

十二年國民基本教育
技術型高級中等學校電機與電子群

程式設計實習進階

(全一冊)

谷町、洪國勝、孫銘宏 編著
蔡懷文、蔡懷介

谷町簡介

1. 台灣大學資訊工程博士
2. 曾經擔任大學多項程式設計競賽命題工作
(CPE、PUPC、TUPC、NCPC、ICPC 等)
3. 曾經擔任 TOI 臺灣國際資訊奧林匹亞競賽選訓
營培訓教練 (IOI 隨隊教練)

泉勝出版有限公司
www.goodbooks.com.tw

編輯大意

程式設計實習，可說資訊電子科的重中之重，不但是單晶片的基礎，也是重要的升學考科，但高一學生往往沒有認真學習，所以沒有開竅，無法享受自己設計程式的樂趣，所以本人就依照程式設計實習課綱給予加深加廣，繼續加強迴圈、陣列、結構、指標、與類別，且各章節蒐集統測與技藝競賽試題，這樣學生就很明確知道各章節的學習重點。

♂ 本書特色

一般的程式設計總複習，總是講一大堆規定、講一大堆語法，學生無法得知知識的來龍去脈，這樣當然很枯燥，所以本書還是提供完整程式，讓學生可以由完整程式學得考試的重點。

♂ 教師配套

1. 提供電腦批改程式功能：以往學生考程式設計，老師必須不斷走動幫學生看執行結果，批改成績，每考一題，每個學生就要走動看一次，全班 30 人，就是看 30 次，老師真的很累。本系統特色是，老師不用走動，老師只要勾選教材範例或自我練習題目，或依照我們給的教學方式，完成命題，學生就可以開始考試，電腦就會自動批改對或錯。
2. 程式考試時，有將學生電腦的複製 / 貼上功能取消，學生無法作弊。
3. 程式考完，每個學生程式也存在資料庫，此時可以統計每個學生程式使用的字數，若發現有學生字數相同，表示有作弊嫌疑，可以認真比對程式，揪出抄襲者。
4. 線上題庫評量系統：提供每一個學生都有帳號，且此系統所有的考試，包含程式設計題、選擇、填充的成績都在同一系統，都可以儲存考試成果，學期末老師只要指定每次考試的比重，就可以算出學期成績。

(備註 1：也可以新增欄位，填入作業與職業道德分數。備註 2：學生可使用電腦、手機考試，或輸出紙本考試)。

5. 隨時更新的題庫：作者已經將教材範例與自我練習做成題庫，也可由任課老師與作者隨時更新，老師只要勾選題目，學生即可使用電腦作答，程式設計題也是，電腦就可以批改程式或選擇與填充題。
6. 不定期提供最新 APCS 術科試題解答：APCS 題目真的難，其主要功能是選拔有天分的學生參加奧林匹亞比賽，本系統則有谷町老師主動協助，提供一些歷屆試題影音教學影片，讓有興趣的老師與學生觀摩與學習，期望挖掘更多軟體優秀人才。或者老師也可以將 APCS 試題丟給出版公司，我們會請谷町教授提供影音教學連結檔案。
7. 教師可任選泉勝圖書 5 本，放在研究室，讓優秀學生參考。

♂ 學生配套

1. 每一學生都有帳號，都可使用線上題庫評分系統

谷町、洪國勝、孫銘宏、蔡懷文、蔡懷介

2024/3/16



目錄

第1章 工場安全衛生及程式應用

- 1-1 實習工場設施環境及機具設備的認識..... 1-1
- 1-2 工業安全及衛生、消防安全的認識..... 1-1

第2章 程式架構的認識與實作

- 2-1 程式語言與演進 2-1
- 2-2 專案架構、程式架構及語言架構 2-3
- 2-3 程式架構..... 2-7
- 2-4 程式設計步驟 2-10

第3章 資料的種類與資料型態

- 3-1 資料種類..... 3-1
- 3-2 資料型態 (Data Type) 3-3

第4章 變數與常數

- 4-1 識別字與保留字 4-1
- 4-2 變數與常數宣告與應用 4-3
- 4-3 資料型態轉換 4-7
- 4-4 資料型態應用實例..... 4-9

第5章 基本輸出/輸入函式

- 5-1 printf() 5-1
- 5-2 scanf()..... 5-4
- 5-3 亂數 5-8
- 5-4 聲音的產生..... 5-10

§ 第6章 運算子及運算式

6-1 運算子	6-1
6-2 運算式	6-21
6-3 專案除錯.....	6-26

§ 第7章 流程指令

7-1 if...else	7-1
7-2 switch...case.....	7-10

§ 第8章 迴圈指令

8-1 for 迴圈	8-1
8-2 while 迴圈	8-9
8-3 巢狀迴圈.....	8-21

§ 第9章 陣列

9-1 一維陣列.....	9-1
9-2 多維陣列.....	9-12
9-3 字元陣列.....	9-18
9-4 結構	9-28

§ 第10章 指標

10-1 指標與單一變數	10-1
10-2 指標與陣列.....	10-3
10-3 指標與字串.....	10-15

§ 第11章 公用函式及自訂函式

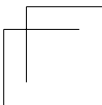
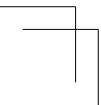
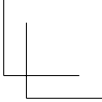
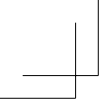
11-1 數值函式.....	11-2
11-2 字元函式.....	11-4
11-3 字串函式.....	11-9
11-4 日期與時間函式	11-18
11-5 自訂函式.....	11-18



11-6 遞迴	11-36
---------------	-------

ㄟ 第12章 物件導向程式設計

12-1 C++ 程式設計	12-3
12-2 類別	12-15
※12-3 繼承	12-31
※12-4 應用實例	12-41



工場安全衛生及程式應用

1-1 實習工場設施環境及機具設備的認識

工場安全教育是要讓從事工業相關之從業人員，透過教育訓練而具有工業安全之知識與技能，讓所有從業人員在安全的工作環境中工作，加上適當的安全防護措施，用科學與安全的方法從事工作，以防止工場意外事故之發生，減少損失。另外，實習課程的教學必須分門別類在各種不同的專門實習工場實施，在一般職業類科學校的實習工場中，主要的設施有：實習儀器設備、場地照明及通風設備、人員機具安全防護設備、及各類標示牌等等，如圖 1-1。

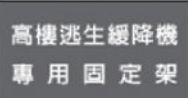
滅火器 	消防栓 	急救箱 	119 
避難方向 EXIT 	疏散方向 EXIT 	安全梯 	高樓逃生緩降機 
高樓逃生緩降機 專用固定架 	安全門 FIRE EXIT 	安全出口 EXIT 	電器 

圖 1-1 實習工場各類標示牌

1-2 工業安全及衛生、消防安全的認識

工業安全與工業衛生以及消防安全的基本目標是一樣的，皆是致力於維護工作人員的安全與健康，避免意外事故的發生，以及災害發生時的緊急應變措施，所以工業安全與工業衛生以及消防安全是先進國家非常重視的課題。工業安全是

指「透過各種安全防護措施，以避免工業災害的發生。」工業安全課程包括：意外事故與職業傷害、機具安全與維護、電器安全與維護等。工業衛生是指「分析工業環境對工作人員健康影響的一切因素，進而利用科學方法去預防和減少工作人員產生疾病和傷害。」工業衛生課程包括：危害物的管理與回收處理、急救、工作環境清潔等。我國憲法第 153 條規定：「國家為改良勞工及農民之生活，增進其生產技能，應制定保護勞工及農民之法律，實施保護勞工及農民政策。」並且明文規定「勞動基準法」為國家實現此一基本國策所制定之法律。

☞ 工業安全與衛生

☞ 工業安全及衛生之標示顏色相關規定

1. 紅色表示：危險或禁止標示。(例如：緊急控停止鈕、禁煙標示等。)
2. 橙色表示：可能發生立即災害或傷害的地方所用之標示。(例如：爆裂物、電擊傷害等。)
3. 黃色表示：注意或警告標示。(例如：易燃物、墜落、跌倒等。)
4. 綠色表示：無危險或衛生有關標示。(例如：逃生門、急救箱。)
5. 藍色表示：提醒注意標示。(例如：控制按鈕、指示燈。)
6. 白色表示：一般標示。(例如：方向指示。)
7. 紫色表示：放射性危險警告或污染標示。(例如：放射性物質現場警告標示。)
8. 黑色表示：輔助或提醒用標示。(例如：文字標語。)

☞ 工業安全及衛生之意外發生與急救處理

一般事故的發生，大多數是人為因素所造成的，只要透過教育訓練，確定每個工作人員具備安全觀念和知識，並且消除不安全之機具設備的因素與環境因素，便可以使意外事故發生減少到最低。為了防止事故的發生，必先探究事故發生的原因，一般意外事故發生的主要原因有三：

1. **人員的因素**：意外事故發生中大部分是肇因於工作人員的粗心大意、不熟悉作業流程或是精神狀態不佳所造成的。
2. **機具設備的因素**：機具設備若缺乏安全防護，或是未落實定期保養也是意外事故發生的主要因素之一。
3. **環境因素**：工場照明不良、通風不良、噪音太大、粉塵、有毒氣體、不適當的建築結構與隔間或是不整潔的廠房也是意外事故發生的主要因素之一。



另一方面，對於意外事故的發生我們除了須盡力防範之外，也必須了解事故的發生仍然無法絕對避免，因此，急救處理便是給予意外事故發生時受傷者的緊急照料，以維持傷患之生命現象，除去傷害因素，阻止傷勢惡化並給予傷患心理上之鼓舞及信心，直到受傷者獲得醫護人員的救治為止。

1. 外傷急救

- (1) 使患者平躺，並將出血部位設法抬高。
- (2) 以清潔的紗布止血，綁緊繃帶，控制出血，但不可過緊以免影響血液循環，使肌肉組織壞死。
- (3) 盡快送外科醫院診治。

2. 骨折急救

- (1) 閉鎖性骨折：斷骨端未刺穿皮膚。
 - a. 使患者平躺，盡可能將傷肢置於自然的位置。
 - b. 使用木板固定患部，並用繃帶或布條固定之。
- (2) 開放性骨折：斷骨端刺穿皮膚。
 - a. 用壓力繃帶止血。
 - b. 使患者平躺，盡可能將傷肢置於自然的位置。
 - c. 使用木板固定患部，並用繃帶或布條固定之。

3. 灼傷急救

人體的皮膚因受到熱、冷、火、化學藥品等傷害，稱之為灼傷，灼傷的急救方法應以「沖」、「脫」、「泡」、「蓋」、「送」五大步驟 施行。

- (1) 沖：以流動的冷水沖洗灼傷部位 15 ～ 30 分鐘。
- (2) 脫：於冷水中小心脫除灼傷部位的衣物。
- (3) 泡：於冷水中浸泡 15 ～ 30 分鐘，以降低傷口疼痛。
- (4) 蓋：以乾淨的毛巾蓋住傷口，防止細菌感染。
- (5) 送：快速送醫治療。

4. 大量出血急救（需注意血液循環的問題）

- (1) 直接壓迫傷口止血法。
- (2) 壓迫止血點止血法。
- (3) 止血帶止血法

5. 昏迷無心跳無呼吸急救

當傷患傷患呼吸停止無心跳時，如沒有即時急救必定死亡，因此必須立即施行心肺復甦術 (Cardio Pulmonary Resuscitation, CPR)(注意：沒有經過完整訓練並取得合格證書者，不可隨意對傷患進行 CPR 急救)，依據 2011 年我國衛生署函文說明的新版 CPR 急救法，分為六個程序，簡稱「叫叫 CABD」，如圖 1-2 所示：

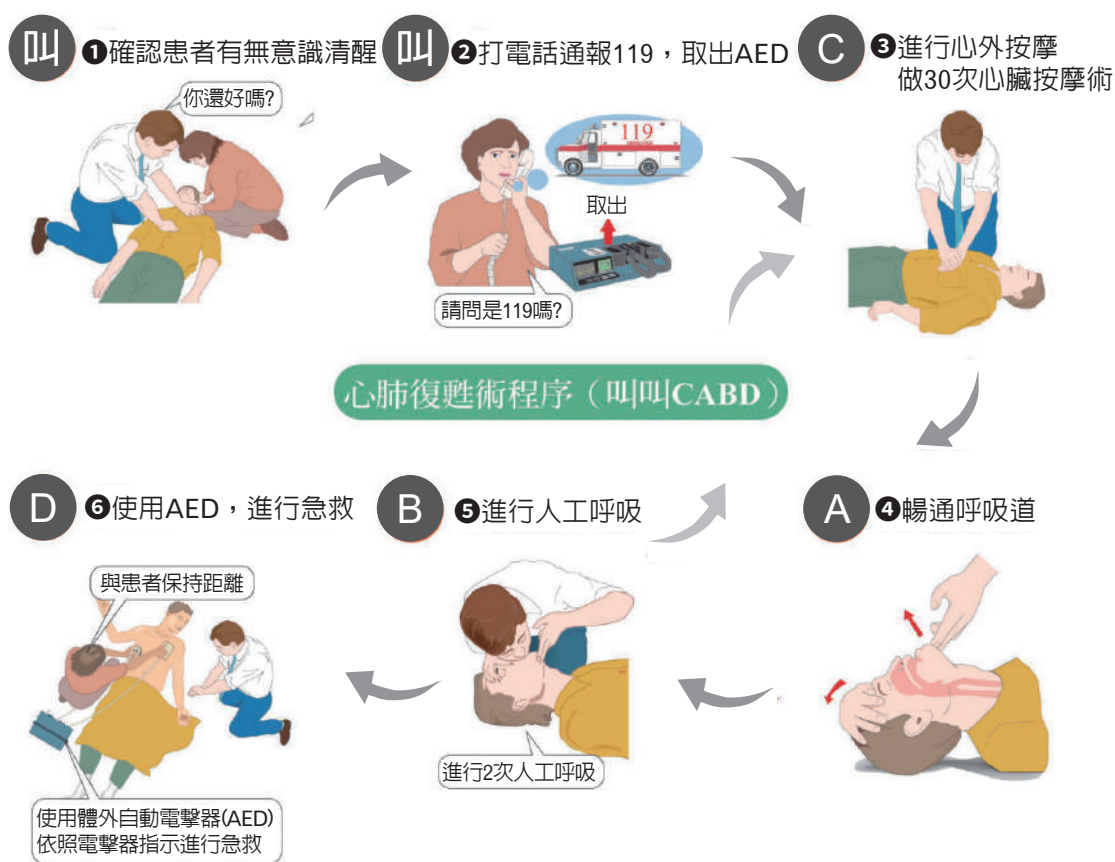


圖 1-2 心肺復甦術程序 (摘自苗栗縣政府消防局一救護 e 點靈叫叫 CABD 之急救程序)

☞ 消防安全。

火災為工業安全之意外災害中最嚴重的災害之一，它所造成的災害損失都很大，尤其是人員的傷亡。因此防火、滅火及火災的應變，為所有人員都必須學習的科目。

火災的種類依據火災燃燒物之性質可將火災分為下列四類，如表 1-1。

► 表 1-1 火災分類表

火災類別	火災名稱	燃燒物	適當滅火方法
A 類 (甲類)	一般火災	由一般可燃性固體所引起的火災。如木材、紙張、衣服、塑膠、橡膠等。	使用水或含有大量水分的溶劑滅火最有效。
B 類 (乙類)	油類火災	由可燃性液體或可燃性氣體與油脂類物質所引起的火災，如汽油、機油、燃料油、溶劑、酒精等。	此類火災禁止使用大量的水來灌注滅火，因為油浮於水面上，水的流動會助長火焰的蔓延。正確的滅火方式常用乾化學劑或二氧化碳、泡沫等方式來滅火。
C 類 (丙類)	電器火災	由電力設施、電氣設備所引起的火災，如馬達、開關設備、接線盒、變壓器等。	此類火災滅火必須使用非導電性滅火劑如乾化學劑、二氧化碳等。泡沫及水均不宜使用，因水具有導電性，可能使救火人員觸電，但是如果確定總電源切斷後，可視為甲 (A)、乙 (B) 類火災。
D 類 (丁類)	金屬火災	由易氧化之金屬所引起的火災，如鉀、鈉、鈦、鋁、鎂等。	此類火災滅火需要特種技術並使用特殊金屬化學乾粉的滅火器才能撲滅。

輔助記憶法：
 憶猶未盡 (一油電金) (一般火災、油類火災、電器火災燒焦味、金屬火災)

各類火災因燃燒物不同，可選用的滅火器對照表，如表 1-2。

► 表 1-2 各類火災滅火器對照表

適用滅火劑 火災類別	水	泡沫	二氧化碳	鹵化烷 (乾化學劑)	乾粉		
					ABC 類乾粉	BC 類乾粉	D類乾粉(特殊 金屬化學乾粉)
甲 (A) 類火災	○	○			○		
乙 (B) 類火災		○	○	○	○	○	
丙 (C) 類火災			○	○	○	○	
丁 (D) 類火災							○

☛ 滅火器的裝設規定

根據我國「各類場所消防安全設備設置標準」第 31 條，滅火器應依下列規定設置：

- (1) 設有滅火器之樓層，自樓面居室任一點至滅火器之步行距離在二十公尺以下。
- (2) 固定放置於取用方便之明顯處所，並設有以紅底白字標明滅火器字樣之標識，其每字應在二十平方公分以上。但與室內消防栓箱等設備併設於箱體內並於箱面標明滅火器字樣者，其標識顏色不在此限。
- (3) 懸掛於牆上或放置滅火器箱中之滅火器，其上端與樓地板面之距離，十八公斤以上者在一公尺以下，未滿十八公斤者在一點五公尺以下。

☛ 火災的預防

1. 去除可燃物質。
2. 防止高溫及無法散熱的問題。
3. 嚴禁工作人員吸煙。
4. 養成良好工作方法及習慣，慎選慎用高溫設備。
5. 做好電力線路及設備之檢查及維護。
6. 做好防災教育訓練。

