資料的儲存空間，宣告b爲字串變數
Serial.println(a);//3
Serial.println(b);//Gwosheng
```

以下程式可輸出『a=3』。有加雙引號『""』的是字串，沒加雙引號的是變數，遇到字串就直接輸出，遇到變數就往前找此變數所儲存的值，並輸出。

```
int a=3;//規定資料的儲存空間，宣告a爲整數變數
Serial.print("a=");//字串就直接輸出a=
Serial.print(a);//變數，輸出變數的值3
```

範例 1b

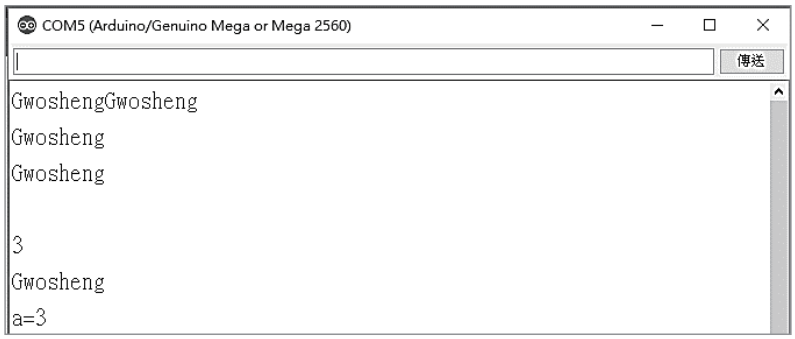
示範以上程式。

操作步驟

本例只要插入 Arduino 微控板就好，不用準備任何電路。

執行結果

程式『驗證』、『上傳』後，請開啓『序列埠監控視窗』（功能表的『工具 / 序列埠監控視窗』）。



程式列印

雙斜線『//』稱爲註解，註解後的文字，僅給人看，電腦不予編譯，初學者可以不用打。

```
void setup() {
    // put your setup code here, to run once:
    Serial.begin(9600);
    Serial.print("Gwosheng");
    Serial.print("Gwosheng");
    Serial.println();//only linefeed
    Serial.println("Gwosheng");
    Serial.println("Gwosheng");
    Serial.println();//only linefeed
    int a=3;//規定資料的儲存空間，請看第十單元
    String b="Gwosheng";//規定資料的儲存空間，請看第十單元
    Serial.println(a);
    Serial.println(b);
    Serial.print("a=");
    Serial.print(a);
}void loop() {}//此函式雖然沒有放程式，但也不能刪除
```

自我練習

1. 請於序列埠監控視窗輸出自己的座號與名字，本例輸出座號後換行歸位，再輸出姓名。
2. 同上題，請在同一列輸出自己座號與名字。

■※ 輸入（不常用，教學時數少時，可先跳過）

Arduino 可以分別使用 read()、readString() 及 parseInt() 等方法讀取使用者於鍵盤所輸入字元、字串與數字，分別說明如下：

字元輸入

要讓序列埠輸入字元（僅一個英文字母或數字視爲字元），也是要同上範例，先使用 begin() 啓動序列埠，再使用 read() 方法讀入字元，程式如下：

```
char c=Serial.read();//宣告c 爲char型態變數，請看第十單元
```

available() 方法

大部分程式語言的鍵盤輸入功能，遇到輸入指令，都會停留在此指令，癡癡的等待使用者輸入資料，直到使用者按『Eenter』，再繼續往下執行。但是 Arduino 有特殊原因，它必須兼顧很多輸入設備，所以不會也不能等待任何輸入設備。若特別需要它一定要原地等待使用者鍵盤輸入，那就要使用 available() 方法，因爲 available() 方法可判斷使用者是否已經輸入，available() 的語法如下：

```
Serial.available()
```

1. 也就是 available() 傳回大於等於『1』，才表示使用者有輸入，例如，以下程式，會請單晶原地等待您輸入，若鍵盤緩衝區沒有資料，就重複迴圈，也就是原地癡癡的等待。while 稱爲迴圈指令，請看第 8 單元

```
void setup() {
    Serial.begin(9600);
}
void loop() {
    while(Serial.available() ==0) {}//若未輸入，則一直在此等待。
    char c=Serial.read();
    Serial.println(c);
}
```

2. 請輸入以下程式（沒有使用 available() 函式），並觀察執行結果。

```
void setup() {
    Serial.begin(9600);
}
void loop() {
    char c;
    c=Serial.read();
    Serial.println(c);
}
```

3. 以下左右程式也不同。請自行鍵入、執行，並觀察結果。下圖左是正確的，while 迴圈有加兩個 {}；下圖右是錯誤的，c =Serial.read() 會屬於 while 迴圈，因爲 while 沒有大括號，那此迴圈會包含與執行一個敘述，這都在第 8 單元詳細介紹。

```
void setup() {
    Serial.begin(9600);
}
void loop() {
    char c;
     while(Serial.available() ==0)
{}
    c =Serial.read();
    Serial.println(c);
    Serial.println("aa");
}
```

```
void setup() {
    Serial.begin(9600);
}
void loop() {
    char c;
    while(Serial.available() ==0)
       c =Serial.read();
    Serial.println(c);
    Serial.println("aa");
}
```

字串輸入

readString() 可讀取字串（一連串的字元稱爲字串），例如，以下程式可讀取一個字串，然後輸出字串。

```
void setup() {
    Serial.begin(9600);
}
void loop() {
    while(Serial.available() ==0) {}
    String c=Serial.readString();
    Serial.println(c);
}
```

自我練習

請輸入以上程式，執行程式，輸入字串，並觀察執行結果。

數字輸入

Mega 2560 還可直接輸入數字，例如，以下程式可輸入一個整數。

```
int a=Serial.parseInt();
```

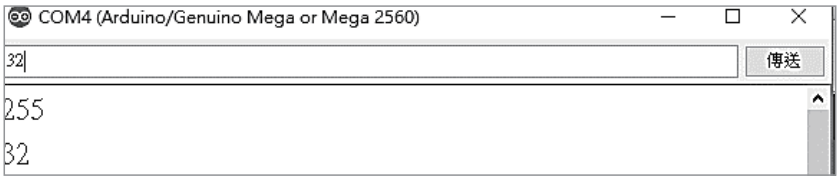
int 是變數的資料型態，表示 a 可儲存 -32768 ～ 32767 的整數，將於下一單元進一步介紹。

範例 1c

示範以上程式。

輸出結果

請開啓序列埠監控視窗（功能表的『工具 / 序列埠監控視窗』）。



程式列印

```
void setup() {
    Serial.begin(9600);
}
void loop() {
    int a;
    while(Serial.available() ==0) {}//程式在此等待使用者輸入
    a=Serial.parseInt();
    Serial.println(a);
}
```

自我練習

1. 有了 Serial 鍵盤與螢幕輸出入方法，那也可把 Arduino 拿來作爲學習 C 語言程式設計的工具，寫出 C 語言能做的程式。例如，請寫一程式，可以輸入兩個整數、計算其和，且輸出結果（提示：每輸入一個整數，就要一個 while 迴圈等待。其次，往後各章還有很多這方面的題目，可用來學習 C 語言）。

第2單元

電腦運算思維與資料數位化

電腦的運算思維

電腦與人類有點不同，人類處理 3+2，可說非常直覺，馬上回答 5。但電腦就有點不同了，電腦要進行資料處理，都要先將資料以變數儲存，然後放到電腦記憶體，接著將記憶體內容放到中央處理運算單元（簡稱 CPU）運算，CPU 運算結果還是要先以變數爲名稱丟到記憶體，然後再將這些記憶體輸出到螢幕，此即爲電腦的運算思維。本單元則要談電腦運算思維的第一步『資料的數位化』。也就是資料要數位化，資料才能放到主記憶體，這也是電腦運算思維的第一步。

其次，電腦和人腦的運算思維有點不同，人腦判斷能力強，運算能力較差，電腦則是執行判斷較費時，計算能力超強且速度快。學習程式設計，當然要先學習電腦這些運算思維，這樣電腦才能精準且快速的完成指定工作。以下將陸續介紹一些電腦運算思維，這樣寫程式才能事半功倍。

電腦的最小記憶單元爲位元 (bit)，每個位元僅能儲存 0 或 1，8 個位元稱爲一個位元組 (Byte)，那資料如何儲存於電腦呢？答案就是要將資料數位化。

資料數位化方法

程式設計最常見的資料爲數值、字元、字串，將以上資料轉爲二進位的方式，才能儲存於電腦，以上資料的數位化方法，分別說明如下：

■ 正整數

所有整數都要先指定資料的長度，本例先假設是 8 位元，其餘 16 位元或 32 位元也是相同原理。其次，還要假設僅儲存正數，還是同時儲存正負整數。首先，以儲存正數爲例，那其二進位編碼如下，共可表示 0 到 255 的正整數。

正數	二進位編碼
0	00000000
1	00000001
2	00000010
3	00000011
4	00000100
5	00000101
6	00000110
7	00000111
254	11111110
255	11111111

範例 2a

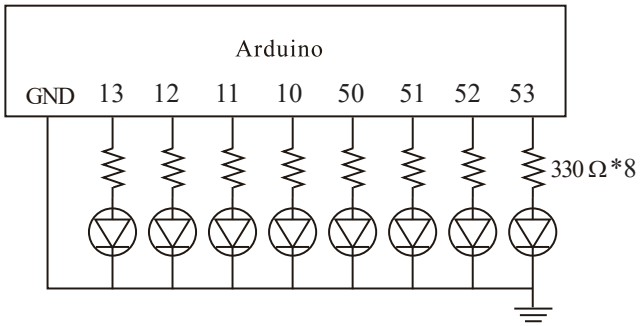
示範正整數的數位化。請鍵入以下程式，並觀察 8 個 LED 的亮滅，請問與上表的正數是否相符。

輸出結果

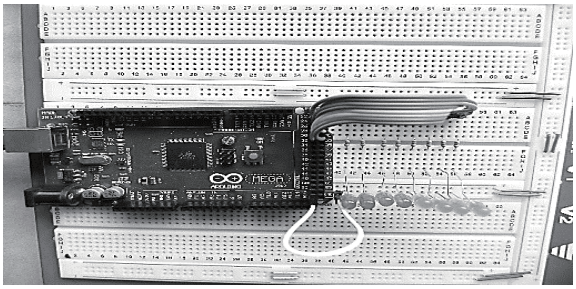
請留意 8 個 LED 代表 8 位元，燈號『亮』就是 1，燈號『滅』就是 0。

操作步驟

1. 請準備 LED 電路如下圖。

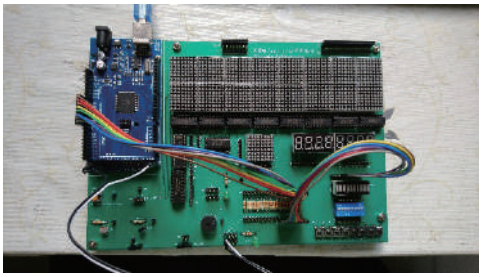


2. 若使用麵包板，則實體接線如下圖：（電阻 330Ω 顏色是橙橙棕，LED 長腳爲正，若腳被剪斷過，則往 LED 裡面看，接點小的爲正端，接點大的爲負端，下圖 LED 長腳請接電阻，短腳接地）



2. 若使用本書以下實驗板，只要將 13,12,11,10,50,51,52,53 的腳位使用杜邦線連接到限流電阻 (J16)，再從限流電阻另一端 (J17) 連接到 LED (J18) 即可，Gnd 內部已自動連接。在電源指示燈正常情況下，即可進行實驗。其次，因爲微控板有過載保護，若微控板電源指示燈熄了，實驗板指示燈也會熄滅，表示您的電路有問題，有導線短路了，微控板自動切掉電源，此時請迅速拔掉電源連接線，檢查電路，直到指示燈正常爲止。

3. 杜邦線的顏色有意義，請依電阻的色碼使用，『棕紅澄黃綠藍紫灰』分別代表 1 到 8，『白黑』請用來接正負電源。

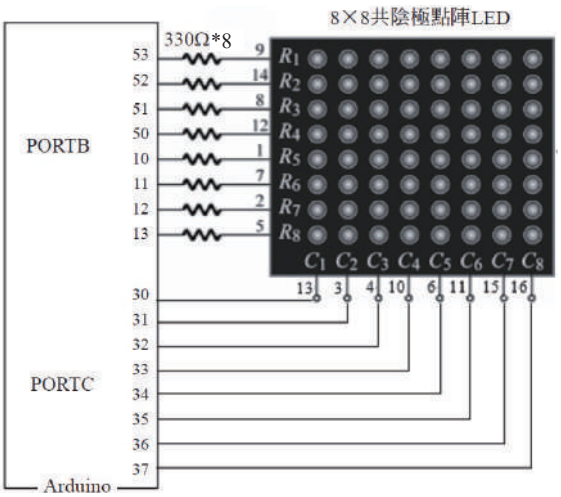


程式列印

1. 鍵入以下程式，並觀察輸出結果。