☛ 火災的應變

火災應變能力絕不是與生俱來的，它是經過教育訓練與不斷的演習才能夠具備的一種能力。當火災發生時才能藉由正確的使用滅火裝置於第一時間滅火，並能適時的疏散逃生。另外，工場應設置消防組織，對所有從業人員實施消防訓練，其訓練內容應包括：

1. 消防常識。
2. 各種消防裝置之使用。
3. 逃生與疏散。
4. 依消防任務編組實施演習。
5. 檢討消防裝置是否足夠與汰舊換新。

☛ 滅火器與消防栓的使用

1. 滅火器的使用：滅火器的使用步驟如圖 1-3 所示：



①提起滅火器



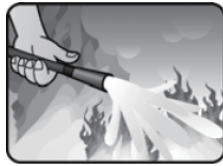
②拔出保險插梢



③將皮管朝向火點



④用力壓下把手



⑤朝火焰基部噴射



⑥左右移動噴射



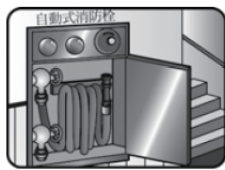
⑦用水冷卻餘燼



⑧繼續監控確保熄滅

圖 1-3 滅火器使用步驟圖

2. 消防栓的使用：消防栓的使用步驟如圖 1-4 所示：



①平時就要多注意消防栓的位置



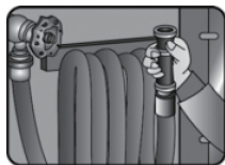
②發現警時，立即按下警報按鈕



③警示燈會閃爍並且鈴聲大作



④打開消防栓箱



⑤取出瞄子(噴嘴)



⑥取下水帶



⑦確定接頭牢固



⑧打開制水閥



⑨A轉動瞄子噴嘴，選擇水柱式射水法



⑩B轉動瞄子噴嘴，選擇水霧式射水法

◎ 水柱式射水法用來撲滅火源使用；水霧式射水法則是用來冷卻火源。



⑪因為反作用力很大所以要緊握噴嘴

圖 1-4 消防栓使用步驟圖

學後測驗

- () 1. 工場安全教育其主要且直接的目的是在 (A) 提高生產力 (B) 避免人員與設備受損 (C) 安定勞工生活 (D) 減少投資成本。
- () 2. 為了防止人員觸電所引起的傷害，下列何者錯誤？ (A) 電器設備均應有接地措施 (B) 手足潮濕，不可碰觸或操作電器設備 (C) 可以用手指測試線路或電源是否有電 (D) 危險的電力設施要有安全標誌，並有適當的限制接近設施。C
- () 3. 下列敘述中何者是正確的？ (A) 在工場中實習可以穿著寬鬆的衣服比較舒服 (B) 工場應保持通風良好，光線充足 (C) 在工場中無需保持整齊、清潔 (D) 學生在工場實習中可不必配戴個人防護器具。
- () 4. 下列對工業用標示顏色所代表之意義，何者為錯誤？ (A) 紅色表示防火設備、禁止 (B) 黃色表示注意、警告 (C) 綠色表示安全、救護設備 (D) 藍色表示放射性危險。
- () 5. 下列因素中何者不是由不安全的個人因素所造成的意外事件？ (A) 粗心 (B) 精神狀態不佳 (C) 不使用防護器具 (D) 噪音。
- () 6. 燙傷時的處理程序為 (A) 叫叫 CABD (B) 沖脫泡蓋送 (C) 牙膏與醬油塗抹 (D) 立即刺破水泡
- () 7. 下列哪一類火災是電氣火災？ (A) 甲 (A) 類 (B) 乙 (B) 類 (C) 丙 (C) 類 (D) 丁 (D) 類。
- () 8. 使用滅火器應站在何處？ (A) 上風 (B) 下風 (C) 側風 (D) 逆風。
- () 9. 下列何種滅火器不適用於撲滅丙 (C) 類電氣火災？ (A) ABC 乾粉滅火器 (B) 泡沫滅火器 (C) 二氧化碳滅火器 (D) 鹵化烷滅火器（俗稱海龍，Halon）。
- () 10. 一般 18 公斤以下滅火器，放置的高度不得超過多少公尺？（滅火器上端與樓地板之距離） (A) 1 公尺 (B) 1.5 公尺 (C) 2 公尺 (D) 2.5 公尺。

程式架構的認識與實作

2-1 程式語言與演進

✧ 程式語言

人與人之間溝通的工具稱為語言，世界上因有許多民族，因其發源地不同，所以就有許多語言。例如，華語、英語及德語等。其次，人與電腦溝通的工具，則稱為電腦程式語言。那麼為什麼沒有電視語言、冰箱語言或冷氣語言呢？那是因為這些機器的功能較為簡單，只要幾個按鈕就能發揮其功能。但是電腦的功能就非常多，多到連用整個鍵盤的所有按鍵都無法表現其功能，所以必須使用一些類似單字所組成的片語與敘述來發揮其所有功能，這些單字與片語的集合就稱為電腦程式語言，簡稱程式語言。就如同人類也無法用 26 個字母表達所有感受與思維，必須藉助這些字母的排類組合，先組成單字，再由單字組合成片語與句字，才能充分表達其思維。

✧ 程式語言的演進

電腦語言依其演化先後，大致可分為低階語言 (Low-level programming language) 與高階語言 (High-level programming language)，分別說明如下：

☛ 低階語言

低階語言依其演化先後，又可分為機器語言 (Machine language) 與組合語言 (Assembly language)，分別說明如下：

機器語言

機器語言是由一堆 0 與 1 所組成的單字。例如，要在螢幕輸出 A，所需鍵入單字如下：

B2 41 B4 02 CD 21 CD 20

2-2 程式設計實習進階

由於這些單字的來龍去脈過於複雜，絕非一般人所能體會。只因為要於螢幕顯示一個 A，就要鍵入以上 8 個單字，那如果要計算一個正方形面積，其所需鍵入的單字可能比火車還長，還不如用筆算還快，哪需要計算機呢！其次，電腦剛發明時並沒有鍵盤，要鍵入以上單字，必須用 8 個位元一組的指撥開關，分別輸入 8 次（共 8 個單字）。例如，B2 是由 (10110010) 所組成，則必須先將 8 個指撥開關分別設定為 (on,off,on,on,off,off,on,off)，由此可見當時使用機器語言的艱辛，真是未蒙其利，先受其害。

組合語言

當時學習與使用機器語言是艱辛的，而且電腦的體積又非常龐大，通常只有作為學術機關研究用。但是當時科學家並未氣餒，為了推廣這一神奇而且好用的工具，所以有組合語言的發展。所謂組合語言就是使用一些助憶碼（Assembly mnemonic），這些助憶碼類似一些簡單的英文單字，以便協助使用者記憶與書寫所要完成的程式。例如，上例中要將字元 A 顯示於螢幕，所需組合語言如下：

```
MOV DL,41
MOV AH,2
INT 21
INT 20
```

其中，MOV 是 MOVE 的縮寫，41 是字元 A 的 ASCII 碼，DL 與 AH 則是 CPU 的內部暫存器，INT 21 是呼叫字元顯示副程式，INT 20 則是呼叫程式結束處理副程式。也就是程式發展已經有一脈絡可循，一切遵循一些簡單規則，方便使用者的查詢與記憶。由於有了組合語言的好工具，學習電腦終於可以突破實驗室，推廣到大學以上的資訊電子相關科系。

☛ 高階語言

有了組合語言，電腦使用的方便性可說向前邁向一大步，但是電腦的使用卻僅限於資訊電子相關科系，因為不同的 CPU 就要有不同的組合語言，也就是說此程式語言沒有硬體相容性。其次，這些助憶碼也要花費很多時間學習，而且每一小小功能的程式長度都要超過一頁，所以科學家並不以此為滿足，所以有高階語言的研發。例如，前例要在螢幕顯示字元 A，若使用 C 語言，則只要撰寫如下：

```
printf("%c",'A');
```

又例如，面積的計算，程式如下：

```
int a=15;
int b=8;
```



```
int c;
c=a*b;
printf("面積=%d\n",c);
```

以上程式與人類學習數學的習慣非常接近，已經可以被大部分人接受。

✂ 程式語言的選擇

電機電子群的專長是電機電子設備的控制，學習程式語言的目的主要是銜接二、三年級的嵌入式晶片控制。目前所有的嵌入式系統的程式語言都是使用 C 語言，所以本書選擇 C 語言作為電機電子群程式設計實習的語言。其次，物件導向已經是程式語言領域的標準配備，C++ 是繼承 C 語言的物件導向語言，所以本書第十二章導入物件導向程式設計語言時，選擇 C++ 闡述物件導向的程式設計。

2-2 專案架構、程式架構及語言架構

✂ 專案架構

一個可執行的 C 語言稱為一個專案，每個專案可由至少一個 `main()` 函式、若干函式、組合語言函式、C 標準函式庫、自訂函式庫所組成，且由 `main()` 函式當作起動函式，如 2-1 圖所示。

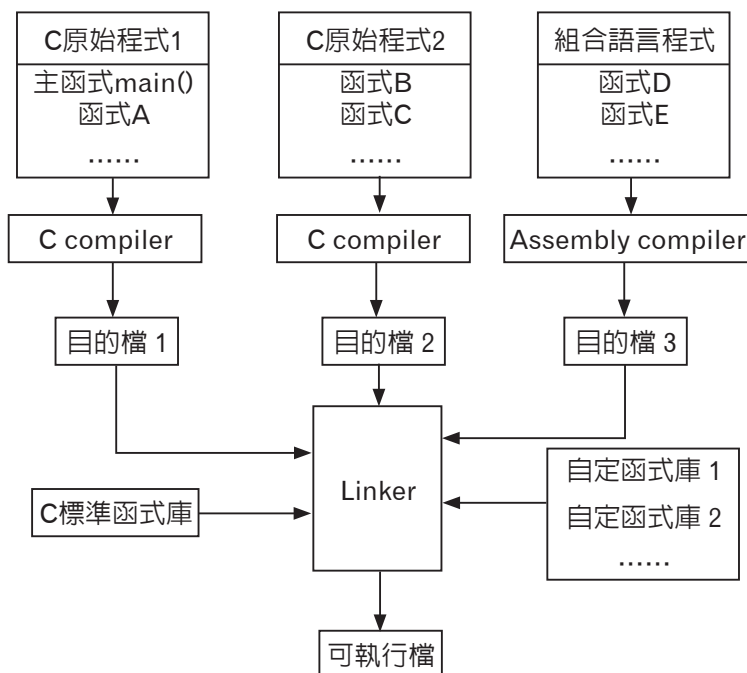


圖 2-1 C 程式專案架構

依據模組化原則，大的專案都要先分解為很多單一函式。所以實用的 C 程式

都是由很多單一功能函式組合而成，因為單一功能函式容易撰寫、除錯與編譯，這些單一功能都先使用編譯程式 (Compiler) 編譯成目的檔「*.obj」，如上圖的目的檔 1、或目的檔 2。每當要改變程式功能時，只要修改對應的函式，編譯此函式，不用全部重新編譯，當然效率也高。最後階段再使用連結程式 (Linker)，連結所有目的檔「*.obj」、對應的標頭檔 (C 標準函式庫共有 29 個標頭檔) 或已經編譯完成的自訂函式庫，即可完成可執行檔。

🔗 程式架構

以上專案架構有點複雜，早期學習 C 語言程式設計，都要自己使用 Compiler 編譯程式，將程式編譯成目的程式 (「*.obj」)。若需修改，則修改後再編譯。最後再使用 Linker 連結所有目的程式 (「*.obj」) 成為可執行檔案「*.exe」。所幸，目前有 Code::Blocks 統整與管理以上瑣事。以下是使用 Code::Blocks 開啓新專案時，系統準備好的基本程式架構。

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    printf("Hello world!\n");
    return 0;
}
```

🔗 標頭檔

所有 C 語言程式都要載入兩個標頭檔

```
#include <stdio.h>
#include <stdlib.h>
```

這些標頭檔內含許多由 C 語言事先編寫好的函式，例如內含許多輸出入函式供我們呼叫，只要載入它們兩個，我們就可以輕鬆作出輸出入的功能。例如：本例的

```
printf("Hello world!\n");
```

就可以輸出「Hello world!」。我們作為程式設計師，只要靈活呼叫這些函式，就可以完成許多常見的工作。

🔗 新增/刪除檔案

若要新增檔案，也是點選功能表「File/New/File」即可新增 C 函式或組合函式；若要修改函式，也是先點選函式名稱再修改。

☛ 執行程式

完成程式的編輯，最後點選「Build/Build and Run」即可編譯所有函式、連結所有目的檔，且執行程式。

以上是最簡單的 C 程式，我們剛剛強調 C 語言是由許多函式組合而成，以下則是自己新增函式的 C 程式架構，程式分為四大區塊，分別是函式引用區、全域變數宣告區、函式實作區、主程式區，如表 2-1 所示：

► 表 2-1 C 程式架構

標題	程式
函式引用區（載入所需標頭檔）	<code>#include <stdio.h></code> <code>#include <stdlib.h></code>
全域變數宣告區	<code>int s;</code>
函式實作區	<code>int add (int a, int b){</code> <code>int c;</code> <code>c = a + b;</code> <code>return (c);</code> <code>}</code>
主程式區 main()	<code>int main() {</code> <code>s = add (6, 2);</code> <code>printf("%d",s);</code> <code>return 0;</code> <code>}</code>

☛ C 語言架構

C 語言的指令分為資料型態的宣告、運算子、決策指令、迴圈指令、函式、指標，其語言指令架構如表 2-2。

► 表 2-2 C 語言指令架構表

C 語言指令架構		
指令分類	指令	本書章節
資料型態的宣告	常數 :const #define char,int,float,double,short,long,void,struct	3,4,9
C 語言指令架構		

指令分類	指令	本書章節
運算子	算術運算 : +=, -=, *=, /=, %, ++, -- 關係運算 : >, <, >=, <=, ==, != 邏輯運算 : !, &&, 位元運算 : !, &, , ^, >>, <<	6
決策指令	if, else, switch, case	7
迴圈指令	for, while, continue, break	8
函式	函式	5, 11
指標	指標	10

以上指令對應章節如表 2-2 的「本書章節」，我們將會在以下各章節陸續介紹。

自我練習

- 以 C 語言開發程式的敘述，下列何者錯誤？（統測 111 答案 B）
 - 程式需要經過編譯及連結產生可執行檔，才能夠執行
 - 使用 `#include` 前置處理命令時，命令結尾需要加上分號
 - `main()` 是一個函式，程式執行時從 `main()` 函式開始執行
 - 單行註解可用 2 個斜線 (`//`) 開頭
- 以 C 語言開發程式的敘述，下列何者錯誤？統測 112 B
 - 程式需要經過編譯及連結產生可執行檔，才能夠執行
 - 使用 `#include` 前置處理命令時，命令結尾需要加上分號
 - `main()` 是一個函式，程式執行時從 `main()` 函式開始執行
 - 單行註解可用 2 個斜線 (`//`) 開頭
- 利用 C 程式語言開發程式的敘述，下列何者錯誤？（技藝競賽修護組 111）
 - 程式需要經過編譯及連結產生可執行檔，才能夠執行
 - 單行註解可用 2 個斜線 (`//`)
 - `main()` 是一個函式，程式執行時從 `main()` 函式開始執行
 - 使用 `#include` 前置處理命令時，命令結尾需要加上句號。
- 請問 C 語言模組（modules）標頭檔的副檔名是下列哪一個？
技藝競賽修護組 111）
 - .c
 - .dev
 - .h
 - .proj
- 請問下列哪一種工具程式的功能是電腦程式語言的翻譯者？
（技藝競賽修護組 111）
 - 編譯器
 - 載入器
 - 瀏覽器
 - 文書處理器
- 有關程式語言的敘述下列何者不正確？

- (A) 高階語言佔用記憶體的空間遠低於低階語言
- (B) 將目的程式碼連結為可執行碼是連結器的主要功能
- (C) 高階語言的程式其可攜性高於低階語言的程式
- (D) C 語言是高階程式語言。(技藝競賽修護組 110)

2-3 演算法的認識

演算法基本概念

演算法 (algorithm) 是指電腦程式完成一項工作所需要『步驟』的集合。演算法嚴謹定義如下：

在有限 (finite) 的步驟 (step) 所構成的集合中，依照給定輸入 (input)，依序執行每個明確 (definite) 且有效 (effective) 的步驟，以便能夠解決特定的問題，而且步驟的執行必定會終止 (terminate)，並產生輸出 (output)。

演算法的流程表示

常見的演算法流程表示有三種，分別是自然語言、虛擬碼與流程圖。分別說明如下：

☛ 自然語言(Natural language processing)

自然語言就是使用我們日常生活的文字表示或已經熟悉的數學語言。例如，以下是求 9 開根號的演算法。

1. 使用迴圈從 1 到 9 分別求其平方。(此稱為循序法)
2. 若其平方大於等於 9，此數即為所求。例如：1 的平方是 1，2 的平方是 4，3 的平方是 9，9 已經大於等於 9，所以 3 即為所求，演算結束。

☛ 虛擬碼(Pseudocode)

在自然語言中，有時嵌入一些慣用常用程式敘述，例如 for 代表迴圈，if 代表決策，這樣可以縮短文字長度，也會讓敘述更加清楚易懂。例如，以下是求 9 開根號的演算法。

```
for i=1 to 9{  
    y=i*i  
    if y>=9 then
```

```

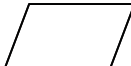
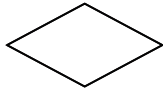
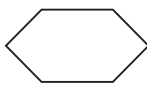
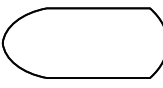
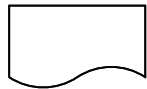
    print(i)
  end for
}

```

☞ 流程圖(flow chart)

流程圖 (flow chart) 是利用各種方塊圖形、線條及箭頭等符號來表達解決問題的步驟及進行的順序，常用的流程符號如表 2-3：

► 表 2-3 流程圖符號表

編號	符號	意義
1	 	起迄符號。 代表流程圖的開始或結束，此符號若是開始，則僅有一出口，若是結束則僅有一入口。
2		輸入與輸出符號。 用來填入輸入與輸出的符號，此符號有一入口與一出口。
3		處理符號。 用來填入處理程序，此符號有一入口與一出口。
4		決策符號。 用來表示決策分歧點，此符號的出口至少有兩個，分別是 true 或 false。
5		重複符號。 聲明重複指令的起點與離開條件，通常配合下面的連結符號表示重複的範圍。
6		連結符號。 可配合上面的重複符號或移至另一頁的起點。
7		副程式。(副程式又稱函式) 用來表示另一個副程式。
8		程式流向符號。 用來連結以上符號，說明程式的流向。
9		代表輸出至螢幕。 將資料由螢幕輸出。
10		代表輸出至印表機。 將資料由印表機輸出。

例如，圖 2-4 是求 9 開根號的演算法。

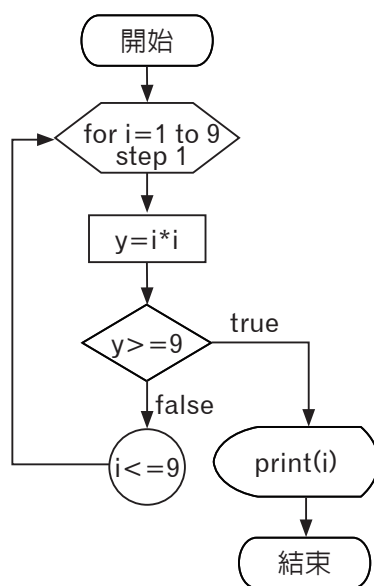


圖 2-4 開根號流程圖

又例如，圖 2-5 的流程圖，可以將輸入成績分類，分為 A、B、C、D 等。

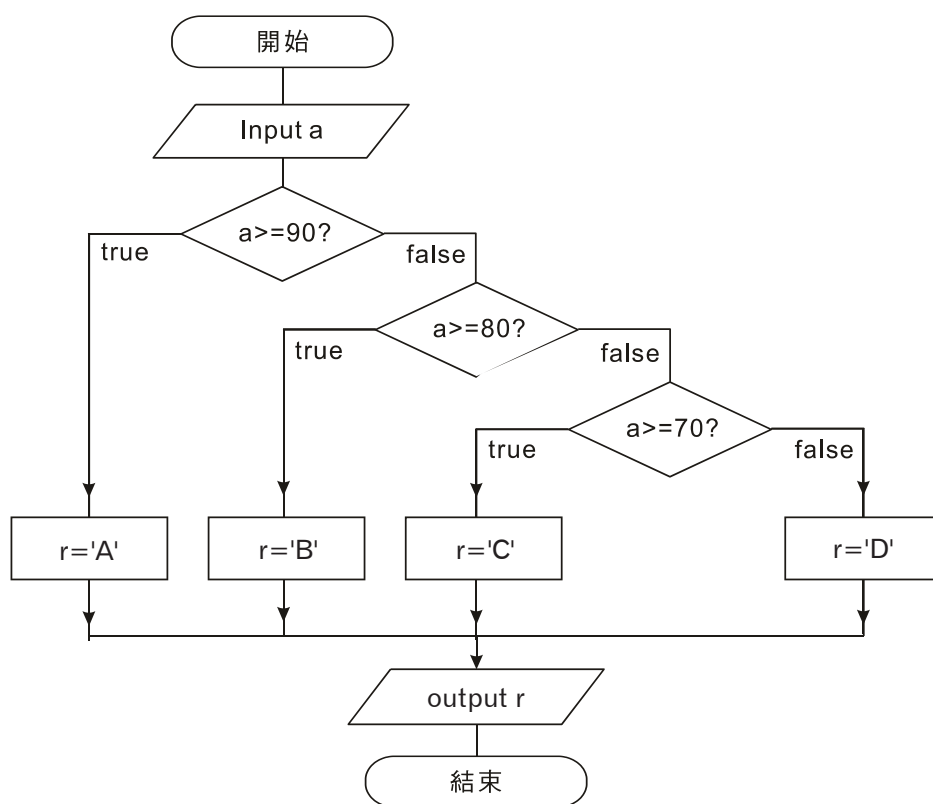
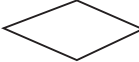


圖 2-5 成績分類流程圖

 自我練習

1. 在流程圖符號中，下列圖形代表何種意義？ (技藝競賽修護組 110)
 (A) 判斷 (B) 輸入 (C) 輸出 (D) 處理。

 2-4 程式設計步驟**範例2-4a**

已知長方形的長與寬，請寫一個程式，求其面積。

 程式設計解題步驟

- 資料的數位化
 - 本題若使用掌上型計算機，其方法是直接將鍵入「長 * 寬 =」，就可得到答案。例如，鍵入「15*8=」，就可得到答案「120」。
 - 但使用 C 語言，必須先將所要計算的值先以一個英文代號儲存，例如 a,b 或 score，以上代號在程式設計的領域，稱為「變數」。例如：

```
a=15
b=8
```

設定變數的目的是將資料與程式分開，這樣當資料改變時，還可重複計算。計算過程都是以變數與程式指令描述，此稱為程式設計。程式設計的優點是有一致性，只要第一次對，往後都對。掌上型計算機就沒有了一致性了，每次都要按兩次，才能確認計算結果是否正確，且沒無法重複使用。以上變數的命名請看 4-1 節。

- 選用資料型態。C 語言的資料型態請看 3-2 節，本例先選用 int(整數型態)，且每一個敘述都要以分號 (;) 結束，所以程式如下：

```
int a=15;
int b=8;
```

- 寫出演算法。本例是輸入長方形的長與寬來計算面積，所以演算法如下：

```
面積=a*b
```

3. 使用變數與程式指令，完成以上演算法。本例是計算面積，面積也要使用變數儲存，本例使用 `c`，所以是

```
c=a*b;
```

4. 輸出結果。

```
printf("面積=%d\n",c);
```

5. 以上全部程式如下：

```
int a=15;
int b=8;
int c;
c=a*b;
printf("面積=%d\n",c);
printf("%d",c);
printf(c)
```

6. 配合 C 語言的程式架構，全部程式如下：

```
#include <stdio.h>
#include <stdlib.h>
int main() {
    int a=15;
    int b=8;
    int c;
    c=a*b;
    printf("面積=%d\n",c);
    return 0;
}
```


資料的種類與資料型態

3-1 資料種類

C 語言所能處理的資料種類分別有數值（含整數、長整數及浮點數等等）、字元與字串、布林值等。

☞ 整數(Integers)

C 語言可以處理的整數有三種進位方式，分別是十進位 (Decimal)、十六進位 (Hexadecimal) 及八進位 (Octal)。其中十進位則以我們平常書寫數字的方式即可；十六進位應以數字 0X 或 0x 開頭（數字 0），且以 0,1,2,3,4,5,7,8,9,a,b,c,d,e,f 代表十進位的 0 到 15，英文字元 a,b,c,d,e,f 大小寫都可以，例如 0xA 或 0xf 均為十六進制，分別代表十進位的 10 與 15；八進位則應以數字 0 開頭，例如 :072 則為八進位，等於十進位的 58。例如：請鍵入以下程式，寫出執行結果。

```
#include <stdio.h> #爲了節省篇幅，以下程式都省略此宣告
#include <stdlib.h> #爲了節省篇幅，以下程式都省略此宣告
int main() {
    int a=10;
    printf("%d\n",a); //_____
    int b=010;
    printf("%d\n",b); //_____
    int c=0x10;
    printf("%d\n",c); //_____
    return 0;
}
```

```
int main(){
    int a=10,b=010,c=0x10;
    int d=a+b+c;
    printf("%d",d);
}
```

☞ 浮點實數(Floating-point literals)

數字中含有小數點或指數的稱為浮點數、實數或浮點實數。以指數為例，E 或 e 表示 10 的次方，例如 0.0023、2.3E-3 及 2.3e-3 都是表示相同的浮點數；又例如 2.3E+2 則代表 230。浮點數可使用標準寫法或科學符號法表示，例如 321.123 即為標準寫法，1.23e+4 即為科學符號表示法。

```
int main(){
    double a=0.1,b=1.1E-2,c=1.1e2;
    double d=a+b+c;
    printf("%lf",d);
}
```

☞ 字元(Character literals)

使用一對單引號『 ' 』圍住的單一字元，稱為字元，例如 'A' 或 'a' 等。C 語言使用 8 bits 表示一個字元，所以可表示 256 個字元 ($2^8=256$)，包含大小寫英文字母、數字、標點符號及其它特殊符號。

☞ 字串

C 語言中，字串使用一對雙引號『 " 』圍住，例如 "Gwosheng", "台灣" 等。

☞ 布林值

我們人類表達對與錯的語句有：對、錯、是、否、正確、錯誤。C 語言則統一使用 1 代表布林的 true，0 代表布林的 false。

☞ 自我練習

1. 如果 X_n 代表 X 這個數字是 n 進位，請問 $D02A_{16} + 5487_{10}$ 等於多少？
(APCS10503)
(A) 1100 0101 1001 10012 (B) 1626318 (C) 5878716 (D) F59916

2. 請問十進制的 234 加上八進制的 1067 再加上 16 進制的 10b 答案為何？
(技藝競賽程式組 110)
- (A) 十進制的 1048 (B) 八進制的 2054
(C) 16 進制的 42d (D) 二進制的 0010_0010_1100
3. 10 進制的數值 1068.875，轉換為 2 進制答案為？
(A) 1000101100.111 (B) 10000101100.011 (C) 1111011.110 (D) 10000101100.111

3-2 資料型態 (Data Type)

前面已經介紹人類經常使用的一些資料常數，每一種資料常數均需要不同大小的記憶體儲存，電腦為了有效率的儲存與處理這些資料，所以有資料型態的規劃。也就是大的資料用大盒子裝，小的資料用小盒子裝，如此即可節省記憶體，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，也拖慢執行效率。

C 語言的資料型態首先分為內建資料型態及複合資料型態，內建資料型態再分為數值 (int、long、float、double、long double)、字元 (char) 資料型態及函式專用型態 (void)，複合資料型態再分為陣列 (array)、結構 (struct) 等型態。

數值資料型態

表 3-2 是 C 語言的數值資料型態，unsigned 表無號整數，即非負數整數：

► 表 3-2 C 語言資料型態表

數值資料型態	佔用記憶體的大小(位元)	所能代表的數值的範圍
short	16	-32,768 ~ +32,767
int	32	-2,147,483,648 ~ +2,147,483,647
long	32	-9223372036854775808 to 9223372036854775807
unsigned short	16	0 ~ 65,535
unsigned int	16	0 ~ 4,294,967,295
unsigned long	32	0 to 18446744073709551615
float	32	$3.4 \times 10^{-38} \sim 3.4 \times 10^{38}$
double	64	$1.7 \times 10^{-308} \sim 1.7 \times 10^{308}$
long double	128	$3.4 \times 10^{-4932} \sim 3.4 \times 10^{4932}$

☞ 整數

我們可以使用 signed(有號)與 unsigned(無號)的 short、int 及 long 等六種型態來儲存整數。例如，一天的秒數是 86400，已經超過整數 short 的範圍，那就要用 int 表示。但真是太複雜了，因此本書整數一律使用整數預設型態的 int 型態，其所能代表的數值的範圍是 -2,147,483,648 ~ +2,147,483,647。

☞ 浮點數

我們可以使用 float、double 及 long double 等三種型態來儲存浮點常數。浮點數的預設值是 double，但若要強制使用 float，則應於數值後面加上英文字母的 F 或 f，例如，3.1415926F 或 3.1415926f。但若要使用 long double，則於數值後面加上 L 或 l，例如，3.1415926L 或 3.1415926l。

► 表 3-3 常用資料型態表

數值資料型態	佔用記憶體的大小(位元)	所能代表的數值的範圍
int	32	-2,147,483,648 ~ +2,147,483,647
double	64	$1.7 \times 10^{-308} \sim 1.7 \times 10^{308}$

☞ 字元資料型態

C 語言使用 char 資料型態來儲存一個 ASCII 字元，因此，C 語言中的 char 型態是 8 位元長。

☞ 布林型態

C 語言設計之初並沒有布林型態，後來才新增 _Bool 布林型態。

☞ 字串資料型態

C 語言並無字串資料型態，而是使用字元陣列，例如：

```
char a[10]="gwosheng";
```

關於字元陣列，請看 9-3 節。

自我練習

題號	題目	結果
1	阿華想要了解 C 語言程式中不同資料型態佔據記憶體空間的大小，於是使用 C 語言中的 sizeof () 函式並撰寫如下程式來測試，所使用的電腦為 64 位元的個人電腦，下列何者為該程式的執行結果？(統測 112) <pre> 1 #include <stdio.h> 2 3 int main() { 4 char *s="ABC"; 5 int i=10,CharPtrSize,CharSize; 6 CharPtrSize = (int)sizeof(s); 7 CharSize = (int)sizeof(*s); 8 printf("%d, %c", CharSize, *(s+2)); 9 10 return 0; 11 }</pre> (A) 8, B (B) B, 1 (C) 1, C (D) C, 8	
2	<pre> int main() { char s="A"; char p[]="ABCD"; int i=10; float j=10; double k=10; char *ptr=p; printf("%d\n", (int)sizeof(s)); //1 printf("%d\n", (int)sizeof(p)); //5 printf("%d\n", (int)sizeof(i)); //4 printf("%d\n", (int)sizeof(j)); //4 printf("%d\n", (int)sizeof(k)); //8 printf("%d\n", (int)sizeof(ptr)); //8 printf("%d\n", (int)sizeof(*ptr)); //1 return 0; }</pre>	

3	<pre> int main() { char *s="ABC";//使用指標表示字串 int i=10,CharPtrSize,CharSize; CharPtrSize = (int)sizeof(s);//s is 指標 CharSize = (int)sizeof(*s);//字元 printf("%d\n", CharPtrSize);//8 指標都是8 printf("%d\n", CharSize);//1 字元 printf("%c\n", *(s+2));//C printf("%c\n", *(s+1));//B printf("%c\n", *s);//A s++; printf("%c\n", *s);//B s++; printf("%c\n", *s);//C return 0; } </pre>	
4	<pre> int main() { char ss[]="ABC";//使用陣列表示字串 char *s=ss; int i=10,CharPtrSize,CharSize; CharPtrSize = (int)sizeof(s); CharSize = (int)sizeof(*s); int IntSize=(int)sizeof(i); printf("%d\n", IntSize);//4 printf("%d\n", CharPtrSize);//8 printf("%d\n", CharSize);//1 printf("%c\n", *(s+2));//C printf("%c\n", *(s+1));//B printf("%c\n", *s);//A s++; printf("%c\n", *s);//B return 0; } </pre>	
5	<pre> int main() { int a=5; int *ptr=&a; printf("%d\n", (int)sizeof(a));//4 printf("%d\n", (int)sizeof(*ptr));//4 printf("%d\n", (int)sizeof(ptr));//8 指標都是8 return 0; } </pre>	

6	<pre> int main() { int a[]={4,1,5}; int *ptr=a; printf("%d\n", (int)sizeof(a)); //12 printf("%d\n", *ptr); //4 printf("%d\n", (int)sizeof(*ptr)); //4 //print("%d\n", ptr); //輸出位址 printf("%d\n", (int)sizeof(ptr)); //8 指標都是8 return 0; } </pre>	
7	下列敘述何者錯誤？ (技藝競賽修護組 112) (A) 宣告為 int 之資料佔 2 Bytes 的資料 (B) 宣告為 float 之資料佔 4 Bytes 的資料 (C) 宣告為 double 之資料佔 8 Bytes 的資料 (D) 宣告為 char 之資料佔 1 Byte 的資料。	
8	關於 C 程式語言的資料型態，下列敘述何者正確？ (技藝競賽修護組 112) (A) float 資料型態可以儲存浮點數資料 (B) 宣告 int 資料型態可以儲存整數和浮點數資料 (C) double 資料型態可以儲存浮點數值，數值精確度跟 float 資料型態相同 (D) 宣告 char 資料型態可以儲存字元符號和浮點數資料。	
9	int 的資料型態佔 4 個 bytes，請問 int 型態能表達的最大正數為何？ (技藝競賽程式組 109) (A) :232 (B) :231 (C) :232 - 1 (D) :231 - 1	
10	int 的資料型態佔 4 個 bytes，請問 int 型態能表達的最小負數為何？ (技藝競賽程式組 109) (A) :-232 (B) :-231 (C) :-232 - 1 (D) :-231 - 1	
11	使用大 5 碼 (Big-5) 儲存 10 個中文字需多少位元組？ (技藝競賽程式組 109) (A) :10 bytes (B) :20 bytes (C) :40 bytes (D) :80 bytes	

2. 繼續於上圖點選『Type』，即可得所有資料型態的詳細資料。

☞ 跳脫字元序列

字元中的單引號 (')、雙引號 (") 及反斜線 (\) 在 ASCII 均已有定義其功能，若您一定要使用這些字元，則應於使用前先加一個反斜線 (\)，以跳脫其原先所定義

的功能，此稱為跳脫字元序列 (Escape Sequence)。表 3-4 是一些常用的跳脫字元：

► 表 3-4 跳脫字元序列表

字元	跳脫字元序列
單引號 (')	\'
雙引號 (")	\"
反斜線 (\)	\\

例如：