```
void setup() {
  DDRB=B11111111;  //指派PORTB功能爲輸出
  Serial.begin(9600);//啓動序列埠
}
unsigned char i=12; //宣告i爲8位元正整數,這樣i可儲存0~255的整數
void loop() {
  PORTB=i;
  Serial.println((int)i);//轉爲整數
}
```

2. 若使用本書中學生實驗版，此實驗板並未準備以上 LED 電路，而是拿 8*8 點陣 LED 來當作 8 位元 LED。此實驗板內部接線如下：



請在以上程式加入以下底線字程式。本書往後使用任何獨立 LED 的程式也都這樣，直接使用軟體設定的方式，設定 C1 ～ C7 皆爲高電位，C8 爲低電位，那此 8*8 點陣 LED，C8 Column 就可當作 8 個 LED 來使用。

```
void setup() {
  DDRB=B11111111;//指派PORTB全爲輸出
  DDRC=0xFF;PORTC=0xFE;//0xFE is B11111110 將8*8點陣LED當作8個LED使用
  Serial.begin(9600);
}
unsigned char i=12; //宣告i爲8位元正負整數,這樣i可儲存0~255的整數
void loop() {
  PORTB=i;
  Serial.println((int)i);//轉爲整數
}
```

3. 請鍵入以下程式，並觀察序列埠視窗與 LED 的燈號，驗證 0 到 255 的數位化。

```
void setup() {
  DDRB=B11111111;
  Serial.begin(9600);
  DDRC=0xFF;PORTC=0xFE;//0xFE is B11111110 將點陣LED當作8個LED使用

}
unsigned char i=0; //宣告i爲8位元正整數,這樣i可儲存0~255的整數
void loop() {
  PORTB=i; //使用LED顯示記憶體內容
  Serial.println((int)i);//轉 爲 整 數 ，使用序列埠輸出記憶體內容
  delay(1000);
  i=(i+1)%256;//保障在0到255循環
}
```

自我練習

請自行將 112 轉爲二進位，並觀察結果是否相符。

■正負整數

若要考慮正負整數，那可用最高位元則用來表示正或負數，最高位元爲 0 時表示正數，所以可表示的範圍是 0 到 127 如下表；

二進位編碼	數字
00000000	0
00000001	1
00000010	2
01111110	126
01111111	127

最高位元爲 1 時表示負數，那負數編碼如下：

二進位編碼	數字
10000000	-0
10000001	-1
10000010	-1
11111110	-126
11111111	-127

這樣會有 0 與 -0 的問題，0 重複出現，造成不是一對一函式，這樣函數與反函數互轉比較麻煩。所以計算機前輩就繼續腦力激盪，想出二補數的編碼，且沿用到今天。二補數的編碼如下：正數就直接編碼，同上；負數就取其 2 補數。以 -3 爲例，先將 3 轉爲二進位

00000011

取 1 補數，1 補數是 0 變 1，1 變 0，所以是

11111100

加 1，所以是

11111101

也就是 2 補數是 1 補數加 1。以上即爲 -3 的 2 補數編碼，等一下我用燈號證明給您看，您會看到『亮，亮，亮，亮，亮，亮，滅，亮』。同理，看到燈號爲

11111101

最高位元爲 1，表示負數，那到底負多少？就取 2 補數，2 補數是先取 1 補數（0 變 1，1 變 0 稱爲 1 補數）再加 1。以上

11111101

先取 2 補數

00000010

加 1

00000011

那就是 -3。所以 10000000 代表 -128，10000001，代表 -127,…11111111 代表 -1，以上即爲 2 補數的編碼，如下表所示，這樣就沒有正負 0 的問題了。其次，2 補數的編碼竟然也同時解決減法問題，眞是一石二鳥。因爲

8-3

可以看成

8+(-3)

也就是直接將 3 取 2 補數，再相加就可以。因爲計算機內部組織只有加法器與比較器，繼續研讀本書，您會發現乘法也是可以用加法完成，除法也是用加減法完成。

二進位編碼	數字
00000000	0
00000001	1
00000010	2
00000011	3
00000100	4
00000101	5
00000110	6
00000111	7
01111110	126
01111111	127 最大正數
10000000	-128 最小負數
10000001	-127
11111110	-2
11111111	-1

範例 2b

示範包含正負整數的數位化。請鍵入以下程式，並觀察 8 個 LED 的亮滅，請問與上表的正負數是否相符？（電路同範例 2a）

程式列印

1. 以 -3 爲例，先將 3 轉爲二進位

   00000011

   取 1 補數

   11111100

   加 1，所以是

   11111101

2. 看到 11111101，最高位元爲 1，表示此爲負數，到底負多少，那也是取 2 補數，先取 1 補數如下：
   00000010
   再加 1，結果是
   00000011
   所以是 -3。
3. 請鍵入以下程式，並觀察執行結果。

```
void setup() {
  DDRB=B11111111;
  DDRC=0xFF;PORTC=0xFE;//0xFE is B11111110 將8*8點陣LED當作8個LED使用
  Serial.begin(9600);
}
char i=-3; //宣告i爲8位元正負整數,這樣i可表示-128~127
void loop() {
  PORTB=i;
  Serial.println((int)i);//轉爲整數
}
```

補充說明

1. 本例 Serial.println((int)i);(int) 爲將字元型態轉爲整數，請修改爲 Serial.println(i), 並觀察執行結果。

自我練習

請自行將 -112 轉爲二補數，並觀察結果是否正確。

■字元

鍵盤能用的字元有大小寫的 a,b,c、數字 0 ～ 9、還有一些控制字元，例如，跳列、歸位（回到該列最左邊）、跳頁、return 等，這些字元總共沒有超過 127 個，因此使用 1 個 byte 儲存就綽綽有餘。所以這些字元的編碼，在 1960 年就已經編碼完成，稱爲 ASCII 碼 (American Standard Code for Information Interchange)，

如下圖所示：也就是我們將所有字元編號，此編號與我們的座號相同，當叫到 65 號，大家都能體認此爲字元『A』，當叫到 33 號，就表示此爲字元『！』。

DEC Value	Character
0	null
1	
2	
3	
4	
5	
6	
7	
8	
9	tab
10	line feed
11	
12	
13	carriage return
14	
15	
16	
17	
18	
19	
20	
21	
22	

DEC Value	Character
32	space
33	!
34	"
35	#
36	$
37	%
38	&
39	'
40	(
41	)
42	*
43	+
44	,
45	-
46	.
47	/
48	0
49	1
50	2
51	3
52	4
53	5
54	6

DEC Value	Character
64	@
65	A
66	B
67	C
68	D
69	E
70	F
71	G
72	H
73	I
74	J
75	K
76	L
77	M
78	N
79	O
80	P
81	Q
82	R
83	S
84	T
85	U
86	V

DEC Value	Character
96	`
97	a
98	b
99	c
100	d
101	e
102	f
103	g
104	h
105	i
106	j
107	k
108	l
109	m
110	n
111	o
112	p
113	q
114	r
115	s
116	t
117	u
118	v

由上圖可知共使用 7 位元，0 到 127，前面 32（0 ～ 31）爲控制字元，48 ～ 57 是 0 ～ 9 的數字，65 ～ 90 是大寫英文字元，97 ～ 122 是小寫英文字元。

範例 2c

示範英文小寫字元的表示。請自行鍵入以下程式，並觀察燈號與輸出結果。（電路同範例 2a）

輸出結果

1. 請留意字元 'a' 的表示，且內部是以 (97)10=(61)16 表示，燈號的明滅就是其二進位，本例會是『滅，亮，亮，滅，滅，滅，滅，亮』，代表 01100001。

```
void setup() {
  DDRB=B11111111;
  Serial.begin(9600);
     DDRC=0xFF;PORTC=0xFE;//0xFE is B11111110 將8*8點陣LED當作8個LED使用
}
char i ='a';//字元用單引號  ;
void loop() {
  Serial.print((int)i);Serial.print(":");Serial.println(i);
  PORTB=i;//97,0x61
  delay(1000);
}
```

自我練習

1. 請自行於 Arduino 線上手冊搜尋『ASCII』。
2. 請將 a 改爲大寫 A，並觀察記憶體內容，填入下表：

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0

3、請將 char i='a' 改爲 char i=1，並觀察記憶體內容，填入下表。

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0

4、請將 char i='a' 改爲 char i='1'，並觀察記憶體內容，填入下表，請問是否與 ACCII 的符號 1 相同。

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0

■字串

連續字元的集合稱爲字串，例如，