```
char a[]="Gwosheng";
printf("%s\n",a);
```

結果是 Gwosheng。又例如：

```
char b[]="\\"Gwosheng\\"";
printf("%s\n",b);
```

結果是 "Gwosheng"。

自我練習

1. 鍵入以下程式，並觀察執行結果。

題號	題目	結果
1	<pre>int main(){ char b[]="\'Gwosheng\'"; printf("%s\n",b); char c[]="\%Gwosheng\\"; printf("%s\n",c); }</pre>	
2	<pre>int main(){ int a=15,b=8,c=3; int d=a*b*c; int e=2*((a*b)+(b*c)+(c*a)); printf("%d\n",d); printf("%d\n",e); return 0; }</pre>	

變數與常數

4-1 識別字與保留字

☞ 識別字(Identifiers)

真實的世界裏，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為每一個變數、常數、函式、結構及類別命名，以上所有變數、常數、函式、結構及類別等名稱，統稱為程式語言的識別字。C 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，但不得包含空白。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. 識別字的長度不可以超過 32 個字元。
4. 識別字的大小寫均視為不同，例如 Score、score 及 SCORE 皆代表不同的識別字。
5. 識別字不得使用保留字，如 if、for 等。但是，if1、fora 等則可使用。
6. 識別字亦可使用有意義的單字的組合，這樣更能傳達變數所代表的意義。例如：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。
7. 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，若擔心單字拼錯亦可寫成 Stu_Num、stu_num、StuNum 或 stumum，其中 StuNum 又稱駝峰表示法，因為大寫字母看起來像駱駝駝峰一樣，可以避免鍵入底線的困擾、且提昇閱讀效率。
8. 不要單獨使用 l,o,O 等字母當作變數名稱，因為這些字母與阿拉伯數字 1,0 很相近，很容易讓人誤判。

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，稱為識別字）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。例如 if 已賦予決策敘述，程式設計者當然不得再定義 if 為另外的用途，表 4-1 是 C 語言的保留字，請於 <https://en.cppreference.com/w> 點選『C Keywords』。

► 表 4-1 C 語言保留字

auto	float	signed	_Alignas (since C11)
break	for	sizeof	_Alignof (since C11)
case	goto	static	_Atomic (since C11)
char	if	struct	_Bool (since C99)
const	inline (since C99)	switch	_Complex (since C99)
continue	int	typedef	_Generic (since C11)
default	long	union	_Imaginary (since C99)
do	register	unsigned	_Noreturn (since C11)
double	restrict (since C99)	void	_Static_assert (since C11)
else	return	volatile	_Thread_local (since C11)
enum	short	while	
extern			

4-2 變數與常數宣告與應用

變數的宣告

變數的功能是用來輸入、處理及儲存外界的資料，而變數在使用以前均要事先宣告才可使用。C 語言的變數宣告語法如下：

```
資料型態 變數名稱 [=初值];
```

例如：

```
int a;
```

即是宣告變數 a 為 int 型態。又例如：

```
char b,c;
```

則是宣告變數 b,c 為 char 型態。變數的宣告亦可連同初值一起設定，如下所示：

```
double d = 30.2;
```

宣告 d 是 double 型態，並設其初值是 30.2。以下程式，宣告 e 是 char 型態，且其初值是 'a'。

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

以下敘述，同時宣告兩個整數，且指派其初值。

```
int f1=3,f2=4;
```

以下敘述可宣告變數為字元陣列，字元陣列就能代表字串。(陣列於 9-3 節有更詳細說明)

```
char g[]="Gwosheng";
```

變數宣告應用

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。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。以下敘述即為變數未宣告的結果，編譯器便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，有些變數的值域僅為整數，則不用宣告為 float 或 double。此即為小東西用小箱子裝，大東西用大箱子裝，才能有效運用空間與提升搬運效率。

自我練習

1. 請鍵入以下程式，編譯、除錯與執行程式。

題號	程式	結果
1	<pre>#include <stdio.h> #include <stdlib.h> int main() { int 7eleven; // 不能由數字開頭 int %as; // 不能由符號開頭 int _a; int A=; // 不能含有 = 號 int sum! ; // 不能含有! 號 int Age#3; // 不能含有 # 號 int a c ; // 不能含空白 int c+3 ; // 不得含有加號 int if; // 不得使用保留字 return 0; }</pre>	
2	<pre>#include <stdio.h> #include <stdlib.h> int main() { int student=0; student=studend+1; printf("%d",student); int a=3.4; char b,c; b="aa" c="a"; int d='a'; double e=3; return 0; }</pre>	

3	下列那一個不是 C 語言的合法變數名稱？ (技藝競賽修護組) (A) _Test (B) TEST (C) 5test (D) test1	
---	---	--

☞ 常數的宣告

程式設計有兩種表示常數的方式，一種是文字式 (Literal)，例如直接以 15 或 3.14159 表示某一常數；另一種是本單元的常數符號式 (Symbolic)。因為有些數字在程式中會不斷的重複出現，為了增加程式的可讀性及減少程式鍵入的麻煩，此時即可用一個有意義的符號代替，我們可以利用 `const` 或 `define` 來定義常數，則該符號的值將永遠保持在你所宣告的符號，程式中任何位置均不可能改變其值，此稱為常數符號，簡稱常數。`const` 宣告常數，語法如下：

```
const 型態 常數名稱=常數值;
```

例如，以下程式令 `PI=3.14`，常數符號通常均用大寫表示。

```
const double PI=3.14;
```

則每次要使用 3.14 時，只要填入 `PI` 即可。例如，以下程式可計算圓面積。

```
a=3*3*PI;
```

`define` 定義常數語法如下：

```
#define 常數名稱 常數值
```

例如：

```
#define PI 3.14;
```

☞ 常數宣告的應用

又例如，於銀行利息的計算，利率亦應以常數符號表示，程式中任何位置若需使用此利率時，均應以此常數符號代替，當要調整此利率時，只要在程式的最前面修改此常數符號的值即可。若未使用常數符號統一此值，則因此常數散落程式各地，而必須到處修改，萬一有些地方漏掉未修改，則無法確保此值的一致性。

 自我練習

1. 請鍵入以下程式，並寫出執行結果。

```
#include <stdio.h>
#include <stdlib.h>
int main() {
    const double PI=3.14;
    double a;
    a=3*3*PI;
    printf("%f\n",a);
    const int b=4;
    b=b+1;//常數不能改變其值
    printf("%d",b);
    return 0;
}
```

2. 請鍵入以下程式，並寫出執行結果。

```
#include <stdio.h>
#include <stdlib.h>
#define PI 3.14
int main() {
    double a;
    a=3*3*PI;
    printf("%lf\n",a);
    PI=PI+1;//常數不能改變其值
    printf("%lf",PI);
    return 0;
}
```

3. 關於 C 程式語言中，使用 define 建立常數的方式，下列何者正確？（111 年統測）
 (A) define PI = 3.14; (B) define PI 3.14;
 (C) #define PI = 3.14 (D) #define PI 3.14 答案：(D)
4. 題組，閱讀下文，回答以下兩題。小芳在一個原本可以編譯 (Compile) 成功的程式中，在 main() 主程式內再加入行號 1 至行號 6 的程式碼，但加入後發生編譯錯誤的情況。 統測 112

```
1  #define Value1 100
2  #define Value2 (Value1 - 1)
3  const int Value3;
4  int CheckValue = 0;
5  Value3 = Value2;
6  CheckValue = Value1 + Value3;
```

(1) 小芳刪除行號 1 至行號 5 中的哪一個部分後，可以使程式編譯成功？

(A) (Value1 - 1) (B) Value3 = Value2;

(C) const (D) #define Value2 (Value1 - 1)

答案：(C)

以上解說如下，請先鍵入以下程式：

```
#include <stdio.h>
#include <stdlib.h>
int main() {
    #define Value1 100
    #define Value2 (Value1 - 1) //99
    int Value3;
    int CheckValue = 0;
    Value3 = Value2;//99
    CheckValue = Value1 + Value3;//100+99
    printf("%d",CheckValue);//199
    return 0;
}
```

以上變數 Value3 加上 const 代表此為常數，常數無法修改其值。

(2) 程式修正後，當程式執行完行號 6 的時候，CheckValue 的值為下列何者？

(A) 200 (B) 199 (C) 198 (D) 100

答案：(B)

4-3 資料型態轉換

每一個變數宣告之後，即有屬於自己的型態，往後此變數均只能指派給相同或相近型態的變數儲存，若執行階段欲指定給相近型態的變數儲存，則稱此為型態轉換。在 C 語言中型態轉換又可分為隱含轉換（Implicit Conversion）與強制轉換（Explicit Conversion），分別說明如下：

隱含轉換（Implicit Conversion）

將值域小的型態轉為值域大的型態，稱為自動轉換或轉型（Convert）。此種轉換，系統可自動處理並確保資料不會流失。例如，將 int 轉為 long，則因後者的值域比前者大，所以可順利的轉換。以下敘述可將型態為 int 的變數 a 指派給型態為 long 的變數 b，且原值不會改變。

```
int a=23;
long b;
b=a;
printf("%d",b);
```

結果是 23。

☞ 強制轉換（Emplicite Conversion）

將值域大的變數轉為值域小的變數（如 long 轉為 int），則稱此為強制轉換或稱為鑄型（cast）。強制轉換的語法如下：

```
變數2（值域小） = （變數2的型態） 變數1（值域大）；
```

以上可將變數 1（值域大）的型態強制轉為變數 2（值域小）。例如，以下敘述可將型態是 long 的 c 變數指定給型態是 int 的 d 變數。

```
long c=23;
int d;
d=(int) c;
printf("%d\n",d);
```

結果是 23。強制轉換的風險比較大，有可能資料流失或溢位。例如，以下敘述將 float 型態強制轉換為 int 型態，將造成小數部份的流失。

```
float e=3.4f;
    //浮點數預設為double，加上f,表示此3.4為float，請複習3-2節浮點數
int k;
k=(int) e;
printf("%d\n",k);//3
```

結果 3。其次，關於數值與字串互轉，則要看 11-3 的字串函式。

☞ 自我練習

題號	題目	執行結果
1	<pre>main(){ short a=32768; printf("%d",a); }</pre>	—32768
2	<pre>int main(){ int a=32768; short b=(short)a; printf("%d\n",b); double c=3.6; int d=(int)c; printf("%d\n",d); }</pre>	—32768 3

4-4 資料型態應用實例

範例4-4a

假如有資料如下，請問如何以電腦儲存？

座號	姓名	國文成績	數學成績	總分	平均
1	洪煥維	98	95		

程式設計步驟

1. 資料數位化且以變數儲存。
 - (1) 座號取 int 型態，變數名稱取 no。
 - (2) 姓名取 char 型態，變數名稱取 name。
 - (3) 國文成績取 int 型態，變數名稱取 chi。
 - (4) 數學成績取 int 型態，變數名稱取 math。
 - (5) 總分成績取 int 型態，變數名稱取 total。
 - (6) 平均取 double 型態，變數名稱取 avg。
2. 程式設計如下：

```
#include <stdio.h>
#include <stdlib.h>
int main(){
    int no=1;
    char name[]="洪煥維";
    int chi=98;
    int math=95;
    int total;
    double avg;
    return 0;
}
```


基本輸出/輸入函式

C 語言採用 `printf()` 與 `scanf()` 作為基本輸出入函式。其次，本單元也要介紹亂數的取得，與聲音的產生，請看以下說明。

5-1 `printf()`

`printf()` 可輸出結果，其中括號內須放置一對雙引號『“”』來將欲輸出的資料當作引數傳給 `printf()` 函式。例如：

```
printf("a");  
printf("aa");
```

的輸出結果是：

```
aaa
```

但若要換行與歸位，則應加上跳脫字元『\n』，例如：

```
printf("a\n");  
printf("aa");
```

的輸出結果是：

```
a  
aa
```

其次，如果是變數的輸出，則必須在輸出敘述中加入一個列印格式，此列印格式以控制字元『%』符號開頭，如表 5-1：

► 表 5-1 printf() 列印格式表

資料型別	列印格式
整數	%d
長整數	%ld
浮點數 float	%f
浮點數 double	%lf
字元	%c
字串	%s

然後編譯器會依照控制字元的順序，到後面引數區取對應資料來輸出。例如：

```
int a=3;
printf("%d", a);
```

『%』即表示要到後面引數區取變數來取代，『d』是將變數視為整數，所以輸出結果是：

```
3
```

若是：

```
int a=3, b=5;
printf("%d %d", a, b);
```

有兩個控制字元，那就到後面引數區依照順序取變數輸出，所以輸出是：

```
3 5
```

字串區也可以同時有一般字元與控制字元『%』，此時一般字元就直接輸出，控制字元就到後面引數區依序取變數輸出。例如：

```
int a=3, b=5;
printf("a=%d b=%d", a, b);
```

因為『"a="』是一般字元，就直接輸出，『% d』是控制字元，就去引數區按照順序取對應變數來取代。第一個『%d』取到 a，第二個『%d』取到 b，所以輸出結果是：

```
a=3      b=5
```

輸出過程中如須換行與歸位，也是以跳脫字元序列『\n』來完成。例如：

```
int a=3, b=5;
printf("a=%d\n", a);
printf("b=%d", b);
```

的結果是：

```
a=3
b=5
```

自我練習

1. 請自行鍵入以下程式，並觀察執行結果。

```
int main() {
    int a=97;
    printf("%d\n", a);
    printf("%c\n", a);
    printf("%c\n", (char)a);
    char b='a';
    printf("%c\n", b);
    printf("%d\n", b);
    printf("%d\n", (int)b);
    char c='ab';
    printf("%c\n", c);
    char d='abc';
    printf("%c\n", d);
    char e='a ' ;
    printf("%c\n", e);
    return 0;
}
```

資料的對齊

輸出資料時，亦可指定資料長度，那就可以將所有資料對齊。例如：

```
#include <stdio.h>
#include <stdlib.h>
int main() {
    int a=3,b=55;
```

```
double c1=3.4,c2=15.678;
printf("a =%8d,b =%8d\n",a,b); //佔8格，靠右
printf("c1=%8.2f,c2=%8.2f",c1,c2); //共8格，靠右，小數點取2位
//請留意遇到『%』就是控制字元，就到後面按順序取對應資料輸出
//非控制字元，就直接輸出，例如『a=』，就直接輸出。
return 0;
}
```

的執行結果如下圖，這樣就可以對齊所有資料。

```
a =      3,b =      55
c1=     3.40,c2=    15.68
```

👉 自我練習

1. 在 C 程式語言如果輸出的文字內容需要換行，我們需要再加上哪一個字元？
(技藝競賽修護組 111)
(A) 「\f」 (B) 「\t」 (C) 「\b」 (D) 「\n」。

🔊 5-2 scanf()

scanf() 可輸入資料，其使用方式與輸出 printf() 是一樣的方式，二者均須使用到控制字元『%』格式來控制輸出入的資料。因此，輸入字元、整數、浮點數及字串等會有不同的處理方式，請看以下說明。

🔗 輸入字元

輸入字元的敘述如下：

```
char a;
scanf("%c", &a); //請留意是&
```

可將輸入的資料放入在某個變數所指定位址中，取址符號『&』用來取得欲放置輸入資料的變數位址。例如，以下程式可輸入一個字元。

```
#include <stdio.h>
#include <stdlib.h>
int main() {
    char a;
```

```
printf("Please press any char: ");
scanf("%c", &a); //請留意是&
printf("Your char is %c", a);
return 0;
}
```

請留意輸入資料要有取址符號『&』，漏掉了就是錯誤。

自我練習

1. 請鍵入以下程式，並觀察執行結果。

```
#include <stdio.h>
#include <stdlib.h>
int main() {
    char a,b;
    printf("Please press any char: ");
    scanf("%c", &a);
    printf("Your char is %c", a);
    printf("Please press any char: ");
    scanf("%c", &b);
    printf("Your char is %c", b);
    return 0;
}
```

以上程式，第二次輸入字元

```
scanf("%c", &b);
```

會因鍵盤緩衝問題（只要有先輸入東西，例如整數、實數都有此問題），而直接跳過，這時請將字元前放一個前導空白字元就可以。所以程式修正如下：

```
scanf(" %c", &b); //留意%c的前導空白
```

這在使用手冊有提到 <https://en.cppreference.com/w/c>，請讀者自行瀏覽。

輸入數值

常用的數值型態有 `int`、`long` 與 `double` 等，若要輸入這些型態的數值，則應使用不同的列印格式，例如，`%d`、`%ld` 與 `%lf` 等。請輸入以下程式，並觀察執行結果。

```

#include <stdio.h>
#include <stdlib.h>
int main() {
    int    a;
    long   b;
    double c;
    printf("Enter an integer:    ");
    scanf("%d", &a); //請留意是&
    printf("Enter a long integer: ");
    scanf("%ld", &b); //請留意是&
    printf("Enter a float:        ");
    scanf("%lf", &c); //請留意是&
    printf("\n Your enter..... \n");
    printf("integer      a =%d  \n",  a);
    printf("long integer b =%ld \n", b);
    printf("float        c =%f", c);
    return 0;
}

```

請留意輸入資料要有取址符號『&』，漏掉了就是錯誤。以下可輸入兩個數值，輸入資料時，資料中間以逗號分隔。

```

int a,b,d;
printf("input a,b:");
d=scanf("%d,%d",&a,&b); //請留意中間是逗號
printf("a+b=%d\n",a+b);
printf("d=%d\n",d);    //2    (傳回輸入資料正確的個數)

```

以上執行結果如下：(請自行補程式的頭尾)

```

input a,b:3,4
a+b=7
d=2

```

以下程式可輸入兩筆資料，輸入資料時，資料中間以空白分隔。

```

int a,b,d;
printf("input a b:");
d=scanf("%d %d",&a,&b); //請留意中間是空白
printf("a+b=%d\n",a+b);
printf("d=%d\n",d);    //2    (傳回輸入資料正確的個數)

```


以上程式執行結果如下：（請自行補程式的頭尾）

```
input a b:3 4
a+b=7
d=2
```

其次，scanf() 還傳回正確的資料個數，所以以上程式輸入 2 筆資料，d 值均傳回 2。若輸入資料型態不對，例如，故意輸入 a 3，由於輸入資料型態錯誤，將傳回『0』，或『EOF』，如下圖：

```
input a b:a 3
a+b=16
d=0
```

✂ 輸入字串

大部分程式語言都有字串 string 型態。但是 C 語言爲了加速處理速度，並沒有字串型態，而是使用字元陣列。因此在輸入字串時，我們必須利用字元陣列來完成輸入。例如，以下程式可輸入一個字串。

```
char c[20]; //字元陣列，請看第七章
printf("Please enter a string: ");
scanf("%s", &c); //
printf("Your enter is %s", c);
```

以上輸出結果如下圖：請自行鍵入程式，並觀察執行結果。