```
"ABC"
```

請留意前面字元用單引號，本例字串則用雙引號『"』。C 語言使用字元陣列表示字串。例如，以下陣列 a 即可儲存以上字串。

```
char a[]="ABC";//字串用雙引號
```

變數經過以上宣告，往後就可以使用索引存取內部字元。請鍵入以下程式，並開啓序列埠視窗，觀察執行結果。

```
void setup() {
  Serial.begin(9600);
  char a[]="ABC";
  Serial.println(a);//ABC
  Serial.println(a[0]); //A
  Serial.println(a[1]); //B
  Serial.println(a[2]);//C
  a[0]='D';
  Serial.println(a);//DBC
}void loop() {}
```

C++ 則新增字串型態 string , 例如，

```
string a="ABC";//字串用雙引號
```

變數經過以上宣告，往後就可以使用索引存取內部字元。請鍵入以下程式，並開啓序列埠視窗，觀察執行結果，就會明瞭。

```
void setup() {
  Serial.begin(9600);
  String a="ABC";
  Serial.println(a);//ABC
  Serial.println(a[0]); //A
  Serial.println(a[1]); //B
  Serial.println(a[2]);//C
  a[0]='D';
  Serial.println(a);//DBC
}void loop() {}
```

以上兩種字串表示，Arduino 都可以接受，如以上程式。

範例 2d

示範字串的表示。請自行鍵入以下程式，並觀察燈號與輸出結果。（電路同範例 2a）

輸出結果

1. 請留意 a[0] 是字元'A'，編號是 65，內部記憶體是 01000001(=0x41)，燈號會是 01000001。
2. a[1] 是字元 'B'，編號是 66，a[2] 是字元 'C'，編號是 67。

```
65:A
66:B
67:C
```

程式列印

```
void setup() {
  DDRB=B11111111;
  DDRC=0xFF;PORTC=0xFE;//0xFE is B11111110 將8*8點陣LED當作8個LED使用
  Serial.begin(9600);
}
void loop() {
  //char a[]="ABC";//C語言的字串宣告方式
  String a="ABC";//C++語言的字串宣告方式，Arduino兩者都可以接受
  Serial.print((int)a[0]);Serial.print(":");
  Serial.println(a[0]);
  PORTB=a[0];//65,0x41
  delay(1000);
  Serial.print((int)a[1]);Serial.print(":");
  Serial.println(a[1]);
  PORTB=a[1];//66,0x42
  delay(1000);
  Serial.print((int)a[2]);Serial.print(":");
  Serial.println(a[2]);
  PORTB=a[2];//67,0x43
  delay(12000);
}
```

第3單元

資料的運算

前面我們已經介紹資料的數位化，本單元則要進行資料的運算，要進行資料的運算必須先介紹一些電腦的基本運算思維。例如，運算元、資料型態、運算子、算術運算、變數宣告、識別字、保留字等等，分別說明如下：

運算元與資料型態

一般的計算機進行算術運算，就直接運算，例如進行 3 * 2，就直接按『3』、『*』、『2』、『＝』就可得到答案，但是以上步驟當要重複計算時，都要重複按以上運算元與運算子（3 與 2 稱爲運算元，*與 = 稱爲運算子），這樣非常耗時與不便，所以就有電腦程式語言的發展。所謂電腦程式設計，就是利用代號儲存以上運算元，這些代號在程式設計領域稱爲變數，意思是說，它所代表的值，運算過程隨時可以改變，再使用變數寫出計算結果的敘述，結果也是先以變數儲存，那此一敘述就可重複利用，例如，計算長方形面積，我們會先將資料數位化：

```
int a=3,b=4;
int c;
```

計算結果與輸出結果如下：

```
c=a*b;
Serial.print(c);
```

以上代號 a, b, c，在程式設計領域裡，我們稱爲『變數, variable』、『int b=3;』稱爲變數宣告與指派初值；『a*b』稱爲『運算式, Expression』；『c=a*b;』稱爲『敘述, Statement』。其次，在上一單元的資料數位化單元，我們已經介紹不同的數字（整數、負整數與實數）與不同的數字的大小（數字的位數），其數位化的方式也不同，所佔用的記憶體大小也不同。電腦爲了有效率的處理這些資料，就有資料型態（Data Types）的規劃，也就是大的資料用大盒子裝，小的資料用小盒子裝，如此才可節省記憶體，並加快處理效率。反過來說，若不分資料大小，通通用大盒子裝資料，那將會非常浪費記憶體，也拖垮執行效率。例如，所有的東西都用冰箱的盒子裝當然也可以，但這樣非常浪費空間，還有，搬運時也很耗時。Arduino 所提供的資料型態、所佔用記憶體、所能代表的數值範圍如下表：

資料型態	中文名稱	佔用記憶體的大小（位元）	所能代表的數值的範圍	備註
byte	位元組	8	0 ～ 255	
int	整數	16	-32768 ～ 32767	
long	長整數	32	-2147483648 ～ 2147483647	
float	浮點數	32	+/-3.4E+-38	
double	倍精度浮點數	32	+/-3.4E+-38	Arduino 的 double 同 float
unsigned char	正字元	8	0 ～ 255	
char	字元	8	-128 ～ 127	
unsigned long	正長整數	32	0 ～ 42949667295	

資料型態	中文名稱	佔用記憶體的大小（位元）	所能代表的數值的範圍	備註
unsigned int	正整數	16	0 ～ 65535	
bool	布林	8	true or false	boolean 也可，但不鼓勵
String	字串			

變數宣告

變數的功能是用來輸入、處理及儲存外界的資料，而變數在使用以前則要事先宣告才可使用。在一些舊式的 Basic 語言中，變數使用前並不需要事先宣告，卻也帶來極大的困擾。例如，以下敘述即爲變數未宣告的後果：編譯器無法回應使用者在拼字上的錯誤，而造成除錯上的困難。

```
student=studend+1;
```

上式若事先宣告 student 如下：

```
int student;
```

則編譯器遇到 studend 時，便會出現 studend 未宣告的錯誤訊息，提醒使用者補宣告或注意拼字錯誤。其次，變數宣告的優點是可配置恰當的記憶體而提高資料的處理效率。例如，有些變數的值域僅爲整數，則不用宣告爲 float。此即爲小東西用小箱子裝，大東西用大箱子裝，才能有效運用空間與提升搬運效率。Arduino 語言的變數宣告語法如下：

```
資料型態 變數名稱[=初值];
```

例如，

```
byte a;
```

即是宣告變數 a 爲 byte 型態，佔用 1 個 Byte，此種型態僅能儲存 0 到 255。又例如，

```
int d;
```

那 d 就佔用 2 個 Byte，但可儲存 -32768 到 32767。又例如，

```
char b,c;
b='A';c='*';
```

則是宣告變數 b, c 爲 char 型態，此種型態可儲存單一字元。變數的宣告亦可連同初值一起設定，如以下敘述：

```
float d = 30.2;
```

宣告 d 是 float 型態，並設其初值是 30.2。以下敘述，宣告 e 是 char 型態，且其初值是字元 'a'。

```
char e='a';
```

以下敘述，同時宣告兩個變數，且設定其初值。

```
int f1=3,f2=3;
```

以下敘述可宣告布林型態:（布林型態用來表示，運算結果的『眞』與『僞』）

```
bool g=true;
```

C/C++/Arduino 都可用字元陣列表示字串，以下程式可宣告 a 爲字串型態，請留意字元是單引號，字串是雙引號。

```
char a[]="ABC";//字串用雙引號
```

C++/Arduino 才有字串型態 String，以下程式可宣告變數 h 爲字串型態：

```
String h="123";
```

變數經過宣告之後，編譯器即會根據該變數的資料型態配置適當的記憶體儲存此變數，所以若要提高程式的執行效率，則應儘量依照資料性質，選擇佔用記憶體較小的資料型態。

變數的有效範圍

任一變數的宣告，若無特殊聲明，均屬於區域變數，其有效範圍僅止於該變數所在的程式區塊。所以，以下變數 i 的有效範圍僅在 setup()，無法在 loop(){} 函式內存取。

```
void setup() {
  Serial.begin(9600);
  byte i=1;
}
void loop() {
  Serial.println(i);
}
```

其次，請比較下圖左與下圖右的差別，下圖左 byte i=0; 放在 void loop() {} 裡面，則每次執行 void loop() {} 時，byte i=0 都被執行，所以其值永遠都相同，沒有累加效果。此時就要把此敘述放在外面，成爲全域變數，所以此 i，稱爲計數器，也可以想像成 loop() 被執行的次數，如下圖右。

```
void setup() {
  Serial.begin(9600);
}
void loop() {
  byte i=0;
  i=i+1;
  Serial.println(i);
}
```

```
void setup() {
  Serial.begin(9600);
}
byte i=0;
void loop() {
  i=i+1;
  Serial.println(i);
}
```

變數有效範圍的優點

因爲一個大型程式通常是很多人合力完成，每個人各自開發自己的函式，各自宣告自己的變數。例如，以下變數 a，彼此獨立，互不干擾。

```
void setup() {
  Serial.begin(9600);
  byte a=1;
  Serial.println(a);
}

void loop() {
  byte a=2;
  Serial.println(a);
  delay(1000);
}
```

若沒有區域變數的觀念，彼此的變數就會互相干擾，例如，以下程式，您用 a，我也用 a，那彼此就互相干擾，執行結果就會錯亂。

```
void setup() {
  a=a+1;
}
void loop() {
  a=a+3;
}
```

若需要共用變數時，大家要先講好，將這一變數宣告爲全域，就可共用。例如，以下變數 a 稱爲全域變數，大家一起共用。

```
byte a=0;
void setup() {
  Serial.begin(9600);
  Serial.println(a);
  a=a+1;
  Serial.println(a);
}
```

```
void loop() {
  Serial.println(a);
  a=a+3;
  Serial.println(a);
  delay(1000);
}
```

識別字（Identifiers）命名規則

眞實的世界裏，每個人、事及物都有一個名稱，程式設計亦不例外，於程式設計時我們必須爲每一個變數、常數、函式命名，以上所有變數、常數、函式等名稱，統稱爲程式語言的識別字（Identifiers）。先以數學爲例，數學的變數命名規則是，已知數用 a, b, c，未知數用 x, y, z；物理則習慣用 v 代表速度，t 代表時間；而 C/C++/Arduino 語言的識別字命名規則如下：

1. 識別字必須是以字母（大小寫的 A 至 Z）或底線（_）開頭。例如，以下是一些合法的識別字。

```
a
i
sum
_sum
Income
```

以下是一些非法的識別字。

```
7eleven      //  不能由數字開頭
%as          //  不能由符號開頭
```

2. 識別字由字母開頭後，僅可由字母、底線及數字組合而成，且不得包含空白與符號。例如，以下是一些合法的識別字。

```
a123
a123b
_a_b
```

以下是一些非法的識別字。

```
A=          //  不能含有 = 號
sum!        //  不能含有 ! 號
Age#3       // 不能含有 # 號
a c         //  不能含空白
c+3         //  不得含有加號
```

3. 識別字的大小寫均視爲不同，例如 Score、score 及 SCORE 皆代表不同的識別字。
4. 識別字不得使用保留字，如 if、for 等。但是，if1、fora 等則可使用。
5. 識別字要用有意義的單字，例如，sum、avg、StudentNumber 或 AverageIncome。除非有效範圍很小（或稱生命週期極短）的變數才用 x、i 或 a 等當識別字，也千萬不要用 k23erp 等這種沒意義又難記的識別字。
6. 識別字有多個單字時，中間可以加上底線（_），例如上例的 StudentNumber 可寫成 student_Number，若擔心打字不靈光亦可寫成 Stu_Num、stu_num、stuNum 或 stunum，其中 stuNum 又稱駝峰表示法，因爲大寫字母看起來像駱駝駝峰一樣，這樣可以避免鍵入底線的困擾，且提昇閱讀效率，Arduino 的函數命名則採用此種命名習慣。

保留字（Keywords）

保留字（又稱關鍵字）是任一程式語言已事先賦予某一識別字（可識別的文字或字串，稱爲識別字）一個特別意義，所以程式設計者不得再重複賦予不同的用途，就像現實生活中，電視、冰箱已經有特別意義了，所以沒有人取名字爲電視或冰箱。Arduino 也是一樣，例如 if 已賦予決策功能，程式設計者當然不得再定義

if 爲另外的用途，請自行線上查詢點選工具列的『說明 / 參考文件』或『Language Reference』，例如表內的 loop、setup、break、coutinue、do 等等，大部分擷取 C/C++ 語言精華。

運算子

所謂運算子（Operator），指的是可以對運算元（Operand）執行特定功能的特殊符號。例如，3 ＋ 2 的『3』與『2』稱爲運算元，『+』稱爲運算子。Arduino 的運算子分爲五大類，分別是：算術（Arithmetic）運算子、比較（Comparison）運算子、布林（Boolean）運算子、位元操作（Bitwise）運算子及複合（Compound）運算子，分別說明如下：

■ 算術運算子

下表是 Arduino 的算術運算子。（請開啓 https://www.arduino.cc/reference/en/）

Arithmetic Operators

% (remainder)

* (multiplication)

+ (addition)

- (subtraction)

/ (division)

= (assignment operator)

以上算術運算子用來執行一般的算術運算，包括指派 (=)、取正負數 (+/-)、加 (+)、減 (-)、乘 (*)、除 (/)、取餘數 (%) 等，下表是以上算術運算子的功能說明：

運算子	定義	優先順序	結合律
=	指派	15	由右至左
+/-	正負號，一元運算子	2	由右至左
*	乘法運算	4	由左至右
/	除法運算	4	由左至右
%	求餘數（Modulus）	4	由左至右
+/-	加法 / 減法運算	5	由左至右

■『=』

『=』符號爲指派或稱賦值運算子，其作用爲將運算符號右邊運算式的值，指派給運算符號左邊的運算元。所以，以下敘述 sum=a+b 是先計算 a+b 的值，再將此值指派給 sum。

```
int sum = 0,a = 3,b = 5;
sum = a + b;
```

上式與數學的『相等』符號是不同的，在程式設計領域裡，此稱爲指派或賦值，右邊先運算，運算後再將結果指派給左邊的變數，所以不要一直糾結爲什麼 0 會等於 8。其次，你是不能將常數放在指派運算子的左邊，例如：

```
8 = x ;
```

爲一個不合法的敘述，但以下敘述將常數 8 指派給變數 x 是合法的。

```
x = 8 ;
```

■四則運算

以下是一些簡單四則運算：

```
int a=5,b=4;
Serial.println(a+b);//9
Serial.println(a-b);//1
Serial.println(a*b);//20
Serial.println(a%b);//1 取餘數
```

以上程式，請放在 setup() 裡面，就可觀察執行結果。

```
void setup() {
  Serial.begin(9600);
  //執行一次的放這裡
}
void loop() {}//雖然沒用到，也不能省略
```

■整數除法或實數除法

Arduino/C/C++ 的除法運算，只有被除數與除數的型態均爲整數，才是整數除法，商的型態爲整數；否則即爲實數除法，得到實數商。例如：

```
int x=5, y=4,z;
float xf=5,yf=4;
Serial.println(x/y); // 1，被除數與除數的型態均爲整數
Serial.println(xf/y);//1.25
Serial.println(x/yf);//1.25
Serial.println(xf/yf);//1.25
```

其次，若運算結果爲實數，但若指派給整數型態的變數，結果也是整數。例如：

```
void setup() {
    Serial.begin(9600);
    int x=5, y=4,z;
    float xf=5,yf=4,zf;
    zf=xf/yf;//1.25
```

```
    Serial.println(zf);
    z=xf/yf;//1
    Serial.println(z); //原本運算結果是實數1.25，但指派給整數，所以會是1
}void loop() {}
```

■整數除法與取餘的應用

整數除法與取餘可以將一個整數分解爲數個數字。例如，若要使用七段顯示器顯示 152，那就要將此數字分解爲 1, 5, 2 三個數字，其方法如下：

```
int a=152;
int a1=152/100;//1百位數
int a2=(152-a1*100)/10;//5十位數
int a3=a %10;//個位數
```

■取餘

若要讓一個數字在累加的過程中，保持一定的循環，那也是用取餘。例如，

```
int i;
void loop() {
    i=(i+1) %4;
}
```

那 i 就永遠在 0, 1, 2, 3 循環。請特別留意，數學有

```
19/10=1…9
```

的書寫習慣，但電腦並沒有此運算方式，因此以上運算方式，一定要先取餘數，再求整數商如下：

```
int a=19,b=10;
int c=a%b;
int d=a/b;
```

順序錯也不行，以下程式先除再取餘，結果就不對。

```
int a=19,b=10;
int d=a/b;
int c=a%b;
```

運算子的優先順序（Precedence）

同一敘述，若同時含有多個運算子，此時即需定義運算子的優先順序。例如，

```
a=3+4*2;
```

『=』優先順序是 12，『+』優先順序是 5 ，『*』優先順序是 4，所以運算順序是

```
(a=(3+(4*2)));
```

運算子的結合律（Associativity）

當同一敘述，相鄰的運算子擁有相同優先順序的運算子，此時即需定義運算子是左結合或右結合。例如：

```
x=a-b-c;
```

連續兩個減號『-』，優先順序相同，此時就要靠定義結合律，減法結合律是由左至右，所以以上同義於

```
x=((a-b)-c);
```

而

```
x=y=z=2;
```

連續三個指派運算子『=』，指派運算子的結合律是由右至左，所以以上式子同義於

```
(x=(y=(z=2)));
```

所以，以上敘述，x、y、z 的結果都是 2。

範例 3a

請寫一個程式，可以指派長方形的長與寬，並計算周長與面積。

題目分析

1. 需要兩個變數儲存長與寬。請選擇此資料來源是整數還是浮點數，若是整數，還要分是 8 位元的 byte (0 ～ 255)、16 位元的 int (-32768 ～ 32767) 或 32 位元的 long (-2147483648 ～ 2147483647)。本書整數若沒特別註明，就一律折衷，通通取 int。
2. 需要兩個變數儲存面積與周長。請選擇整數或是浮點數，本例也是選擇 int。
3. 先使用變數指派方式，指派變數的值，程式如下。這樣可以省略冗長的變數輸入，先專注於演算法的實現。

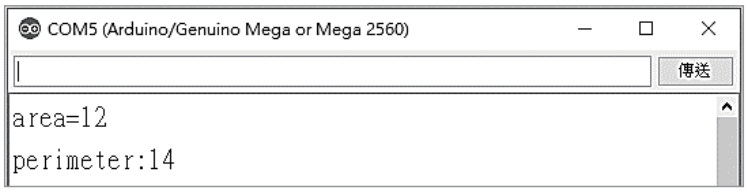
```
void setup() {
    Serial.begin(9600);
    int a,b,area,perimeter;
    a=3;
    b=4;
    area= a*b;
    perimeter=2*(a+b);
    Serial.print("area=");
    Serial.println(area);
    Serial.print("perimeter:");
    Serial.println(perimeter);
}void loop() {}
```

4. 請特別留意

```
area= ab;
perimeter=2(a+b);
```

爲數學語言，不是電腦語言，以上寫法電腦看不懂。

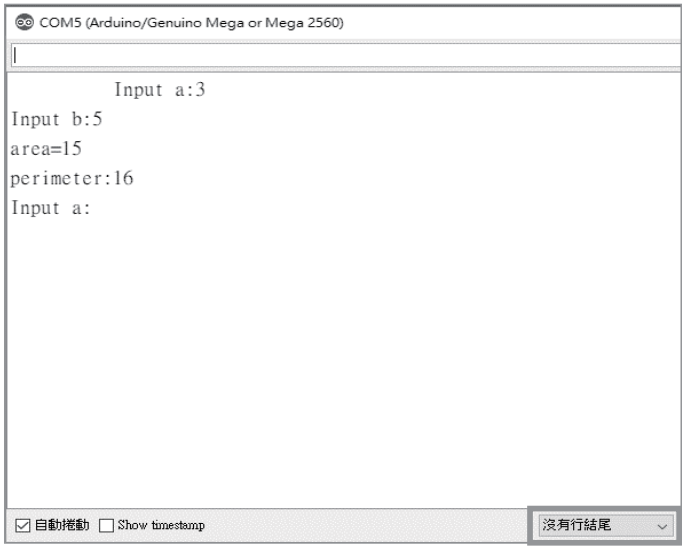
5. 以上程式執行結果如下：



6. 使用電腦的鍵盤輸入變數的數值，程式如下：