```
Please enter a string: Gwosheng
Your enter is Gwosheng
```

關於字串的使用，請看本書第 9-3 節。

👉 自我練習

題號	題目	結果
1	<pre>int main(){ char c[20]; char d; printf("Please enter your name sex: "); scanf("%s %c", &c,&d); // 輸入「國勝 m」 printf("Your name is %s,your sex is %c", c,d); }</pre>	

```

2  int main(){
    char a[20]; //字元陣列，請看第七章
    int chi;    int math;    int sum;
    double avg;
    printf("Please enter your name chi math: ");
    scanf("%s %d %d", &a,&chi,&math);
    //輸入 aa 60 80
    sum=chi+math;
    avg=sum/2;
    printf("%s,your chi is %d,your math is %d,your
sum is %d,your avg is %f", a,chi,math,sum,avg);
    }

```

5-3 亂數

我們寫程式時，有時需要一個亂數，例如樂透開獎，撲克牌發牌等都需要產生一個亂數。所以電腦內部就以記憶體儲存一個亂數表，以供使用者取用。取用亂數的函式是：

```
rand();
```

它將傳回一個 0 到 32767 的任意整數。所以我們宣告一個整數變數暫存，如以下程式：

```
int a;
a=rand();
```

其次，此函式放在 `stdlib.h`，所以還要在標頭檔宣告如下：

```
#include <stdlib.h>
```

還有，因為亂數表是固定的，那要如何取到不同亂數呢？答案是，可取系統時間當依據。因為每次程式執行時，系統的時間一定不一樣，就可取到不同的亂數。以下敘述，以當時的系統時間，當作亂數表的起點。

```
srand(time(0)); //以當時的系統時間，當作亂數表的起點
```

其次，`time()` 函式放在 `time.h`，所以也要載入以下標頭檔。

```
#include <time.h>//本例使用Time() 函式
```

範例5-3a

示範以上亂數的取得。

程式列印

```
#include <stdio.h>
#include <stdlib.h>
//步驟1，引入標頭檔
#include <time.h>//本例使用time() 函式
int main() {
    int a;
    //步驟2，以當時的系統時間，當作亂數表的起點
    srand(time(0));
    //步驟3，取亂數
    a=rand();
    printf("The Random Number is %d .\n", a);
    return 0;
}
```

以下敘述，可取 0~5 的亂數。「%」為取餘運算子，將於 6-1 節介紹。

```
int a=rand()%6;
```

以下敘述，可取 1~6 的亂數。

```
int a=1+rand()%6;
```

以下敘述，可取 97~122 的亂數。而 97~122 為小寫字元，所以可輸出小寫字元。

```
int a=97+rand()%26;
printf("%c",a);
```

自我練習

題號	題目	結果
1	請將以上程式的『 <code>srand(time(0));</code> 』去掉，並觀察執行結果，是否每次都一樣。	
2	若函式 <code>rand()</code> 的回傳值為一介於 0 和 10000 之間的亂數，下列那個運算式可產生介於 100 和 1000 之間的任意數 (包含 100 和 1000) ? (APCS10603) (A) <code>rand() % 900 + 100</code> (B) <code>rand() % 1000 + 1</code> (C) <code>rand() % 899 + 101</code> (D) <code>rand() % 901 + 100</code>	
3	<pre>int main() { int a; srand(time(0)); a=rand()%14-5; }</pre> 以上 a 不可能 (A)-3 (B)0 (C)6 (D)9	
4	<pre>int main() { double a; srand(time(0)); a=double(rand()%2001-1000)/1000.0; printf("The Random Number is %f .\n", a); }</pre> 以上 a 不可能 (A)-3.2 (B)0.1 (C)-0.3 (D)0.8	
5	<pre>int main() { srand(time(0)); int a=97+rand()%26; printf("%c",a); }</pre> 以上輸出有可能 (A)A (B)1 (C)98 (D)d	

5-4 聲音的產生

要讓電腦發出聲音，可使用 `_beep()` 函式，此函式放在 `stdlib.h`，所以要載入標頭檔 `#include <stdlib.h>`，其語法如下：

```
void _beep(unsigned int x,unsigned int y);
```

x 是要發出的頻率，單位是 Hz，y 是發聲的時間，單位是 ms。其次，鋼琴鍵盤頻率對照表如表 5-2：

► 表 5-2 音階頻率表

	n	1	2	3	4	5	6	7	8	9	10	11	12
音階	音符	(Do)	(Do#)	(Re)	(Re#)	(Mi)	(Fa)	(Fa#)	(So)	(So#)	(La)	(La#)	(Si)
低音	頻率 (Hz)	262	277	294	311	330	349	370	392	415	440	466	494
中音	頻率 (Hz)	523	554	587	622	659	698	740	784	831	880	932	988
高音	頻率 (Hz)	1046	1109	1175	1245	1318	1397	1480	1568	1661	1760	1865	1976

例如，以下敘述可發出『Do』的音，且持續 0.5 秒。

```
_beep(523, 500);
```

範例5-4a

示範發出 Do、Re、Me、Fa…等八個音。

程式列印

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    _beep(523, 500);
    _beep(587, 500);
    _beep(659, 500);
    _beep(698, 500);
    _beep(784, 500);
    _beep(880, 500);
    _beep(988, 500);
    _beep(1046, 500);
    return 0;
}
```

🎵 演奏音樂

要讓電腦演奏音樂，那就要先下載一個簡譜，如圖 5-8。



圖 5-8 小毛驢簡譜

本例一分鐘 80 拍，所以 1 拍的時間是， $60000\text{ms}/80$ ，以一個四分音符為 1 拍，所以上圖一個八分音符所佔的時間是 $60000\text{ms}/(2*80)$ ，每小節是 2 拍。其次，上圖，所有音符的最小時間是八分音符，所以我就取八分音符的時間為 1 個單位。第 1 音『1 我』半拍，取 1 個單位；『5- 騎』是 2 拍，取 4 個單位。

範例 5-4b

示範演奏音樂。本範例僅作 4 小節，『我有一隻小毛驢我從來也不騎』。

👉 程式列印

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    int t=60000/160;
    _beep(523,t);
    _beep(523,t);
    _beep(523,t);
    _beep(659,t);
    _beep(784,t);
    _beep(784,t);
    _beep(784,t);
    _beep(784,t);
    _beep(880,t);
    _beep(880,t);
    _beep(880,t);
    _beep(1064,t);
    _beep(784,4*t);
    return 0;
}
```

👉 補充說明

1. 請熟悉以上發音方式，第七章還會介紹如何製作電子琴。
2. 以上僅輸出一小節，程式就有點冗長，那寫完一首歌不就累垮。請放心，待介紹迴圈與陣列就會明瞭，本書於範例 9-5g，將會以陣列與迴圈重做本範例。

👉 自我練習

1. 請繼續完成以上音樂。
2. 請用輸出指令讓輸出音樂的同時，螢幕也輸出歌詞，那就是 KTV 了。

課後習題

題號	題目	結果
1	<pre>main() { int a=65; printf("%d\n",a); printf("%c\n",a); printf("%f\n",a); }</pre>	
2	<pre>int main() { int a=5; double b=3.14; printf("%d\n",a);// printf("%f\n",a);// printf("%d\n",b);// printf("%f\n",b);// printf("%lf\n",b);// }</pre>	
3	<pre>int main() { int a=3.4; double b=3; printf("%d\n",a);// printf("%f\n",a);// printf("%d\n",b);// printf("%f\n",b);// printf("%lf\n",b);// }</pre>	

4	挑出對的敘述。 <pre> int main() { char word[]="good"; printf("%s",word);//A printf("%c",word);//B printf("%s",word)//C printf("%s",word[]);//D return 0; } </pre>	
5	<pre> int main() { int a=97; printf("%d\n", a); printf("%c\n", a); printf("%c\n", (char)a); char b='a'; printf("%c\n", b); printf("%d\n", b); printf("%d\n", (int)b); char c='ab'; printf("%c\n", c); char d='abc'; printf("%c\n", d); char e='a '; printf("%c\n", e); } </pre>	
6	<pre> int main() { int a=3,b=55; double c1=3.4,c2=15.678; printf("a =%8d,b =%8d\n",a,b); printf("c1=%8.2f,c2=%8.2f",c1,c2); return 0; } </pre> a = 3,b = 55 c1= 3.40,c2= 15.68	

7	請問哪一敘述錯誤？ <pre> int main() { char a,b; printf("Please press any char: "); scanf("%c",&a); printf("Your char is %c", a); scanf("%c",b); return 0; } </pre>	
8	請問連續輸入 a b，將會輸出什麼？ <pre> int main() { char a,b; printf("Please press any char: "); scanf("%c", &a); printf("Your char is %c", a); printf("Please press any char: "); scanf("%c", &b); printf("Your char is %c", b); return 0; } </pre>	
9	請問連續輸入 a b，將會輸出什麼？ <pre> #include <stdio.h> #include <stdlib.h> int main() { char a,b; printf("Please press any char: "); scanf("%c", &a); printf("Your char is %c", a); printf("Please press any char: "); scanf(" %c", &b); printf("Your char is %c", b); } </pre>	
10	程式執行後，輸入「ab」，寫出輸出結果。 <pre> int main() { char a,b; printf("Please press two char: "); scanf("%c %c", &a,&b); printf("Your char is %c\n", a); printf("Your char is %c", b); } </pre>	

11	程式執行後，輸入「ab」，寫出輸出結果。 <pre> int main() { char a,b; printf("Please press two char: "); scanf("%c, %c", &a,&b); printf("Your char is %c\n", a); printf("Your char is %c", b); } </pre>	
12	程式執行後，輸入「a,b」，寫出輸出結果。 <pre> int main() { char a,b; printf("Please press two char: "); scanf("%c, %c", &a,&b); printf("Your char is %c\n", a); printf("Your char is %c", b); } </pre>	
13	以下程式輸入 3,4，寫出輸出結果。 <pre> int main() { int a,b,d; printf("input a,b:"); d=scanf("%d,%d",&a,&b); //請留意中間是逗號 printf("a+b=%d\n",a+b); printf("d=%d\n",d); // } </pre>	
14	以下程式輸入 3 4，寫出輸出結果。 <pre> int main() { int a,b,d; printf("input a b:"); d=scanf("%d %d",&a,&b); //請留意中間是空白 printf("a+b=%d\n",a+b); printf("d=%d\n",d); return 0; } </pre>	

運算子及運算式

前面我們已經使用 $c = a * b$ 計算長方形面積，「a」、「b」稱為運算元 (Operand)；「*」稱為運算子 (Operator)；「 $a * b$ 」稱為運算式 (Expression)，運算式是一種結合序列的運算子、運算元的敘述，用來計算產生結果；「 $c = a * b$ 」稱為敘述 (Statement)。敘述是程式組成的基本元素，是程式的主體；每個敘述執行有意義的工作。程式是由許多不同意義的敘述組成的有目的的工作程序。以上內容將在本章介紹。

6-1 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。C 語言的運算子分為五大類，分別是算術 (Arithmetic) 運算子、複合指派運算子、關係運算子、邏輯運算子、位元運算子。

算術運算子 (Arithmetic operators)

算術運算子用來執行一般的數學運算，包括指派 (=)、取正負數 (+/-)、加 (+)、減 (-)、乘 (*)、除 (/)、取餘數 (%)、遞增 (++) 及遞減 (--) 等，表 6-1 是 C 語言的算術運算子列表：

► 表 6-1 C 語言算術運算子

運算子	定義	優先順序	結合律
=	指派或賦值	14	由右至左
a++,a--	後置遞增，遞減	1	由左至右
++a,--a	前置遞增，遞減	2	由右至左
+, -	正負號，一元運算子	2	由右至左
*	乘法運算	3	由左至右

運算子	定義	優先順序	結合律
/	除法運算	3	由左至右
%	求餘數 (Modulus)	3	由左至右
+, -	加法, 減法運算	4	由左至右

以上運算子的定義、優先順序、結合律分別說明如下：

☛ 指派運算子 (Assignment operators)

指派運算子的符號為 (=)，又稱為賦值運算子，其作用為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` 是將 `a+b` 的值先運算，再指派給 `sum`。

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

上式與數學的等號是不同的，所以不要一直糾結為什麼 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 ;
```

此為一個不合法的敘述，但以下敘述，將常數 8 指派給變數 `x` 是合法的。

```
x = 8 ;
```

☛ 五則運算

以下是一些簡單算術五則運算，請自行鍵入程式，寫出執行結果。

```
int a=5,b=4;
printf("%d\n",a+b); //9
printf("%d\n",a-b); //1
printf("%d\n",a*b); //20
printf("%d\n",a/b); //1
printf("%d\n",a%b); //1
```

☛ 整數除法或實數除法

C 語言的除法運算，只有被除數與除數的型態均為整數，才是整數除法，商的型態為整數。其次，只要被除數與除數有一型態為實數即為實數除法，得到實數商。例如：

```

int x=5, y=4;
double xf=5,yf=4;
printf("%d\n",x/y);    //1 兩者都是 int
printf("%f\n",xf/y);  //1.25
printf("%f\n",x/yf);  //1.25
printf("%f\n",xf/yf); //1.25

```

自我練習

題號	題目	結果
1	<pre> int main() { int a,b=7,c=2; double d; d=b/c; printf("%lf\n",d); return 0; } </pre>	
2	<pre> int main() { int a,b=7,c=2; double d=2,e; e=b/d; printf("%lf\n",e); a=b/d; printf("%d\n",a); return 0; } </pre>	
3	<pre> int main(){// 技藝競賽 107 int a=2,b=3; int c=4,d=5; int val=b/a+c/b+d/b; printf("%d",val); } </pre> (A) 3 (B) 4 (C) 5 (D) 6	
4	下列 C 語言程式碼片段執行後，x 與 y 的結果為何？ (統測 111) <pre> int x, a = 7, b = 2; float y; x = a/b; y = (float)a/b; </pre>	

5	請問下列左右兩個 C 語言片段程式執行後，b 變數的值分別為何？ (技藝競賽 112) (A)3, 2.5 (B)2.5, 2.5 (C)3, 2 (D)2, 2.5 int a = 5; int a=5; float b; float b; b=a/2; b=a/2.; //請留意是2.0	
6	C 語言指令 printf("%d", 4/3*3); 答案為？ (技藝競賽程式組 109) (A) 4 (B) 3 (C) 3.99999 (D) 0.44444	
7	以下程式用來計算球體體積，計算結果錯誤，請問錯在哪一行？ (技藝競賽程式組 108) <pre> 1 #include<stdio.h> 2 int main(){ 3 double ratio, pi, r, vol; 4 ratio=4/3; 5 pi=3.1416; 6 scanf("%lf", &r); 7 vol=ratio*pi*r*r*r; 8 printf("%lf", vol); 9 return 0; 10 }</pre>	

☛ 整數除法與取餘數的應用

整數除法與取餘數可以將一個整數分解為數個數字。例如，若要使用七段顯示器顯示 152，就要將此數字分解為 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

```

以上是已知數字長度，可以從左邊分解。以下是從末端一直先取餘數，再相除。

```

int a=152;
int a3=152%10;//個位數
a=a/10;
int a2=a%10;//十位數
a=a/10;
int a1=a ;//百位數
printf("%d,%d,%d",a1,a2,a3);

```

從末端分解的優點是，可以由商是否為零，判斷是否結束。等到學了決策與迴圈指令就能寫程式完成任意長度整數的分解。

☛ 取餘數

若要讓一個數字在累加的過程中，保持一定的循環，那也是用取餘數。例如：

```
while(1)  { //無窮迴圈，請看第六章
    b=a[i];
    i=(i+1) %4;
}
```

那 i 就永遠在 0,1,2,3 循環。

☞ 自我練習

題號	題目	結果
1	<pre>int main(){//技藝競賽108 int n=76; printf("%d",n/50); n=n%50; printf("%d",n/10); n=n%10; printf("%d",n/5); n=n%5; printf("%d\n",n); }</pre> (A) 1222 (B) 1221 (C) 1211 (D) 1111	
2	<pre>int main(){//數字分解 int a=1528; int a1=a/1000;//千位數 int a2=(a-a1*1000)/100;//百位數 int a3=(a-a1*1000-a2*100)/10;//百位數 int a4=a %10;//個位數 printf("%d" a1+a2+a3+a4); }</pre>	
3	<pre>int main(){//數字分解 int a=1528; int a1=a%10;//個位數 a=a/10; int a2=a%10;//十位數 a=a/10; int a3=a%10 ;//百位數 a=a/10; int a4=a; printf("%d%d%d%d,%d",a1,a2,a3,a4,a1+a2+a3+a4); }</pre>	

4	<pre> int main(){//轉為秒數 int a=12345; int s=a%60; int m=(a/60)%60; int h=a/3600; printf("%d:%d:%d",h,m,s); } </pre>	
5	<pre> int main(){//數字分解 int a=25678; int a1=a%100; a=a/10; int a2=a%100; a=a/10; int a3=a%100; a=a/10; int a4=a%100; printf("%d,%d,%d,%d",a4,a3,a2,a1); } </pre>	
6	<pre> int main(){//數字分解 int a=25678; int a1=a%100; a=a/100; int a2=a%100; a=a/100; int a3=a%100; printf("%d,%d,%d",a3,a2,a1); } </pre>	
7	<pre> int main(){//6-5-5數字分解 int a=12345; int a1=a/10000; int a2=a %10;//個位數 printf("%d",a1+a2); } </pre>	
8	<pre> int main(){//6-5-6數字分解 int a=12345; int a1=a/1000; int a2=a %100;//個位數 printf("%d",a1+a2); } </pre>	

9	<pre>int main() { //6-5-6 數字分解 int a=123456; int a1=a/1000; int a2=a %1000; //個位數 printf("%d",a1+a2); }</pre>	
10	<pre>int main() { //時間計算 int h1=10,m1=10,h2=8,m2=50,h,m,t1,t2,t; t1=h1*60+m1;t2=h2*60+m2; t=t1-t2; h=t/60;m=t%60; printf("%2d:%2d %3d",h,m,t); }</pre>	

☛ 遞增(++)及遞減(--)

遞增 (++) 及遞減 (--) 是 C 語言特有的運算子，此二個運算子又分為前置與後置，前置是運算子在運算元之前，例如：