```
void setup() {
    Serial.begin(9600);
}
void loop() {
    int a,b,area,perimeter;
    Serial.print("Input a:");
    while(Serial.available() ==0) {}//等待使用者輸入資料
    a=Serial.parseInt();
    Serial.println(a);
    Serial.print("Input b:");
    while(Serial.available() ==0) {}//等待使用者輸入資料
    b=Serial.parseInt();
    Serial.println(b);
    area= a*b;
    perimeter=2*(a+b);
    Serial.print("area=");
    Serial.println(area);
    Serial.print("perimeter:");
    Serial.println(perimeter);
  }
```

6. 以上程式執行結果如下:(請將輸入方式點選『沒有行結尾』，如下圖，不然無法輸入第二個變數)



自我練習

1. 指派長方體的長、寬、高，計算其表面積與體積。
2. 請寫一個程式，可以指派一個台斤數，且轉爲公斤數輸出。
3. 請寫一個程式，可以指派一個 0 ～ 86399 的整數，且轉爲『時：分：秒』的格式輸出。
4. 請寫一個程式，可以指派一個 4 位數，並從千位數一一輸出、且求其數字和。例如，指派 1234，那就輸出 1,2,3,4 與 10。
5. 請寫一個程式，可以指派一個 4 位數，並從個位數一一輸出、且求其數字和。例如，指派 1234，那就輸出 4,3,2,1 與 10。

6. 請寫一程式，滿足以下條件。可以指派兩個座標。(x1=0;y1=3;x2=4;y2=0)，然後計算此兩點座標距離，輸出距離。

   提示：已知兩點座標分別是 (x1,y1),(x2,y2)，則其距離的公式的數學語言是：

$$d=\sqrt{(x_1-x_2)^2+(y_1-y_2)^2}$$

7. 指派三角形三邊長 a、b、c，求其面積。

   提示：先指派 a,b,c，然後計算 d=(a+b+c)/2，則三角形面積 s $=\sqrt{d(d-a)(d-b)(d-c)}$ ，輸出 s。本例假設所輸入的三角形三邊長可圍成三角形，例如指派 a=3;b=4;c=5 則得三角形面積 6，但並不是任意三條線都可圍成三角形，若要判斷是否可圍成三角形，請繼續研讀下一單元。

第4單元

霹靂燈

各種進位制

我們人類習慣使用 10 進制，逢 10 填 0 進 1，例如，$(242)_{10}$ 是表示 $2*10^2 + 4*10^1 + 2*10^0 = 242$；若是 8 進位，那就是逢 8 填 0 進 1，僅用 0, 1, 2, 3, 4, 5, 6, 7 等 8 個數字，所以 $(11)_8 = 1*8^1 + 1*8^0 = (9)_{10}$；若是 2 進位，那就是逢 2 填 0 進 1，僅用 0, 1 兩個數字，所以 $1011 = 1*2^3 + 0*2^2 + 1*2^1 + 1*2^0 = 11$；若是 16 進位，那就用 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F 表示 0 到 15，且逢 16 填 0 進 1，所以 $(A2E)_{16} = 10*16^2 + 2*16^1 + 14*16^0 = (302)_{10}$。

Arduino 語言可以處理的整數有四種進位方式，分別是十進位（Decimal）、二進位（Binary）、八進位（Octal）及十六進位（Hexadecimal）。其中十進位則以我們平常書寫數字的方式即可，例如 12；二進位則以 B 開頭，例如，B11 則代表十進位的 3；八進位則應以 0 開頭，例如 011 代表十進位的 9(1*8+1)；十六進位應以 0x 開頭，例如，0x11 代表十進位的 17(1*16+1)。請鍵入以下程式，並觀察執行結果。

```
void setup() {
    Serial.begin(9600);
    int a=242;
    int b=B11;
    int c=011;//零，不是字母O
    int d=0x11;//零x，不是字母OX
    Serial.println(a);
    Serial.println(b);
    Serial.println(c);
    Serial.println(d);
}void loop() {}
```

2進位與16進位

前面資料的數位化已經介紹，電腦是以 2 進位儲存數值，所以若以 8 位元儲存一個正整數，那

```
5
```

就會以

```
00000101
```

表示，我們若以 LED 觀察結果就是『滅滅滅滅滅亮滅亮』，那反過來說，若有 8 顆 LED 呈現

```
滅滅滅滅滅亮滅亮
```

要將此現象數位化，我們也可用 2 進位表示爲

```
B00000101
```

其次，以上 2 進位有點長，所以我們習慣以 16 進位表示爲

```
0x05
```

也就是在自動控制的領域裡，我們會習慣以 2 進位或 16 進位來表示一些燈號或控制結果，請讀者要慢慢習慣這種 2 或 16 進位表示方式。以下是一些常用數字的 16 進位書寫表示方式。

10進位	2進位	16進位	10進位	2進位	16進位
0	B0	0x0	11	B1011	0xb
1	B1	0x1	12	B1100	0xc
2	B10	0x2	13	B1101	0xd
3	B11	0x3	14	B1110	0xd
4	B100	0x4	15	B1111	0xe
5	B101	0x5	16	B10000	0x10
6	B110	0x6	17	B10001	0x11
7	B111	0x7	18	B10010	0x12
8	B1000	0x8	127	B01111111	0x7F
9	B1001	0x9	254	B11111110	0xFE
10	B1010	0xa	255	B11111111	0xFF

自我練習

1. 若有 8 個燈號連續排列，且其燈號是『滅亮亮滅亮亮亮亮』，請問該如何以 16 進制數字回報。

時序圖

以上是一個瞬間結果的資料數位化，但有很多情況，例如，廣告燈、耶誕樹燈、紅綠燈等，還會隨著時間的移動有不同的變化，此時可以以時間爲軸，取一段時間爲單位，將這些單位的結果記錄下來，這些以時間爲橫軸的連續資料數位化，稱爲時序圖，程式設計師就按照此時序圖，將結果以陣列存放，再依序輸出此陣列，即可完成指定動作。

範例 4a 耶誕樹燈。

小明買了一顆耶誕樹，現在要將它裝上 8 顆 LED，且讓其閃爍如下，請問如何規劃。

時序	LED7	LED6	LED5	LED4	LED3	LED2	LED1	LED0
1	○	○	○	○	○	○	○	●
2	○	○	○	○	○	○	●	●
3	○	○	○	○	○	●	●	●
4	○	○	○	○	●	●	●	●
5	○	○	○	●	●	●	●	●
6	○	○	●	●	●	●	●	●
7	○	●	●	●	●	●	●	●
8	●	●	●	●	●	●	●	●

（實心代表燈亮，空心代表不亮）

操作步驟

1. 將電路接線如同範例 2a。
2. 以秒爲單位，規劃時序圖。本例規劃如下，1 代表亮，0 代表不亮：

時序	LED7	LED6	LED5	LED4	LED3	LED2	LED1	LED0	值
0	0	0	0	0	0	0	0	1	0x01
1	0	0	0	0	0	0	1	1	0x03
2	0	0	0	0	0	1	1	1	0x07
3	0	0	0	0	1	1	1	1	0x0F
4	0	0	0	1	1	1	1	1	0x1F
5	0	0	1	1	1	1	1	1	0x3F
6	0	1	1	1	1	1	1	1	0x7F
7	1	1	1	1	1	1	1	1	0xFF

3. 一一將以上資料數位化，得到 0x01, 0x03, 0x07, 0x0F, 0x1F, 0x3F, 0x7F, 0xFF，如上表最右一欄。
4. 使用變數儲存以上資料。本例使用 a0, a1, a2, a3,a4, a5, a6, a7 表示如下：

```
a0=0x01;
a1=0x03;
a2=0x07;
a3=0x0F;
a4=0x1F;
a5=0x3F;
a6=0x7F;
a7=0xFF;
```

3. 將以上資料寫入 Arduino，程式如下：

```
void setup() {
   DDRB=B11111111;
   //Serial.begin(9600);
   DDRC=0xFF;PORTC=0xFE;//0xFE is B11111110 將點陣LED當作8個LED使用
}
void loop() {
  byte a0,a1,a2,a3,a4,a5,a6,a7;
  //宣告a0,a1...是8位元正整數，可表示0~255
  a0=0x01;
  PORTB=a0;
  delay(1000);
  a1=0x03;
  PORTB=a1;
  delay(1000);
  a2=0x07;
  PORTB=a2;
  delay(1000);
  a3=0x0F;
  PORTB=a3;
  delay(1000);
  a4=0x1F;
  PORTB=a4;
  delay(1000);
```

```
  a5=0x3F;
  PORTB=a5;
  delay(1000);
  a6=0x7F;
  PORTB=a6;
  delay(1000);
  a7=0xFF;
  PORTB=a7;
  delay(1000);
}
```

霹靂燈與陣列

前面的霹靂燈共 8 個時序，我們用了 8 個變數表示 8 個顯示方式，這樣程式寫起來，眞是又臭又長，那如果遇到更複雜的變化呢？本單元我們將要介紹一個很好用的工具，那就是陣列，陣列就是要解決儲存連續相同的大批資料。例如，上一範例，我們一共 8 筆連續資料，如果用陣列結構，就可以宣告一個一維陣列如下：

```
byte a[8];
```

共可儲存 8 筆資料，分別是 a[0], a[1], a[2], a[3], a[4], a[5], a[6], a[7]，中括號內的數字稱爲索引。索引可用來指派資料來源，就如同我們常用座號來指派同學一樣，只是人類習慣從 1 號開始，但電腦因爲了方便配合取餘運算與重複循環控制，所以陣列從 0 號開始。但是 0 號可用，也可不用，只是重複控制從 0 號開始，若配合取餘『%』會較方便。上一範例的資料設定，若以陣列表示，則陣列指派如下：

```
a[0]=0x01;
a[1]=0x03;
a[2]=0x07;
a[3]=0x0F;
a[4]=0x1F;
```

```
a[5]=0x3F;
a[6]=0x7F;
a[7]=0xFF;
```

以上宣告陣列與陣列初值指派兩個步驟，亦可宣告陣列的同時，就指派其值如下：

```
byte a[]={0x01,0x03,0x07,0x0F,0x1F,0x3F,0x7F,0xFF};
```

有了陣列，往後就可以使用迴圈與陣列索引存取陣列的值，程式如下：

```
void setup() {
   DDRB=B11111111;
   DDRC=0xFF;PORTC=0xFE;//0xFE is B11111110 將8*8點陣LED當作8個LED使用
   Serial.begin(9600);
}
void loop() {
  byte a[]={0x01,0x03,0x07,0x0F,0x1F,0x3F,0x7F,0xFF};
  for (int i=0 ;i<=7;i++){
    PORTB=a[i];
    delay(1000);
  }
  delay(3000);
}
```

也可以使用以下無窮迴圈

```
void setup() {
   DDRF=B11111111;
   Serial.begin(9600);
   DDRC=0xFF;PORTC=0xFE;//0xFE is B11111110 將8*8點陣LED當作8個LED使用
}
byte i=0;
void loop() {
  byte a[]={0x01,0x03,0x07,0x0F,0x1F,0x3F,0x7F,0xFF};
  PORTBF=a[i];
  delay(1000);
  i=(i+1)%8;//i每次遞增1，但%8可保障數字i在0～7之間
}
```

本例的