```
a=1; ++a; printf("%d\n",a); //2
```

後置是運算子在運算元之後，例如：

```
a=1;a++; printf("%d\n",a); //2
```

原則上不論 ++a 或 a++ 都是將 a 值加 1 並放回 a。但若是：

```
a=1;b=++a; // a=a+1;b=a; a is 2 b is 2
```

和

```
a=1;b=a++ ; // b=a;a=a+1;a is 2 b as 1
```

則其 a 值均會加 1，但 b 值會有差異，前置 b 值會得到加 1 的結果，後者只能得原 a 值。

👉 自我練習

題號	題目	結果
1	<pre>int main(){//技藝競賽程式組108 int a=5; printf("%d",a); printf("%d",a++); printf("%d\n",a); }</pre> (A) 556 (B) 566 (C) 666 (D) 555	
2	<pre>int main(){//技藝競賽程式組108 int a=5; printf("%d",a); printf("%d",++a); printf("%d\n",a); }</pre> (A) 556 (B) 566 (C) 666 (D) 555	

☞ 次方與根號

值得注意的是，C 語言的數學運算子只有以上 +, -, *, /, %。若所需運算沒有運算子，例如，要次方或根號運算，那要如何處理呢？答案是要使用數值處理函式。例如，以下 pow() 函式可計算 3 的 2 次方：

```
a = pow(3,2); //include<math.h>
```

若要執行根號運算，則應使用數值處理函式的 sqrt() 函式。如以下敘述是計算 9 的開平方。

```
x= sqrt(9); //include<math.h>
```

但以上函式都放在 math.h，所以都要在程式最前面引入 math.h 標頭檔使用，如以下敘述：

```
#include<math.h>
```

✂ 複合指派運算子 (Compound assignment operators)

結合指派與算術的運算子稱為複合指派運算子，此乃 C 語言特有的運算子。例如：程式設計者常會鍵入 sum=sum+5，程式語言乃制定此一複合指派運算子「+=」。所以上述 sum=sum+5，即可寫成 sum += 5。表 6-2 是 C 語言常用的複合指派運算子：

► 表 6-2 複合指派運算子

運算子	定義	優先順序	結合律
=	指派內容	14	由右至左
+=	相加之後再指派內容	14	由右至左
-=	相減之後再指派內容	14	由右至左
*=	相乘之後再指派內容	14	由右至左
/=	相除之後再指派內容	14	由右至左
%=	取餘之後再指派內容	14	由右至左

例如：

```
int sum=0,a=5;
sum+=5;
printf("%d\n",sum); //5
sum/=a;
printf("%d\n",sum); //1
```

⌘ 比較(Comparison)運算子

比較運算子又稱為關係運算子 (Relational Operators)，用於資料之間的大小比較，比較的結果可得到 1(true) 或 0(false)，表 6-3 是 C 語言中的比較運算子符號。

► 表 6-3 比較運算子

運算子	定義	優先順序	結合律
<	小於	6	由左至右
>	大於	6	由左至右
<=	小於等於	6	由左至右
>=	大於等於	6	由左至右
==	等於	7	由左至右
!=	不等於	7	由左至右

例如：

```
int a=8,b=5;
printf("%d\n",a>b); //1
printf("%d\n",a==b); //0
printf("%d\n",a>=b); //1
printf("%d\n",a<=b); //0
printf("%d\n",a!=b); //1
```

🔗 邏輯運算子(Logical Operators)

當同一個運算式要同時存在兩個以上的關係運算子時，每兩個關係運算子之間必須使用邏輯運算子連結。例如：您要找男生且年齡大於 40，此一選擇就同時含有兩個關係運算式，此時就要使用邏輯運算子給予連結。C 的邏輯運算子有否定 not 『!』、邏輯 and 『&&』、邏輯 or 『||』，如表 6-4 所示：

► 表 6-4 邏輯運算子

運算子	定義	優先順序	結合律
!	邏輯否定 not 運算	2	由右至左
&&	邏輯 and 運算	11	由左至右
	邏輯 or 運算	12	由左至右



『!』是取邏輯否定，也就是將『true』轉為『false』，『false』轉為『true』。例如：

```
int a=8,b=5,c=3;
printf("%d\n", (a>b)); //1
printf("%d\n", !(a>b)); //0
printf("%d\n", !(a==b)); //1
```



&&

『&&』是邏輯 and，兩件事都 true，結果才是 true，其餘都是 false，其真值表如表 6-5：

► 表 6-5 and 真值表

事件1	事件2	結果
true	true	true
true	false	false
false	true	false
false	false	false

例如：

```
int a=8,b=5,c=3;
printf("%d\n", a>b && b>c ); //1
printf("%d\n", a>b && b==c ); //0
```



『||』是邏輯 or，也就是兩件事只要有一件為 true，那就為 true；只有兩件事全為 false，才為 false。其真值表如表 6-6：

► 表 6-6 or 真值表

事件1	事件2	結果
true	true	true
true	false	true
false	true	true
false	false	false

例如：

```
int a=8,b=5,c=3;
printf("%d\n",a>b || b==c );//1
printf("%d\n",a==b || b==c );//0
```

題號	題目	結果
1	<pre>int main() { //區間判別 int x=3; printf("%d\n",x>1 && x<=6); x=9;printf("%d",x>1 && x<=6); } }</pre>	
2	<pre>int main() { //三角形判別 int a=3,b=4,c=5; printf("%d", (a+b)>c && (b+c)>a && (c+a)>b); }</pre>	
3	若要邏輯判斷式 $!(X1 \parallel X2)$ 計算結果為真 (True)，則 X1 與 X2 的值分別應為何？ (APCS 106) (A) X1 為 False，X2 為 False (B) X1 為 True，X2 為 True (C) X1 為 True，X2 為 False (D) X1 為 False，X2 為 True	
4	<pre>int main() { //區間判別 int x=7; int d=(x>3 x <-2); printf("%d",d); }</pre>	

5	<pre>int main() { //重合判別 int a1=1,b1=2,c1=3,a2=3,b2=6,c2=1; int d=(a1*b2==a2*b1 && b1*c2==b2*c1); printf("%d",d); }</pre>	
6	<pre>int main() { //歪斜判別 int a1=1,b1=2,c1=3,a2=3,b2=6,c2=1; int d=(a1*b2==a2*b1 && b1*c2!=b2*c1); printf("%d",d); }</pre>	

⌘ 位元運算子(Bitwise operation)

在介紹位元運算子前，請您思考一下複選題答案應該如何數位化？例如，答案有可能 ABC，也有可能 BE，那如何將這些資料數位化，選用適當的變數型態處理呢？是不是用一個字串表示呢？例如：以

```
a="ABD"
```

表示。這樣表示當然也可以，但每個答案就要占 1 ~ 5 byte(1 byte=8bit)，但 "ABD" 若是以 11010 表示呢？"A" 就以 10000 表示，這樣就可以分別用 26(11010=2⁴+2³+2¹=26) 與 16(10000=2⁴=16) 表示 "ABD" 與 "A"，所以只要 1byte 就可以記錄所有答案。好的，以上已經完成非常精簡的資料數位化方式，那電腦如何評分呢？答案就是電腦也備有位元運算子，可以錙銖必較的處理每個位元 (bit)。

位元 (Bitwise) 運算子是將一個 8 位元的整數逐位元進行運算，此類運算子還可以分為兩類：位移 (Shift) 運算子與布林運算子。位移運算子可以用來將各個位元向左或是向右移；布林運算子則可以逐位元進行布林運算。表 6-7 是 C 語言位元操作運算子的列表：

► 表 6-7 位元運算子

運算子	定義	結合律
~	逐位元 not 運算	由右至左
<<	逐位元向左位移	由左至右
>>	逐位元向右位移	由左至右
&	逐位元 and 運算	由左至右
^	逐位元 xor 運算	由左至右
	逐位元 or 運算	由左至右



『~』是位元 not 運算，not 是將 0 變 1，1 變 0。例如：

```
int a,b//本單元適用
a=1;
printf("%d",~a); //-2
```

爲什麼答案是 -2 呢？這有點難懂，筆者簡單說明如下，因爲電腦是以 2 補數表示負數。a=1，分解爲 2 進位是 00000001

```
~a= not 00000001=11111110
```

首位元是 1 表示負數，那到底負多少，就取 2 補數。2 補數的步驟是先取 1 補數再加 1。其次，所謂 1 補數是『逐位元將 1 變 0，0 變 1』，所以上面 11111110 的 1 補數是

```
00000001
```

將 1 補數再加 1，就是 2 補數，上面 00000001 加 1 就是 00000010，其大小是 2。上面首位元符號既然是『1』，那就表示其爲負數，所以 11111110 代表此數是 -2，所以輸出 -2，此即爲二補數的觀念。

自我練習

題號	題目	結果
1	— 3 的 2 補數以 8 位元表示，其值為何？ (A) 11111101 (B) 00000011 (C) 11111110 (D) 1111011	
2	2 補數的 10000001 代表 (A) -128 (B) -1 (C) -2 (D) -127	
3	有關數字系統 2 的補數 (2's)，以二進制表示負 13(-13)，下列何者正確？ (技藝競賽程式組 110) (A) 0000 1101 (B) 1000 1101 (C) 1111 1101 (D) 1111 0011	
4	有二個 2 的補數數字相加 1111 1001(2's) + 1111 1101(2's) 的結果爲下列何者？ (技藝競賽程式組 110) (A) -22 (B) -21 (C) -10 (D) -9	
5	下列 16 進位數，哪一個是用 2 的補數 (two's complement) 來表示的負數？ (技藝競賽程式組 110) (A) :4E (B) :66 (C) :7D (D) :A7	
6		

7	<pre>int main(){ printf("%d",1&&2); }</pre> (A) 0 (B) 1 (C) 3 (D) 4	(技藝競賽程式組 107)
---	--	---------------

自我練習

1. 請輸入以下程式，並觀察執行結果，且自行使用以上原理推敲，看答案是否一致。

```
int a;
a=-3;
printf("%d",~a);
a=-127;
printf("%d",~a);
```

☛ <<

『<<』是左移運算子，例如：

```
a=1;
printf("%d",a<<1);//2    1(00000001) 向左移位1位元(00000010)，結果是2
a=1;
printf("%d",a<<2);//4    1(00000001) 向左移位2位元(00000100)，結果是4
```

☛ >>

『>>』是右移運算子，例如：

```
a=4;
a=a>>1;
printf("%d",a);//2    4向右移位1位元，結果是2
a=7;
a=a>>2;
printf("%d",a);//1
```

☛ &

『&』是位元 and 運算，and 運算真值表如表 6-12，當 a 與 b 同時為 1，c 才得到 1：

► 表 6-12 位元 and 真值表

a	b	c= a & b
0	0	0
0	1	0
1	0	0
1	1	1

例如：

```
a=1;b=5;
printf("%d",a&b); //00000001 & 00000101=00000001=1
```

☞ ^

『^』是位元 xor 運算，xor 運算真值表如表 6-13，當 a 與 b 不同時，c 才得到 1。

► 表 6-13 位元 xor 真值表

a	b	c=a^b
0	0	0
0	1	1
1	0	1
1	1	0

例如：

```
a=1;b=5 ;
printf("%d",a^b); //00000001 ^00000101=00000100=4
```

☞ |

『|』是位元 or 運算，or 運算真值表如表 6-14，當 a 與 b 有一為 1，c 就得到 1。

► 表 6-14 位元 or 真值表

a	b	c=a b
0	0	0
0	1	1
1	0	1
1	1	1

例如：

```
a=1;b=5;
printf("%d",a|b); //00000001 | 00000101 =00000101=5
```

自我練習

題號	題目	結果
1	<pre>int main(){ printf("%d",1&2); }</pre> (技藝競賽程式組107) (A) 0 (B) 1 (C) 3 (D) 4 答案：(A)	
2	<pre>int main(){ printf("%d",1&&2); }</pre> (技藝競賽程式組107) (A) 0 (B) 1 (C) 3 (D) 4 答案：(B)	
3	阿華想要了解 C 語言程式 if 條件敘述中常用的運算子 & 與 && 的不同，撰寫如下程式，下列何者為程式執行結果？ (統測 112 A) <pre>1 #include <stdio.h> 2 3 int main() { 4 5 int a=0x0a; 6 int b=0x05; 7 8 if(a & b) 9 printf("a&b=%d\n", a&b); 10 else 11 printf("a&&b=%d\n", a&&b); 12 13 return 0; 14 }</pre> (A) a&&b = 1 (B) a&&b = 0 (C) a&b = 1 (D) a&b = 0 解： (1) && 是邏輯 and (A) 非 0 都是 1 (B) 兩個 1 才是 1，1 && 1 = 1 (C) 1 && 0 = 0 (2) & 是逐位元 and，請鍵入以下程式。 <pre>#include <stdlib.h> #include <stdio.h> int main() {</pre>	

	<pre> int a=0x0a;//1010 int b=0x05;//0101 int d=1; int e=0; //邏輯and 1.非0都是1 2.兩個1才是1 1 &1 =1 3. 1&0=0 printf("a&d=%d\n", a&d);//1 printf("b&d=%d\n", b&d);//1 printf("b&e=%d\n", b&e);//0 if(a & b) //0 false 執行else &逐位元 and printf("a&b=%d\n", a&b); else printf("a&b=%d\n", a&b);//1 return 0; } </pre>	
4	<pre> int main(){//技藝競賽程式組108 int a=1; int b=2; int c=a<<b; printf("%d",c); } </pre> (A) 1 (B) 2 (C) -3 (D) 4	
5	<pre> int main(){//技藝競賽程式組108 int n=255; int mask=15; printf("%d",n & mask); } </pre> (A) 0 (B) 1 (C) 5 (D) 15	
6	<pre> int main(){//技藝競賽程式組108 int n=255; int mask=15; printf("%d",n & mask); } </pre> (A) 0 (B) 1 (C) 5 (D) 15	
7	<pre> int a=3,b=7;//統測111 printf("%d",a^b); </pre> (A) 4 (B) 1 (C) 2 (D) 3	

8	<pre>int main(){//技藝競賽程式組108 int a=1; int b=2; int c=a^b; printf("%d",c); }</pre> (A) 0 (B) 1 (C) 2 (D) 3	
9	C 語言的指令 <code>1&&2</code> 與 <code>1&2</code> 的值分別為何？（技藝競賽程式組 107） (A) 1, 2 (B) 1, 0 (C) 0, 1 (D) 2, 1	
10	C 語言的指令 <code>1 2</code> 與 <code>1 2</code> 的值分別為何？（技藝競賽程式組 107） (A) 1, 3 (B) 3, 1 (C) 1, 2 (D) 2, 1	

☞ 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：

```
x=a+b*c;
```

在數學裡，我們定義先乘除再加減。程式語言也是相同，必須定義這些運算子的優先順序，C 語言的運算子優先順序如表 6-15：

► 表 6-15 運算子優先順序表

優先順序	運算子	說明	結合律
1	++,--	後置遞增與遞減。例如 <code>a++,a--</code> 。	由左至右
2	++,-- +,- !	前置遞增與遞減。例如 <code>++a,--a</code> 。 取正或取負。 邏輯否定。	由右至左
3	*,/,%	算術乘、除、取餘。	由左至右
4	+, -	算術加、減	
6	>, >=, <, <=	關係運算的大小。	
7	==, !=	關係運算的相等或不相等。	
11	&&	邏輯的 and。	
12		邏輯的 or。	由右至左
14	=, +=, -=, *=, /=, %=	指派與複合指派。	

由以上各運算子的『優先順序』可知，乘號『*』的優先順序是第 3，而加號『+』則是第 4。所以，以上敘述：

```
x=a+b*c;
```

的結果與

```
x=(a+(b*c));
```

的效果相同。同理，

```
x=a>b && b>z;
```

的結果同義於

```
x=(a>b) && (b>z);
```

自我練習

題號	題目	結果
1	假設在 C 語言程式的某個運算式中，有好幾個運算子，下列何者的優先權最高？ (技藝競賽修護組) (A) && (B) == (C) >> (D) ()	
2	依 C 語言之運算子優先權順序，下列運算式的結果，何者為真 (true)？ (統測 111) (A) !(1 != 3) 1 = 3 (B) 1 != 3 && !(1 = 3) (C) !(1 < 3) 1 > = 3 (D) 1 < 3 && !(1 > = 3)	
3	執行完下列片段程式後，Num1 與 Num2 的數值分別為何？(統測 112) 1 int Num1 =10, Num2 = 5; 2 int Num3 =3; 3 Num1 = Num1 << Num3-1; 4 Num2 = Num2 * Num1 >> 1; (A) Num1 = 79、Num2 = 197 (B) Num1 = 79、Num2 = 195 (C) Num1 = 40、Num2 = 200 (D) Num1 = 40、Num2 = 100 解：請鍵入以下程式。 int main() { int Num1 =10, Num2 = 5; int Num3 =3; Num1 = Num1 << Num3-1; //00001010 <<2 左移2位元 101000 =40	

	<pre> Num2 = Num2 * Num1 >> 1; printf("%d\n", Num1); //40 printf("%d\n", Num2); //100 (5*40)>>1=100 return 0; } </pre>	
4	<pre> int main() { int a,b,c; a=1;b=2;b=3; a=a+a*2/2-3%2; printf("%d\n",a); a=5;b=4; c= a>2 && !(b>3) (a==3); printf("%d\n",c); return 0; } </pre>	

✂ 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『由左至右』的左結合或『由右至左』的右結合。例如：

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

同樣是減號(-)，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

```
a=b=c=2;
```

指派運算子的結合律是由右至左，所以以上式子同義於：

```
(a=(b=(c=2))) ;
```

以上式子，a、b、c的結果都是2。

👉 自我練習

1. 鍵入、執行以下程式，並寫出執行結果。

題號	題目	結果
1	<pre>int main(){//結合律 int a = 96,b=3,c=2; a=a/b/c; printf("%d\n",a); a=a-b-c; printf("%d\n",a); a=b=c; printf("%d\n",b); }</pre>	

2. 若 a, b, c, d, e 均為整數變數，下列哪個算式計算結果與 $a+b*c-e$ 計算結果相同？（APCS 試題）
- (A) $((a+b)*c)-e$
- (B) $((a+b)*(c-e))$
- (C) $((a+(b*c))-e)$
- (D) $(a+((b*c)-e))$

6-2 運算式

任何一個可求得一個值的式子，都稱為一個運算式，例如 $5+3$ 會傳回一個數值，所以 $5+3$ 是一個運算式。以下是一些合法的運算式，這些運算式通常放在 `printf("%d", 運算式)`、第七章的 `if(運算式)`、第八章的 `for( ; 運算式; )` 及 `while(運算式)`。一般而言，可以放在等號右邊的東西，都可以稱為運算式。

```
6           // 傳回6
4 + 5 ;     // 傳回9
sum > 3 ;   // 傳回 true or false
a==b;      // 傳回true or false
a + b ;    // 傳回8
```

敘述 (Statement)

凡是控制執行的順序、對運算式取值或不作任何事，均可稱為敘述。所有的敘述都要以分號(;)作結束，例如：以下式子即是一個敘述。

```
sum = sum + 1 ;
```

於 C 語言中，若前一個敘述未以分號結束，則錯誤訊息通常出現在下一個敘述的開頭。

☞ 敘述區塊 (Block Statement) 或複合敘述 (Compound Statement)

敘述區塊亦稱複合敘述，在任何可以放置單一敘述的地方，你都能放置敘述區塊。一個複合敘述是由兩個大括號組合而成，如下所示：

```
{
    t = a ;
    a = b ;
    b = t ;
}
```

請留意，大括號之後不可再加分號，且兩個刮號對稱，方向都是朝程式區塊。

☞ 註解 (Comments)

適當的程式註解才能增加程式的可讀性，C/C++ 語言的註解方式有兩種，分別是多列註解與單列註解。首先，多列註解如下：

```
/* 我是註解 */
```

上式「/*」符號以後的字串視為註解，編譯程式不予處理，直到遇到「*/」為止。其次，單列註解如下：

```
//我是註解
```

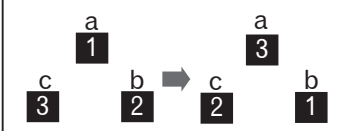
同一列中，雙斜線（//）後面的文字均視為註解，編譯器均不予處理。例如，以下二式分號右邊均為註解：

```
x= x+ 3;          /* x值加3 */
sum = sum + y;    // 將y之值加至sum
```

前者，因為可超過兩列，較適合編寫較長的註解；後者，則僅能寫在同一列。

☞ 縮排與空行

除了適當的註解，程式設計應善用縮排與空行，才能使你的程式讓人易於閱讀。因為有了縮排（請使用 Tab 鍵，而不是自行按空白鍵，因為空白鍵很費力，且無法完全對齊），程式才有層次感。其次，空行則可使段落更加明顯。所以，本書範例則都遵守這些規定，請讀者自行觀察。

題號	題目	結果
1	在 C 語言中，「//」代表下列何種格式？（技藝競賽修護組 110） (A) 開始 (B) 註解 (C) 分隔 (D) 結束。	
2	在下圖中，a,b,c 的值分別為 1,2,3，請問下面四個指令的順序為何可以將 a,b,c 的值順時鐘調換？（技藝競賽程式組 110）  ① b*=*temp; ② c*=*b; ③ a*=*c; ④ temp*=*a; (A) ①②③④ (B) ②③①④ (C) ②③④① (D) ④③②①	
3	<pre>int main() { //兩點距離 int x1,y1,x2,y2;double d; x1=3;y1=0; x2=0;y2=4; d=sqrt(pow((x1-x2),2)+pow((y1-y2),2)); printf("%f\n",d); }</pre>	
4	<pre>int main() { //三角形面積計算 int a = 3, b = 4,c=5; double d=(a+b+c)/2; double e=sqrt(d*(d-a)*(d-b)*(d-c)); printf("%f", e);// }</pre>	
5	<pre>int main() { //台斤轉公斤 int a = 3; double e=a*5/3; printf("%f", e); }</pre>	
6	<pre>int main() { //公斤轉台斤 int a = 28;double b; b=a*9/5+32; printf("%lf\n",b); double c=(b-32)*5/9; printf("%lf\n",c); }</pre>	
7	<pre>int main() { //三角形面積計算 int x1=0,y1=0,x2=3,y2=0,x3=0,y3=4; double e=(x1*y2+x2*y3+x3*y1-x2*y1-x3*y2-x1*y3)/2; printf("%f", e); return 0; }</pre>	

8	<pre> int main() { //時速秒速互轉 int a=1; double b=(double)a*1000/(60*60); printf("%lf\n", b); double c=1; b=c*1000/(60*60); printf("%lf\n", b); b=a*1000/(60*60); //0 printf("%lf\n", b); return 0; } </pre>	
9	<pre> int main() { //點(1,2)與直線3x+4y+c=0距離 int a=3,b=4,c=5; int x=1,y=2; double d=fabs(a*m+b*n+5)/sqrt(a*a+b*b); //abs()傳回整數絕對值，fabs()傳回實數絕對值 printf("%f",d); return 0; } </pre>	
10	<pre> int main() { //給定兩點，造出方程式（兩點式） int x1=2,y1=1; int x2=4,y2=6; int d=y2-y1; int f=x2-x1; printf("%dx%+dy%+d=0",d,-f,-d*x1+f*y1); return 0; } </pre>	
11	<pre> int main() { //點斜式 x1,y1=1,2 ;m=3 a=-m b=1 c=m*x1-y1 print('%dx%+dy%+d=0'%(a,b,c)) } </pre>	
12	<pre> int main() { //截距式 a,b=4,3 print('%dx%+dy%+d=0'%(b,a,-a*b)); } </pre>	

13	<pre> int main() { //解一元二次方程式 int a,b,c; //係數 double d,x1,x2;a=2;b=-7;c=3; d=sqrt(b*b-4*a*c); //開根號 x1=(-b+d)/(2*a); //留意所有括號不能漏掉 </pre>	
	<pre> x2=(-b-d)/(2*a); //留意所有括號不能漏掉 printf("x1=%f\n",x1); printf("x2=%f\n",x2); return 0; } </pre>	
14	<pre> int main(void) { //解二元一次方程式 int a1=3,b1=1,c1=5; int a2=1,b2=-2,c2=-3; int d=a1*b2-a2*b1; int x=(c1*b2-c2*b1)/d; int y=(a1*c2-a2*c1)/d; printf("x=%d\n",x); printf("y=%d", y); } </pre>	
15	<pre> int main(void) { //兩數交換 int a=3, b=4, t; t=a; a=b; b=t; printf("a=%d , b=%d", a, b); } </pre>	
16	<pre> int main() { //右旋轉 int a=2,b=-7,c=3;int t; t=a;a=b;b=c;c=t; printf("a=%d,b=%d,c=%d",a,b,c); } </pre>	
17	<pre> int main() { //除法與計數器 int a=8,b=3,q=0,r=0; a=a-b; q=q+1; //計數器 printf("a=%d,q=%d\n",a,q); a=a-b; q=q+1; //計數器 printf("a=%d,q=%d\n",a,q); r=a; printf("q=%d,r=%d\n",q,r); return 0; } </pre>	

```

18  int main() { // 輾轉相除法
        int a=21,b=9,r=0;
        r=a%b;
        printf("%d\n",r);
        a=b;
        b=r;
        r=a%b;
        printf("%d\n",r);
        printf("%d\n",b);
    }

```

6-3 專案除錯

程式設計的過程當中，會產生程式錯誤 (Bugs) 是很正常的現象，修正錯誤的過程，稱之為除錯 (Debug)。常見的程式錯誤有以下兩種，分別是語法錯誤 (Syntax Error) 與語意錯誤 (Semantic Error)，分別說明如下：

語法錯誤

每一種程式語言都有其撰寫的語法規則，初學者難免疏漏而卡關。所幸目前的編譯器功能也提升了，當錯誤發生時，編譯器會直接顯示相關錯誤的位置與訊息，方便我們進行除錯。以下是一些鍵入程式時常犯的錯誤，整理如下，方便初學者除錯。

1. 指令拚字打錯。CodeBlock 編譯器會將所有指令，也就是保留字以藍色字體呈現。例如：圖 6-1 的 `int`、`double` 都已經以藍色字體呈現，第 2 個 `doublz` 則未呈現藍色字體，此時就很方便除錯。

```

4  int main(){
5      int a,b,c; //係數
6      double d,x1,x2;
7      doublz d2;
8      a=2;b=-7;c=3;

```

圖 6-1 編譯器解析指令回饋的錯誤訊息

2. 變數重複宣告。例如：

```

int a,b,c;
double c;

```

3. 變數未遵守識別字命名規則。例如：
int 3a; (變數不能以數字開頭)
4. 忘記在敘述行尾端打上分號 (;)，例如：

```
printf("x1=%f\n",x1)
```

則錯誤訊息如圖 6-2：

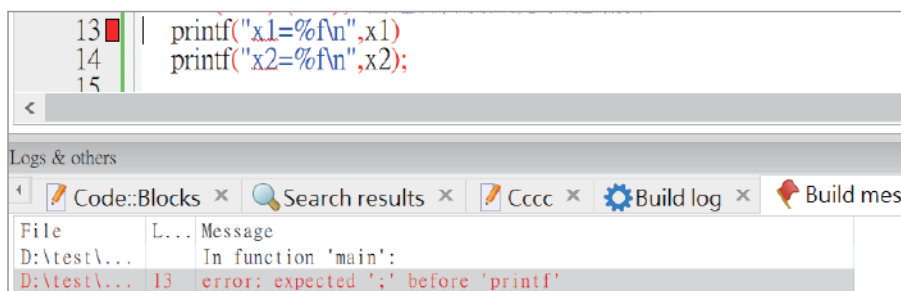


圖 6-2 編譯器解析指令回饋的錯誤訊息

且分號『;』是英數下的半形，不是中文狀態下的全形。

5. 載入標頭檔要使用「#」，且不需要「;」結束。例如：

```
#include <stdio.h>
```

6. 大小寫未區分。C 語言規定大小寫的字母是不同的字元。例如：

```
printf("Hello"); Printf("Hello"); //請留意P的大小寫
```

就不一樣了。

7. 不該出現的空白符號。例如，把

```
printf("Hello");
```

打成

```
print f("Hello");
```

也不行。

8. 拼錯字。例如將

```
printf("Hello");
```

打成

```
printf("Hello"); //請留意m與n或1（數字）與l（字母）
```

打字錯誤，導致指令錯誤，也會出現錯誤信息。

9. 括弧未配成對。有一個左大括弧『{』，務必要配一個右大括弧『}』；同樣地，左小括弧『(』務必要配一個右小括弧『)』；左中括弧『[』務必要配右中括弧『]』。初學者經常會忘記打上右括弧，導致語法錯誤。
10. 引號未成對。不論雙引號『"』（用於字串），或是單引號『'』（用於字元）的使用，都應該成雙成對出現。
11. 數學語言與程式語言。於前面範例中 $x_1 = \frac{-b+d}{2a}$ 是數學語言，程式語言如下：

```
x1=(-b+d)/(2*a);
```

請留意所有括號不能漏掉與乘號都不能省略。

☞ 語意錯誤 (Semantic Error)

語意錯誤又稱之為邏輯錯誤，一般發生在語法正確且程式可以執行的情況下，但執行結果卻不如預期。語意錯誤時，程式語法正確所以不會顯示任何的錯誤訊息，程式設計者必須重新思考程式的邏輯錯誤點，才能排除錯誤。例如，以下程式的結果不如預期，且沒有錯誤訊息。

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
int main () {
    int a,b,c;//係數
    double d,x1,x2;
    printf("input a,b,c :");
    scanf("%d,%d,%d",&a,&b,&c);//留意&
    d=sqrt(b*b-a*c);
    x1=(-b+d)/(2*a);//留意所有括號不能漏掉
    x2=(-b-d)/(2*a);//留意所有括號不能漏掉
    printf("x1=%f\n",x1);
    printf("x2=%f\n",x2);
    return 0;
}
```

此時可以將輸入值先指派，且將運算過程先輸出，程式如下：

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
int main () {
    int a,b,c; //係數
    double d,x1,x2;
    //將輸入先改為註解
    //printf("input a,b,c :");
    //scanf("%d,%d,%d",&a,&b,&c); //留意&
    a=2;b=-7;c=3; //指派輸入值
    d=sqrt(b*b-a*c);
    printf("d=%lf\n",d); //輸出運算過程
    x1=(-b+d)/(2*a); //留意所有括號不能漏掉
    x2=(-b-d)/(2*a); //留意所有括號不能漏掉
    printf("x1=%lf\n",x1);
    printf("x2=%lf\n",x2);
    return 0;
}
```

由以上 d 值的輸出，此時範圍縮小，可以發現 d 值就錯了，關鍵就在這一行，就可以很快發現問題。其次，初學者經常會忽略運算子的優先順序。例如：

```
x1=(-b+d)/(2*a)
```

若寫成

```
x1=(-b+d)/2*a
```

結果就不一樣，但程式不會出現錯誤訊息。

自我練習

1. 在 C 程式語言輸入 scanf() 函式的程式碼時，忘了輸入最後的「;」分號，請問會產生下列哪一種錯誤？
(A) 邏輯錯誤 (B) 數學運算錯誤 (C) 語法錯誤 (D) 執行期錯誤。

課後習題

題號	題目	結果
1	<pre> int main(){//APCS int a=2,b=3; int c=4,d=5,val; val=b/a+c/b+d/b; printf("%d\n",val); return 0; } </pre>	
2	<pre> int main(){//APCS int Total=23, Paid=1000, Change; Change = Paid - Total; printf ("%d\n", (Change-Change%500)/500); Change = Change % 500; printf ("%d\n", (Change-Change%100)/100); Change = Change % 100; printf ("%d\n", (Change-Change%50)/50); Change = Change % 50; printf ("%d\n", (Change-Change%10)/10); Change = Change % 10; printf ("%d\n", (Change-Change%5)/5); Change = Change % 5; printf ("%d\n", (Change-Change%1)/1); Change = Change % 1; } </pre>	
3	<pre> int main(){//APCS _Bool x=1,y=1,z=0,a; printf("%d\n",!(y z) x); printf("%d\n",!y (z !x)); printf("%d\n",z (x&&(y z))); printf("%d\n", (x x)&&z); return 0; } </pre>	

4	<pre> int main() { int a=1,b=2,c=3,d=4,e=5; cout<<a*b-c*d<<endl; cout<<a/b*c+d++e<<endl; a=(b=2)+(c=3); cout<<a<<endl; } </pre>	
5	<pre> int main() { int a=1,b=2,c=3; a+=b+=c+=5; cout<<"a="<<a<<endl; cout<<"b="<<b<<endl; cout<<"c="<<c<<endl; } </pre>	
6	<pre> int main() { int a=1,b=2,c=3; a=++b + ++c; cout<<a<<","<<b<<","<<c<<endl; } </pre>	
7	<pre> int main() { int a=1,b=2,c=3; a=b++ +c++; cout<<a<<","<<b<<","<<c<<endl; } </pre>	
8	<pre> int main() { int a=1,b=2,c=3; a=++b + c++; cout<<a<<","<<b<<","<<c<<endl; } </pre>	
9	<pre> int main() { int a=1,b=2,c=3; a=b-- + --c; cout<<a<<","<<b<<","<<c<<endl; } </pre>	
10	<pre> int main() { int a=1,b=2,c=3,d=4,e=5; c=(a+3)==b; cout <<c<<endl; } </pre>	

11	<pre>int main() { int a=1,b=2,c=3,d=4,e=5; c=(a+1)==b; cout <<c<<endl; }</pre>	
12	<pre>int main() { int a=5,b=4,c=3,d=2,e=1; cout<<((a!=b) & (b>=3))<<endl; }</pre>	
13	<pre>int main() { int a=5,b=4,c=3,d=2,e=1; cout<<((a!=b) ^ !(b>=3))<<endl; }</pre>	
14	<pre>int main() { int a=5,b=4,c=3,d=2,e=1; cout<<(!(a!=b) !(b>=3))<<endl; }</pre>	
15	<pre>int main() { int a=5,b=4,c=3,d=2,e=1; cout<<((a==1) & (a>0) && (b>=2))<<endl; }</pre>	
16	<pre>int main() { int a=5,b=4,c=3,d=2,e=1; cout<<((a==1) (a>0) & (b>2) ^ (c==3))<<endl; }</pre>	
17	<pre>int main() { int a=5,b=4,c=3,d=2,e=1; cout<<((a==1) (a>0) && (b>2) ^ (c==3))<<endl; }</pre>	
18	<pre>int main() { int a=3,b=-2,c=3,d=4; cout<<(a & b c ^ d)<<endl; }</pre>	

流程指令

C 語言依決策流程點的多寡，分為以下兩種決策流程指令，第一是雙向分歧決策流程 `if...else...`，例如肚子餓了就吃飯，否則繼續前進；第二是多向分歧決策流程的 `switch case`，例如你身上有 5000 元，走進一家五星級的大飯店用餐，你的分歧點就很多，有自助餐、中式套餐、日本料理、泰國餐點等等分歧點。

7-1 `if...else`

在日常生活領域中，當出現“假如～則～，否則～”時，此種決策流程模式有兩種解決問題的方案，故稱為雙向分歧決策流程，此時可使用 `if...else` 指令。`if...else` 指令的語法如下：

```
if (運算式)
{
    敘述區塊1；
}
[else
{
    敘述區塊2；
}]
```

以上語法說明如下：

1. 運算式的值若為 1(true)，則執行敘述區塊 1；運算式的值若為 0(false)，則執行敘述區塊 2，其流程圖如圖 7-1：

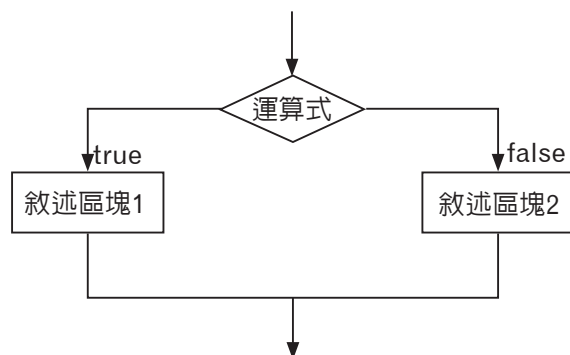


圖 7-1 if...else 流程圖

請鍵入以下程式，並觀察執行結果。