```
byte i=0;
```

要放在 void loop() {} 外面，此稱爲全域變數，請移至 loop() 裡面，並觀察結果。

自我練習

1. 若希望霹靂燈的變化如下，請寫程式完成。

時序	LED7	LED6	LED5	LED4	LED3	LED2	LED1	LED0	值
0	1	0	0	0	0	0	0	1	
1	0	1	0	0	0	0	1	0	
2	0	0	1	0	0	1	0	0	
3	0	0	0	1	1	0	0	0	
4	0	0	0	1	1	0	0	0	
5	0	0	1	1	1	1	0	0	
6	0	1	1	1	1	1	1	0	
7	0	1	1	1	1	1	1	1	
8	1	1	1	1	1	1	1	1	
9	1	1	1	1	1	1	1	0	
10	1	1	1	1	1	1	0	0	
11	1	1	1	1	1	0	0	0	
12	1	1	1	1	0	0	0	0	
13	1	1	1	0	0	0	0	0	
14	1	1	0	0	0	0	0	0	
15	1	0	0	0	0	0	0	0	
16	0	0	0	0	0	0	0	0	

2. 紅綠燈。若有一個單向紅綠燈時序如下，請寫程式完成。

時序	LED2（紅燈）	LED1（黃燈）	LED0(綠燈)	值
0	0	0	1	
1	0	0	1	
2	0	0	1	
3	0	0	1	
4	0	1	0	
5	0	1	0	
6	1	0	0	
7	1	0	0	
8	1	0	0	

3. 請自行觀察路口的雙向紅綠燈，並寫程式完成。

   提示：將所有燈號建立時序圖。

時序	LED5紅	LED4黃	LED3綠	LED2紅	LED1黃	LED0綠	值
0	1	0	0	0	0	1	
1	1	0	0	0	0	1	
2	1	0	0	0	0	1	
3	1	0	0	0	0	1	
4	1	0	0	0	1	0	
5	1	0	0	1	0	0	
6	0	0	1	1	0	0	
7	0	0	1	1	0	0	
8	0	0	1	1	0	0	
9	0	0	1	1	0	0	
10	0	0	1	1	0	0	
11	0	0	1	1	0	0	
12	0	0	1	1	0	0	
13	0	1	0	1	0	0	
14	1	0	0	1	0	0	

4. 高鐵停靠站燈號。高鐵的月台上，都有一個告示板，用來顯示即將到站列車的停靠站。假設以下是南港月台的列車停靠方式，1 代表有停靠，0 代表未停靠，請問您如何來設計。

列車編號	南港	台北	板橋	桃園	新竹	苗栗	台中	彰化	嘉義	台南	高雄	PORTK	PORTF
	PORTK			PORTF									
0	1	1	0	0	0	0	1	0	0	1	1	0x6	0x13
1	1	1	1	1	1	1	1	1	1	1	1	0x7	0xff
2	1	1	0	0	1	0	1	0	1	1	1	0x6	0x57
3	1	1	0	0	1	0	1	0	0	0	1	0x6	0x51

5. 捷運到站燈號。搭乘捷運時，車上都有列車停靠站顯示說明。例如，淡水線的站名如下，若要用 LED 顯示到站資訊，請問您如何設計。

時序	淡水	紅樹林	竹圍	關渡	忠義	復興崗	北投	奇岩	其哩岸	石牌	明德	芝山	士林	劍潭	圓山	民權西路	雙連	中山	台北車站	台大醫院	中正紀念堂	東門	大安	安和	101	象山
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
PORTA							PORTB							PORTC							PORTF					

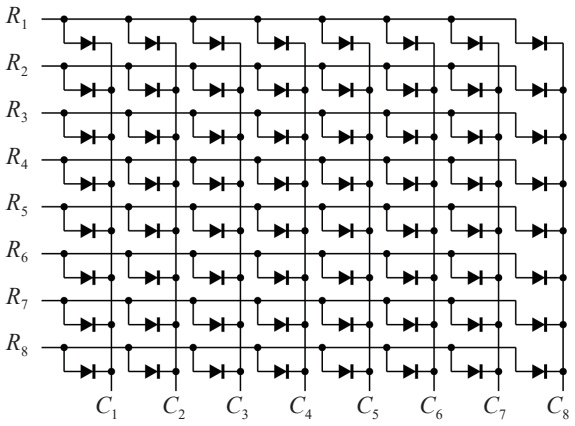
6. 教學機。本單元的 LED 您可以拿來作爲順序教學的指示燈。例如，霓虹燈、鉤毛線順序、電子琴、跳舞機、CPR 急救順序、葉問拳法練習樁等，也就是您可以用 LED 來提醒使用者要按哪裡、踩哪裡、打哪裡等等等。請自己思考身邊的應用，寫程式完成。

電子琴教學機：https://youtu.be/CsvtgIQ7Vx0

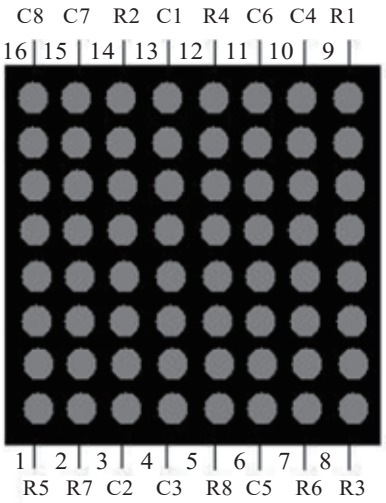
第5單元

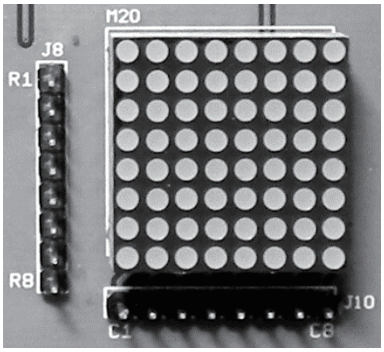
8*8點陣LED

8*8 點陣 LED 分爲共陰極與共陽極，因爲 Arduino 高電位輸出電流有 20mA，已經可以直接驅動 LED，所以本書採用共陰極，這樣電路才會簡單。共陰極表示每一行 (R1,R2..R8) 的陰極都共用輸出腳位，如下圖的 C1,C2..C8。（本書使用行優先，因爲有些國家以列優先，所以到材料行買零件，請由下圖確認共陰或共陽）



不論是共陽極或共陰極其腳位排列都相同，如下圖左（LED 朝上），廠商通常將型號印在下面，所以請將有型號的那一面朝下，本書以 R5 爲腳位 1，逆時針繼續編號 (此與 IC 腳位編號相同)，例如腳位 9 是 R1。





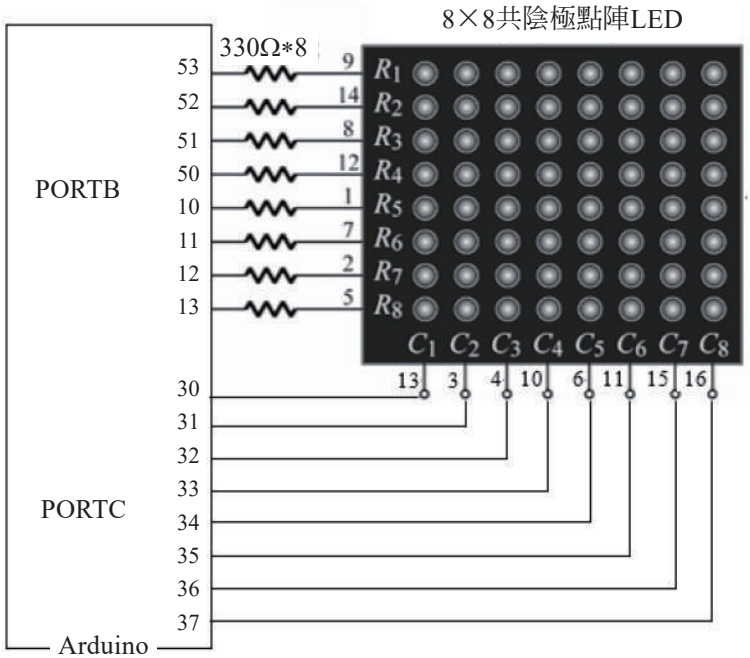
行(Column)接腳：C1, C2, C3 ... C8

列(Row)接腳：R1, R2, R3 ... R8

請自行拿傳統型指針式三用電表驗證（請撥 R×10，小型數位電錶不行）。例如，共陰極的測量如下:正極（黑棒）接 R1，負極（紅棒）接 C1，則第 1 列第 1 行將亮；正極接 R1，負極接 C2，則第 1 列第 2 行將亮（補充說明 : 三用電表電池正極是用黑棒拉出。）。其次，亦可用本書附錄 A 自製的電路連通測試計，正極接 a，負極接 com1，那千位數的 a 棒亮。

點陣LED驅動電路

現在請將點陣 LED 接線如下：



硬體測試

請鍵入以下程式，觀察 LED 有沒有一排一排輪流全亮。

```
byte c[]={0x7f,0xbf,0xdf,0xef,0xf7,0xfb,0xfd,0xfe};
void setup() {
    DDRB=0xFF;DDRC=0xFF;
}
void loop() {
    for (int i=0;i<=7;i++){
        PORTC=c[i];//位址
        PORTB=0xff;//資料
        delay(500); //請修改爲1，並比較其效果
    }
}
```

請將 delay(500); ，修改爲 delay(1);，並觀察所有 LED 有沒有全亮。以上程式的 c[] 陣列如下：

```
byte c[]={0x7f,0xbf,0xdf,0xef,0xf7,0xfb,0xfd,0xfe};
```

時序圖如下表：

顯示位置	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	值
C1	0	1	1	1	1	1	1	1	0x7f
C2	1	0	1	1	1	1	1	1	0xbf
C3	1	1	0	1	1	1	1	1	0xdf
C4	1	1	1	0	1	1	1	1	0xef
C5	1	1	1	1	0	1	1	1	0xf7
C6	1	1	1	1	1	0	1	1	0xfb
C7	1	1	1	1	1	1	0	1	0xfd
C8	1	1	1	1	1	1	1	0	0xfe

以上 c[] 陣列稱爲位址，主要是讓 C1..C8 輪流接地，這樣資料就可以輪流傳送到 C1..C8，只要速度夠快，因爲人類眼睛有視覺暫留現象，看起來就會一起亮，這樣就可以顯示任何文字與影像。例如，以下程式，可讓左邊 C1 第一行 8 個燈全亮，

```
PORTC=0x7f; //位址
PORTB=0xff; //資料
```

以下程式，將會使點陣 LED 左邊 C1 最上面 1 個 LED 亮。

```
PORTC=0x7f; //位址
PORTB=0x01; //資料
```

以下程式，將會使點陣 LED 左邊 C2 的上面兩個 LED 全亮。

```
PORTC =0xbf;
PORTB =0x3;
```

文字數位化

若要將文數字顯示在此 8*8 的點陣 LED，也是要將此文字數位化。將此文數字資料數位化的步驟如下：將此文數字寫在以下方格紙，並計算每行 (Column) 的值。以下是將『洪』寫在方格紙上，則 C1 值二進位是 B10010001（高位元在下面），以 16 進位表示是 0x91、C2 值是 0x4A，依此類推。

	C1	C2	C3	C4	C5	C6	C7	C8
R1	1	0	0	0	1	0	1	0
R2	0	1	0	1	1	1	1	1
R3	0	0	0	0	1	0	1	0
R4	0	1	0	0	1	0	1	0
R5	1	0	0	1	1	1	1	1
R6	0	0	0	0	1	1	0	0
R7	0	1	0	1	0	0	1	0
R8	1	0	1	0	0	0	0	1
值	0x91	0x4A	0x80	0x52	0x3F	0x32	0x5F	0x92

將以上每一行的值以陣列儲存如下：（影像變識要將文字數位化，也是此相同的原理）

```
byte d[]={0x91,0x4a,0x80,0x52,0x3f,0x32,0x5f,0x92};
```

然後快速依序傳送到對應的行，例如，0x91 送到 C1、0x4A 送到 C2…我們稱此爲掃描輸出，即可顯示『洪』，程式如下：

```
byte c[]={0x7f,0xbf,0xdf,0xef,0xf7,0xfb,0xfd,0xfe};
byte d[]={0x91,0x4a,0x80,0x52,0x3f,0x32,0x5f,0x92};
void setup() {
    DDRB=0xFF;DDRC=0xFF;
}
void loop() {
    for (int i=0 ;i<=7;i++){
        PORTC=c[i];//位址
        PORTB=d[i];//資料
        delay(1);//delay(500)
    }
}
```

自我練習

1. 請自行找一個筆畫較少的中文字，以能填入 8*8 方格爲原則，計算其陣列值，並顯示在此 8*8 LED 上。

	C1	C2	C3	C4	C5	C6	C7	C8
R1								
R2								
R3								
R4								
R5								
R6								
R7								
R8								
值								

第6單元

跑馬燈與告白板

跑馬燈

上一單元，我們先顯示一個字，若要顯示的文字長度超過字幕機的寬度，就要使用跑馬燈的原理，請鍵入以下程式，並觀察執行結果。

```
byte c[]={0x7f,0xbf,0xdf,0xef,0xf7,0xfb,0xfd,0xfe};
byte d[]={0,0,0,0,0,0,0,0,0,0x91,0x4a,0x80,0x52,0x3f,0x32,
0x5f,0x92,0,0,0x47,0x45,0x45,0x39,0,0,0,0x7c,0x12,0x11,0x12,
0x7c,0,0,0,0,0,0,0,0,0,0};
int slen= 41;
byte a[8];
void setup() {
  DDRB=0xFF;
  DDRC=0xFF;
}
void loop() {
  //以跑馬燈左旋顯示文字
    for (int k=0 ;k<=slen-8;k++){
        for (int i=0;i<=7;i++){//依序每次抓8個
          a[i]=d[i+k] ;
        }
        for (int i=0;i<=50;i++){//停留時間
          for (int j=0 ;j<=7;j++){
                  PORTC=c[j];//位址
                  PORTB=a[j];//資料
                  delay(1);
              }
       }
  }
}
```

本例以跑馬燈顯示『洪 5A』，先將『洪 5A』數位化，計算字型如下：

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
R1	1	0	0	0	1	0	1	0	0	0	1	1	1	1	0	0	0	0	0	1	0	0	0	0
R2	0	1	0	1	1	1	1	1	0	0	1	0	0	0	0	0	0	0	1	0	1	0	0	0
R3	0	0	0	0	1	0	1	0	0	0	1	1	1	0	0	0	0	1	0	0	0	1	0	0
R4	0	1	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0
R5	1	0	0	1	1	1	1	1	0	0	0	0	0	1	0	0	0	1	1	1	1	1	0	0
R6	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0
R7	0	1	0	1	0	0	1	0	0	0	1	1	1	0	0	0	0	1	0	0	0	1	0	0
R8	1	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x	91	4a	80	52	3f	32	5f	92	0	0	47	45	45	39	0	0	0	7c	12	11	12	7c	0	0

將以上資料以陣列儲存如下，文字前後都要加上空白 0，這樣使用者才來得及看。slen 是輸出文字的行數，也是 d 陣列的長度

```
byte d[]={0,0,0,0,0,0,0,0,0,0x91,0x4a,0x80,0x52,0x3f,0x32,
0x5f,0x92,0,0,0x47,0x45,0x45,0x39,0,0,0,0x7c,0x12,0x11,0x12,
0x7c,0,0,0,0,0,0,0,0,0};
int slen=41;
```

現在逐一將 d[] 陣列複製 8 筆資料到 a[] 陣列，

```
for (int i=0;i<=7;i++){//依序每次抓8個
          a[i]=d[i+k] ;
        }
```

第 1 次抓到 0,0,0,0,0,0,0,0; 第 2 次抓到 0,0,0,0,0,0,0,0x91; 第 3 次抓到 0,0,0,0,0,0,102,153…依此類推，再將 a[] 陣列，同上範例掃瞄輸出，但電腦眞的太快了，每一次還要重複一點時間，如以上程式的 i 迴圈，請自行調整 i 迴圈的值，即可調整跑馬燈輪轉速度。

告白板

以上是跑馬燈的輸出，輸出效果比較平淡，適用於一般公告，若要讓文字逐一閃爍顯示，那會比較霸氣，這樣可用在告白板、升官、競賽、考試榜單等的顯示。例如，以下程式逐一顯示 8,2,5,7,1,0，每 1 秒自動變換 1 個數字。

```
byte i=0;
byte c[]={0x7f,0xbf,0xdf,0xef,0xf7,0xfb,0xfd,0xfe};//位址
byte d[8];
byte e[6][8]={ { 102, 153, 137, 150, 96, 0, 0, 0},
               { 130, 193, 161, 145, 206, 0, 0, 0},
               { 132, 135, 137, 113, 0, 0, 0, 0},
               { 2, 1, 225, 25, 7, 0, 0, 0},
               { 0, 130, 255, 128, 0, 0, 0, 0},
               { 126, 129, 129, 129, 126, 0, 0, 0}};
long t,t1,t2;
void setup() {
    DDRB=0xFF;
    DDRC=0xFF;
}
void loop() {
    //放字型
    for (int j=0 ;j<=7;j++)
        d[j]=e[i][j];
    //掃描輸出
    t1= millis();
    do{
        for (int j=0 ;j<=7;j++){
            PORTC=c[j];//位址
            PORTB=d[j];//資料
            delay(1);
        }
        t2=millis();
        t=t2-t1;
    }while( (t<1000)  );//1秒，根據需求調整
    i=(i+1) % 6;//本例6個字
}
```

那要如何思考？以下是我的思考步驟：

1. 依序將以上數字 8,2,5,7,1,0 數位化，每組 8 個 byte，如以下 d[] 陣列。
2. 可用一維陣列儲存，那此一維陣列長度爲 48，如下：

```
byte d[]= { 102, 153, 137, 150, 96, 0, 0, 0,
            130, 193, 161, 145, 206, 0, 0, 0,
            132, 135, 137, 113, 0, 0, 0, 0,
            2, 1, 225, 25, 7, 0, 0, 0,
            0, 130, 255, 128, 0, 0, 0, 0,
            126, 129, 129, 129, 126, 0, 0, 0};
```

3. 也可用二維陣列儲存此 6 組輸出碼如下：

```
byte e[6][8]={ { 102, 153, 137, 150, 96, 0, 0, 0},
               { 130, 193, 161, 145, 206, 0, 0, 0},
               { 132, 135, 137, 113, 0, 0, 0, 0},
               { 2, 1, 225, 25, 7, 0, 0, 0},
               { 0, 130, 255, 128, 0, 0, 0, 0},
               { 126, 129, 129, 129, 126, 0, 0, 0}};
```

4. 可用時間、也可用按鈕變換要輸出的數字。
5. 不論是一維陣列或二維陣列，每次都要取 8 個 byte 進行掃描輸出，二維陣列的寫法，如以上程式。
6. 方法二：將 0 到 9 通通依序計算其輸出碼，通通放到陣列，然後將要顯示的文字以字串儲存，每次抓一個字元，且到陣列找對的輸出碼，並輸出此輸出碼，程式如下：

```
byte i=0;
byte c[]={0x7f,0xbf,0xdf,0xef,0xf7,0xfb,0xfd,0xfe};//位址
byte d[8];
byte e[10][8]={{ 126, 129, 129, 129, 126, 0, 0, 0},
               { 0, 130, 255, 128, 0, 0, 0, 0},
               { 130, 193, 161, 145, 206, 0, 0, 0},
               { 130, 137, 137, 118, 0, 0, 0, 0},
```

```
                { 48, 40, 38, 255, 32, 0, 0, 0},
                { 132, 135, 137, 113, 0, 0, 0, 0},
                { 124, 146, 137, 137, 112, 0, 0, 0},
                { 2, 1, 225, 25, 7, 0, 0, 0},
                { 102, 153, 137, 150, 96, 0, 0, 0},
                { 14, 145, 145, 81, 62, 0, 0, 0}};
long t,t1,t2;
String f="825710";
void setup() {
    DDRB=0xFF;
    DDRC=0xFF;
    Serial.begin(9600);
}
void loop() {
    //放字型
    byte g=byte(f[i]) ; //取到的字元，是其ASCII碼，所以要減48
    Serial.println(g);
    g=g-48;
    Serial.println(g);
    for (int j=0 ;j<=7;j++)
        d[j]=e[g][j];//到e陣列取到對應的輸出碼
    //掃描輸出
    t1= millis();
    do{
        for (int j=0 ;j<=7;j++){
            PORTC=c[j];//位址
            PORTB=d[j];//資料
            delay(1);
        }
        t2=millis();
        t=t2-t1;
    }while( (t<1000)  );//1秒，根據需求調整
    i=(i+1) % 6;//本例6個字
}
```

7. 方法三，將 Ascii 從 32 到 122 一一計算其輸出碼，這樣包含了所有空白、數字、大小寫英文字元。此一計算有點多了，可以用人工算，也可以用電腦算，用電腦算的 VB 程式如下：（更多的 VB 程式，或想要瞭解此 VB 程式的撰寫，請看拙作

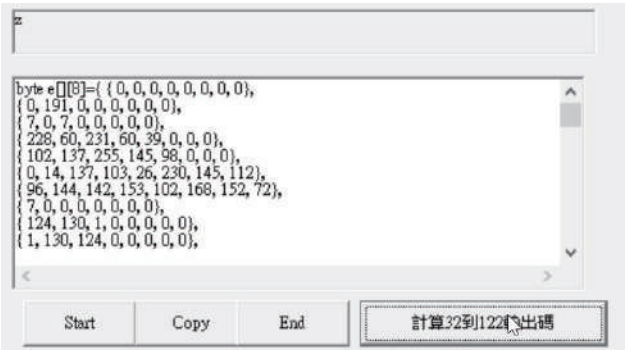
『高中生活科技實作 - 使用 Arduino』或『Ardunio 字幕機自造與程式設計』）

```
Private Sub Command1_Click()
    lf = Chr(13) + Chr(10) '跳列
    '請先安排一個Picturebox,修改Name爲 pic
    pic.ScaleMode = 3 '以像素爲座標單位
    pic.FontSize = 8
    pic.CurrentX = 0: pic.CurrentY = 0 '輸出位置

    's = "byte c[]={0x7f,0xbf,0xdf,0xef,0xf7,0xfb,0xfd,
                    0xfe};" + lf
    s = s + "byte e[][8]={ "
    '當場計算e[][]
    For k = 32 To 122
        s = s + "{"
        pic.Cls
        pic.Print Chr(k)
        For i = 0 To 7
            p = 0
            For j = 1 To 8
                If pic.Point(i, j) = 0 Then '黑點
                    p = p + 2 ^ (j - 1)
                End If
            Next
            If i = 7 Then
                s = s + Str(p) + "}," + lf
            Else
                s = s + Str(p) + ","
            End If
        Next

    Next
    s = s + "};"
    Text2.Text = s
End Sub
```

9. 以上程式執行結果如下圖：



10. 配合以下 Arduino 程式，此程式與方法二相同，就可顯示所有數字與大小寫字元。

```
byte i=0;
byte c[]={0x7f,0xbf,0xdf,0xef,0xf7,0xfb,0xfd,0xfe};//位址
byte d[8];
byte e[][8]={ { 0, 0, 0, 0, 0, 0, 0, 0},//32 is space
{ 0, 191, 0, 0, 0, 0, 0, 0},
{ 7, 0, 7, 0, 0, 0, 0, 0},
{ 228, 60, 231, 60, 39, 0, 0, 0},
{ 102, 137, 255, 145, 98, 0, 0, 0},
{ 0, 14, 137, 103, 26, 230, 145, 112},
{ 96, 144, 142, 153, 102, 168, 152, 72},
{ 7, 0, 0, 0, 0, 0, 0, 0},
{ 124, 130, 1, 0, 0, 0, 0, 0},
{ 1, 130, 124, 0, 0, 0, 0, 0},
{ 9, 6, 31, 6, 9, 0, 0, 0},
{ 16, 16, 254, 16, 16, 0, 0, 0},
{ 128, 128, 0, 0, 0, 0, 0, 0},
{ 32, 32, 32, 0, 0, 0, 0, 0},
{ 0, 128, 0, 0, 0, 0, 0, 0},
{ 224, 24, 7, 0, 0, 0, 0, 0},
{ 126, 129, 129, 129, 126, 0, 0, 0},
{ 0, 130, 255, 128, 0, 0, 0, 0},
{ 130, 193, 161, 145, 206, 0, 0, 0},
{ 130, 137, 137, 118, 0, 0, 0, 0},
{ 48, 40, 38, 255, 32, 0, 0, 0},
```

```
{ 132, 135, 137, 113, 0, 0, 0, 0},
{ 124, 146, 137, 137, 112, 0, 0, 0},
{ 2, 1, 225, 25, 7, 0, 0, 0},
{ 102, 153, 137, 150, 96, 0, 0, 0},
{ 14, 145, 145, 81, 62, 0, 0, 0},
{ 0, 136, 0, 0, 0, 0, 0, 0},
{ 0, 136, 136, 0, 0, 0, 0, 0},
{ 16, 40, 40, 68, 68, 0, 0, 0},
{ 72, 72, 72, 72, 72, 0, 0, 0},
{ 68, 68, 40, 40, 16, 0, 0, 0},
{ 2, 1, 177, 14, 0, 0, 0, 0},
{ 0, 60, 66, 153, 165, 157, 162, 92},
{ 128, 224, 156, 19, 156, 224, 128, 0},
{ 129, 255, 137, 137, 137, 118, 0, 0},
{ 60, 66, 129, 129, 130, 71, 0, 0},
{ 129, 255, 129, 129, 129, 66, 60, 0},
{ 129, 255, 137, 137, 157, 195, 0, 0},
{ 129, 255, 137, 9, 29, 3, 0, 0},
{ 60, 66, 129, 129, 146, 247, 16, 0},
{ 129, 255, 137, 8, 137, 255, 129, 0},
{ 129, 255, 129, 0, 0, 0, 0, 0},
{ 192, 129, 127, 1, 0, 0, 0, 0},
{ 129, 255, 137, 20, 163, 193, 128, 0},
{ 129, 255, 129, 128, 128, 192, 0, 0},
{ 129, 255, 134, 56, 192, 56, 134, 255},
{ 129, 255, 134, 24, 97, 255, 1, 0},
{ 60, 66, 129, 129, 129, 66, 60, 0},
{ 129, 255, 137, 9, 9, 6, 0, 0},
{ 60, 66, 129, 129, 129, 66, 60, 0},
{ 129, 255, 137, 25, 41, 198, 128, 0},
{ 230, 73, 145, 146, 103, 0, 0, 0},
{ 0, 3, 129, 255, 129, 3, 0, 0},
{ 1, 127, 129, 128, 129, 127, 1, 0},
{ 1, 7, 57, 192, 57, 7, 1, 0},
{ 1, 15, 56, 225, 31, 49, 192, 57},
{ 129, 195, 165, 24, 165, 195, 129, 0},
```

```
{ 1, 3, 141, 240, 141, 3, 1, 0},
{ 195, 161, 145, 137, 133, 195, 0, 0},
{ 0, 255, 1, 0, 0, 0, 0, 0},
{ 3, 28, 224, 0, 0, 0, 0, 0},
{ 1, 1, 255, 0, 0, 0, 0, 0},
{ 8, 6, 1, 6, 8, 0, 0, 0},
{ 0, 0, 0, 0, 0, 0, 0, 0},
{ 1, 2, 0, 0, 0, 0, 0, 0},
{ 80, 168, 168, 112, 128, 0, 0, 0},
{ 1, 127, 136, 136, 112, 0, 0, 0},
{ 112, 136, 136, 80, 0, 0, 0, 0},
{ 112, 136, 137, 127, 128, 0, 0, 0},
{ 112, 168, 168, 176, 0, 0, 0, 0},
{ 132, 254, 133, 1, 0, 0, 0, 0},
{ 144, 232, 168, 152, 8, 0, 0, 0},
{ 129, 255, 8, 240, 128, 0, 0, 0},
{ 136, 249, 128, 0, 0, 0, 0, 0},
{ 8, 249, 0, 0, 0, 0, 0, 0},
{ 129, 255, 32, 216, 136, 0, 0, 0},
{ 129, 255, 128, 0, 0, 0, 0, 0},
{ 8, 240, 8, 8, 240, 8, 8, 240},
{ 136, 248, 8, 240, 128, 0, 0, 0},
{ 112, 136, 136, 136, 112, 0, 0, 0},
{ 8, 248, 136, 136, 112, 0, 0, 0},
{ 112, 136, 136, 248, 8, 0, 0, 0},
{ 136, 240, 136, 0, 0, 0, 0, 0},
{ 144, 168, 168, 72, 0, 0, 0, 0},
{ 8, 126, 136, 0, 0, 0, 0, 0},
{ 8, 120, 128, 248, 128, 0, 0, 0},
{ 8, 56, 192, 56, 8, 0, 0, 0},
{ 8, 56, 192, 56, 192, 56, 8, 0},
{ 136, 216, 32, 216, 136, 0, 0, 0},
{ 8, 56, 192, 56, 8, 0, 0, 0},
{ 152, 200, 168, 152, 200, 0, 0, 0},
};
long t,t1,t2;
String f="This is a book 825710";
void setup() {
    DDRB=0xFF;
    DDRC=0xFF;
    Serial.begin(9600);
}
```

```
void loop() {
    //放字型
    byte g=byte(f[i]) ; //取到的字元，是其ASCII碼，本例從32
                          開始計算，所以要減32
    Serial.print(i);Serial.print(",");
    Serial.println(g);
    g=g-32;
    Serial.println(g);
    for (int j=0 ;j<=7;j++)
        d[j]=e[g][j];
    //掃描輸出
    t1= millis();
    do{
        for (int j=0 ;j<=7;j++){
            PORTC=c[j];//位址
            PORTB=d[j];//資料
            delay(1);
        }
        t2=millis();
        t=t2-t1;
    }while( (t<1000)  );//1秒，根據需求調整
    i=(i+1) % f.length();//長度
}
```

11. 明滅。因爲有時候連續兩個字元若相同，例如顯示『88255』，若沒有加上顯示空白碼，會變成『825』。要控制點陣 LED 明滅，一樣要掃瞄空白碼 (0x0) 一個短暫的時間，顯示空白碼的測試程式如下，請加在以上程式『掃描輸出』的後面就可以。

```
t1= millis();
    do{
        for (int j=0 ;j<=7;j++){
            PORTC=c[j];//位址
            PORTB=0;//資料滅掉
            delay(1);
        }
        t2=millis();
        t=t2-t1;
    }while( (t<200)  );//滅掉0.2秒，根據需求調整
```

自我練習

1. 同上題，那如何使用 2 個指撥開關調整文字輸出速度呢？
2. 如何使用按鈕變換數字？例如，每按一下按鈕，變換下一個文字。
3. 請寫一程式，可以每按一下按鈕，就產生 1 個 1 到 6 的亂數，且由點陣 LED 輸出且停住一直顯示。
4. 請寫一程式，使用 8 個按鈕，編號是 1 到 8，每按一個按鈕，分別顯示對應的 1 到 8，停留 1 秒就熄滅。
5. 請寫一程式，可以用 8 個指撥開關指派輸出 0 到 8 的數字。
6. 請寫一程式，可以用 3 個指撥開關指派輸出 0 到 7 的數字。
7. 學習以上程式有樂趣的同學，可使用以上原理與演算法，自製 16*16 與 16 ✳ 64 字幕機，這樣就可用跑馬燈或告白板顯示任何中文字。請參考拙作『高中生活科技實作 - 使用 Arduino』或『Ardunio 字幕機自造與程式設計』。