```

int main(){
    int a=66;
    char r='';
    if (a>=60){
        r='P';
    }
    else{
        r='F';
    }
    printf("%c",r) ;
}
  
```

2. if 或 else 後面敘述區塊若僅執行一個敘述，則 {} 可省略。例如：以上程式可以表示如下：

```

if (a>=60)
    r='P';
else
    r='F';
  
```

3. 程式設計語法的成對中括號的內容，表示可省略。例如，以下程式僅使用 if。

```

int main(){
    int a=66;
    char r='';
    if (a>=60){
        r='P';
    }
    printf("%c",r) ;
}
  
```

4. 有時爲了簡化程式的撰寫，可將否則的部分寫在 if 前面，並省略 else。例如，以下程式同義於上面程式。

```
int main(){
    r='F';
    if(a>=60)
        r='P';
}
```

5. if 或 else 後面預設僅執行一個敘述，若要執行兩個以上敘述，請加上 {}，且 {} 請要對稱排列。請鍵入以下程式，並寫出執行結果。

```
int main(){
    int a=33,b=0,c=0;
    if (a>=60)
        b=b+1;
        c=c+1;
        printf("%d\n",b) ;
    printf("%d\n",c) ;
    return 0;
}
```

```
int main(){
    int a=33,b=0,c=0;
    if (a>=60){
        b=b+1;
        c=c+1;
    }
    printf("%d\n",b) ;
    printf("%d\n",c) ;
    return 0;
}
```

6. 由以上兩段程式可以看出，左邊的程式碼，其縮排型式與程式執行流程截然不同，會造成後續維護的困擾。以下程式縮排型式的撰寫風格才與程式執行流程相符。

```
int main(){
    int a=33, b=0, c=0;
    if (a>=60)
        b=b+1;
    c=c+1;
    printf("%d\n", b);
    printf("%d\n", c);
    return 0;
}
```

7. if() 後面不用加分號 (;)，請鍵入以下程式，並觀察執行結果。

```
int main(){
    int a=33,b=0,c=0;
    if (a>=60)
        b=b+1;
        printf("%d\n",b) ;
    return 0;
}
```

```
int main(){
    int a=33,b=0,c=0;
    if (a>=60);
        b=b+1;
        printf("%d\n",b) ;
    return 0;
}
```

8. C 語言爲了強調語法的簡潔性，若決策流程的結果，只爲求得簡單的運算式，則可用以下敘述的三元運算子（?:），

```
r=(運算式1)?(運算式2):(運算式3);
```

其中『運算式 2』與『運算式 3』的括號只是增加程式的可讀性，此兩括號的有無並不影響程式的執行結果。當運算式 1 的值爲 1(true) 時，r= 運算式 2，當運算式 1 的值爲 0(false) 時，則 r= 運算式 3，例如，請鍵入以下程式，並寫出執行結果。

```
int main(){
    int a=66;
    char r;
    r=(a>=60)?('P'):( 'F');
    printf("%c\n",r) ;
    return 0;
}
```

9. 敘述區塊內可以放置任何合法敘述，當然也可以再放置 if；if 中有 if，稱爲巢狀 if，請看以下範例。

自我練習

題號	題目	結果
1	<pre>int main(){//技藝競賽程式組108 int a=5,b=2; if (a>b){ a=a*b+2; b++; }else{ a=a/2;b=b+4; } printf("%3d%3d\n",a,b); }</pre> (A) 8 2 (B) 12 3 (C) 2 6 (D) 4 6	
2	<pre>int main(){//技藝競賽程式組108 int x=30,y; if (x<=5){ y=x^2;x+=5; } else{</pre>	



	<pre> if (x<25) y=x+10; else y=x/10; x++; } printf("%3d%3d",y,x); } </pre> (A) 28 31 (B) 40 31 (C) 3 31 (D) 900 35	
3	<pre> int main(){//統測111 int y, a = 45; if(a >= 60) y = a + 1; else if(a >= 50) y = a + 2; else y = a + 3; printf("%d",y); } </pre>	
4	請問下面 C 程式中，若輸入值為 68，最後 salary 的值為何？ (技藝競賽程式組 111) <pre> int quantity, salary; scanf("%d",&quantity); if(quantity<=20) salary=15000+quantity*380; else if(quantity<=40) salary=15000+quantity*420; else salary=15000+quantity*420+(quantity-40)/10*1500; printf("%d\n",salary); </pre> (A) 29440 (B) 43560 (C) 46560 (D) 47760	
5	<pre> int main() { //分數評等第 int a=88; char r; if(a>=90) // 高於90分爲A r='A'; else if(a>=80) // 介於 80與90分爲B r='B'; else if(a>=70) //介於 70與80分爲C r='C'; else r='D'; //不符合上述情況則爲D printf("%c\n",r) ; return 0; } </pre>	

補充說明

1. 此題有人會寫成 `if (90<a<100)`，但 C/C++ 並沒有這種運算式，因為 `90<a<100` 是數學語言，不是程式語言。
2. 有人會這樣寫，這樣也可以，這樣可以減少一直縮排。

```
if(a>=90)           //高於90分爲A
    r='A'
else if(a>=80)       //介於 80與90分爲B
    r='B';
else if(a>=70)       // 介於 70與80分爲C
    r='C';
else
    r='D';           // 不符合上述情況則爲D
printf("%c\n",r) ;
```

3. 有人會寫成

```
if (a<100 && a>=90) r='A';
if (a<90  && a>=80)  r='B';
if (a<80  && b>=70)  r='C';
if (a<70  && b>=0)   r='D';
```

這樣雖然也可以，但是其執行效率非常差。因為不管分數為何，僅有一個敘述會成立，但卻都要進行四次判斷，如此會讓電腦不斷進行無效的判斷。而且，電腦的判斷時間遠大於執行計算的時間。

- 6 請問下列程式執行後，輸出結果為？ (技藝競賽程式組 110)

```
#include <stdio.h>
void main(){
    float salary = 400.0;
    if (salary > 400.0) {
        float bonus = 10.0;
        salary += bonus;
    }
    else {
        salary += salary * 0.2;
    }
    printf("%.2f", salary);
}
```

- (A) 410.000000 (B) 410.00 (C) 480.00 (D) 480.000000



7	<pre> int main() { //正負數判斷 int a=5; if (a>0) printf("plus"); else if (a==0) printf("Zero"); else printf("minus"); } </pre>	
8	<pre> int main() { //折扣問題 int a=100; int b=20; int c=0; if (b>=500) c=a*b*0.7; else if (b>=300) c=a*b*0.8; else if (b>=100) c=a*b*0.9; else c=a*b; printf("%d",c); } </pre>	
9	<pre> int main() { //判斷點是否在直線 int a=2,b=-3,c=5; int x=-1,y=1; char ans='n'; if (a*x+b*y+c==0) ans='y'; printf("%c",ans); } </pre>	
10	<pre> int main() { //判斷是否母音 char b='e'; printf("%c\n",b); if (b=='a' b=='e' b=='i' b=='i' b=='u') printf("母音"); else printf("子音"); } </pre>	

11	<pre> int main() { //分數轉燈號 int a=10; if (a>=21) b=5; else if (a>=16) b=4; else if (a>=11) b=3; else if (a>=6) b=2; else if (a>=1) b=1; else b=0; printf("a=%d,b=%d",a,b); } </pre>	
12	<pre> int main() { //象限轉換 int x=3,y=4; char r=' '; if(x>0) if(y>0) r='1'; /* 第一象限 */ else r='4'; /* 第四象限 */ else if(y>0) r='2'; /* 第二象限 */ else r='3'; /* 第三象限 */ printf("x =%d,y=%d : 第 %c 象限",x,y,r); } </pre> 補充說明 1. 此題目有人會寫成 <pre> if (x>0 && y>0) r='1'; if (x<0 && y>0) r='2'; if (x<0 && y<0) r='3'; if (x>0 && y<0) r='4'; </pre> 這樣雖然沒有錯，但是執行效率非常差，因為電腦要不斷的比較。	



13	以下程式要求輸入密碼，當密碼是 1234 時，才執行程式，請問底線要填入什麼？ <pre> int main() {// printf("Please input password:"); int a; scanf("%d",&a); _____ printf("Hello"); } } </pre>	
14	以下程式可以讓使用者輸入小寫字元，電腦評判是否正確，請填入底線程式。 <pre> int main(){ srand(time(NULL)); int _____ cout<<a<<endl;//輸出a cout<<(char)a; char b; cin>>b;//輸入b if _____//資料型態相同才能比較 cout<<"right" //輸出 else cout<<"wrong"; return 0; } </pre>	
15	以下程式可練習心算，請填入底線程式。 <pre> int main(){ srand(time(NULL));//以時間當作亂數種子 int a=rand()%10; int b=rand()%10; int c; printf("%d+%d=",a,b); scanf("%d",&c); _____ printf("right"); else printf("wrong"); } </pre>	

7-2 switch...case

一個決策流程點若同時擁有三個或三個以上的解決方案，則稱此為多向分歧決策流程。多向分歧決策流程雖也可使用 7-1 的巢狀 if else 解決，但卻增加程式的複雜度及降低程式可讀性。若此一決策流程點能找到適當的運算式，能使問題同時找到分歧點，則可使用 switch case 敘述。switch case 語法如下：

```
switch (運算式)
{
    case 常數1:
        敘述區塊1;
        break;
    case 常數2:
        敘述區塊2;
        break;
    case 常數3:
        敘述區塊3;
        break;

    [default:
        敘述區塊n;]
}
```

以上語法說明如下：

1. switch 的運算式值僅能為『整數』或『字元』。
2. case 的常數僅能『整數』或『字元』，且其資料型態應與上面的 switch 運算式相同。
3. 電腦將會依 switch 的運算式值，逐一至常數 1、常數 2 尋找合乎條件的 case，並執行相對應的敘述區塊，直到遇到 break 敘述，才能離開 switch。
4. default 可放置特殊情況，也就是沒有適當的 case，則執行 default。若省略 default，且若沒有任何 case 滿足 switch 運算式，則程式會默默離開 switch 敘述。
5. 語法中，成對中括號裡面的敘述，表示可省略。
6. 敘述區塊可放置任何合法的敘述，當然也可放置 switch 或 if。
7. 以下敘述，可將 1、2、3、4 轉為字元 'A'、'B'、'C'、'D'。

```
int main() {  
    int a=1;  
    char r;  
    switch(a) {  
        case 1:  
            r='A';  
            break;  
        case 2:  
            r='B';  
            break;  
        case 3:  
            r='C';  
            break;  
        case 4:  
            r='D';  
            break;  
        default :  
            r=' ';  
    }  
    printf("%c\n",r) ;  
    return 0;  
}
```

8. 有些語言可用逗號將兩種 case 放在一起，但在 C/C++ 語言中每一 case 僅能放置一個常數，所以若兩個或兩個以上 case，有相同的處理方法，則應將兩個 case 分成兩個敘述，且共用 1 個 break，如以下程式。

```
case 10:  
case 9:  
    r='A';  
    break;
```

請看以下範例。

題號	題目	結果
1	<pre> int main(){//技藝競賽程式組108 int n=4,x=7,y=8; switch(n){ case 1:x=n;break; case 2:y=y+4;//缺break case 3:x=n+5;break; case 4:x=x*4;//缺break default:y=y-4; } printf("%2d%2d",x,y); } </pre> (A) 28 4 (B) 4 8 (C) 9 12 (D) 9 8	
2	電腦發出 Do Re Mi, 使用者輸入 1 2 3, 電腦評判是否正確 <pre> int main(){// srand(time(NULL));//以時間當作亂數種子 int a=1+rand()%7; int b=0; int c; switch (a){ case 1: b=523; break; case 2: b=587; break; case 3: b=659; break; } _beep(b,500); scanf("%d",&c) ; printf("Right"); else printf("Wrong"); } </pre>	



3	<pre> int main() { int a=1;char r; switch(a) { case 1: r='A'; break; case 2: r='B'; break; case 3: r='C'; break; default : r=' '; } printf("%c\n",r) ; } </pre>	
4	<pre> int main() { int a=1;char r; switch(a) { case 1: r='A'; case 2: r='B'; case 3: r='C'; default : r=' '; } printf("%c\n",r) ; } </pre>	
5	以下程式可評判等第：90 ~ 100 為 A，80 ~ 89 為 B，底線部分請填入程式。 <pre> int main() { int a=99; char r=' '; switch(a) { _____ r='A'; break; _____ r='B'; break; } printf("the grade is %c",r); } </pre>	

6	<pre> int main() { int 22; int b=0; switch(a){ case 21 ...25: b=5; case 16 ...20: b=4; break; } printf("a=%d,b=%d",a,b); } </pre>	
7	以下程式可當作電子琴，輸入後不用按 Enter，請填入底線程式 <pre> #include <conio.h>//本例使用getche int main() { char a; while(1){//無限迴圈，請看下一章 _____//輸入後，不用按Enter switch(a) { case('1'): _beep(523,500);//Do break; case ('2'): _beep(587,500);//Le break; case ('3'): _beep(659,500);//Mi break; } } } </pre>	

課後習題

題號	題目	結果
1	<pre>void main () { int count = 10; if (count > 0) { count = 11; } if (count > 10) { count = 12; if (count % 3 == 4) { count = 1; } else { count = 0; } } else if (count > 11) { count = 13; } else { count = 14; } if (count) { count = 15; } else { count = 16; } printf ("%d\n", count); }</pre>	
2	<pre>void main () { int x=20;char y; if (x==10) y = 'a'; if (x==20 x==30) y = 'b'; y = 'c'; printf ("%c",y); }</pre>	

3	<pre> void main () { int x=15;char y; if (x==10) y = 'a'; else if (x==20 x==30) y = 'b'; else y = 'c'; printf("%c",y); } </pre>	
4	<pre> void main () { int x=20;char y; if (x==10) y = 'a'; if (x>=20 && x<=30) y = 'b'; y = 'c'; printf("%c",y); } </pre>	
5	<pre> void main () { int x=20;char y; if (x==10) y = 'a'; else if(x>=20 && x<=30) y = 'b'; else y = 'c'; printf("%c",y); } </pre>	
6	<pre> int main() { int a=1; char r='0'; switch(a) { case '1': r='A'; break; default : r='X'; } printf("%c\n",r) ; return 0; } </pre>	



7	<pre>int main() { int a=1;char r='0'; switch(a) { case 1: r='A'; case 2: r='B'; break; } printf("%c\n",r) ; return 0; }</pre>	
8	<pre>int main() { int a=5; char r='0'; switch(a) { case 1: r='A'; break; case 2: r='B'; break; default : r='X '; } printf("%c\n",r) ; return 0; }</pre>	
9	<pre>int main() { int a=1; char r='0'; switch(a) { case 1: r='A '; break; case 2: r='B '; break; default : r=' X '; } printf("%c\n",r) ; return 0; }</pre>	

10	<pre> void main () { //APCS int x=20;char y; switch (x) { case 10: y = 'a'; break; case 20: case 30: y = 'b'; break; default: y = 'c'; } printf("%c",y); } </pre>	
11	<pre> void main () { //APCS for (int s=100 ;s>=0;s--){ printf("s=%d: ",s); if (s>=90) { printf ("A \n"); } else if (s>=80) { printf ("B \n"); } else if (s>60) { printf ("D \n"); } else if (s>70) { printf ("C \n"); } else { printf ("F\n"); } } printf("\n"); } </pre>	
12	<pre> int main() { int a=5,b=4,t; if (a>b) t=a;a=b;b=t; printf("%d",a); return 0; } </pre>	

13	<pre>//請留意else 是預設最接近的if int main(int argc, char** argv) { //The else default to the nearest int a=-5,b=4,t=0; if (a>0) if (b>0) t++; else t--; printf("%d\n",t); a=-5;b=4;t=0; if (a>0) if (b>0) t++; else t--; printf("%d\n",t); //If you want to change it,you need to use the curly brackets. a=-5;b=4;t=0; if (a>0){ if (b>0) t++; } else t--; printf("%d\n",t); return 0; }</pre>	
14	<pre>int main() { int a=1,b=2,c=2,d=2,e=2; if (a=='1') b=3; printf("%d",b); if (a=1) c=3; printf("%d",c); if (a='1') d=3; printf("%d",d); }</pre>	

15	<pre> int main() { //求極小值 int a=33,b=23,c=77,d=66; int m=a; if (b<m) m=b; if (c<m) m=c; if (d<m) min =d; printf("max:%d",m } </pre>	
16	<pre> int main() { //時間區間判斷 int h1=5,m1=10,h2=8,m2=50,h3=3,m3=20; int h,m,t1,t2,t3,t; t1=h1*60+m1;t2=h2*60+m2;t3=h3*60+m3; if (t3>=t2) printf("A"); else if(t3>t1) printf("B"); else printf("C"); } </pre>	
17	<pre> int main() { //複選題評分 int a=25; //標準答案ABE b=17; // AE答對4個，錯1個3分 int c=0; int d=0; //答對幾個 int e=0; //得分 if (b==0) e=0; else { c=a^b; //逐位元xor，位元相同是0，不同是1 d=0; printf("%d\n",c); c=~c; //取補數，1變0，0變1 printf("%d\n",c); d=d+(c & 1); //計算1的個數 //d=d+c & 1 // 錯，因為+的優先順序大於& c=c>>1; //右移 d=d+(c & 1); //位元邏輯and c=c>>1; d=d+(c & 1); } } </pre>	

	<pre> c=c>>1; d=d+(c & 1); c=c>>1; d=d+(c & 1); } if (d==5) e=5; else if (d==4) e=3; else if (d==3) e=1; else e=0; printf("%d\n",d); printf("%d\n",e); } </pre>	
18	<pre> int main() { //去头尾，求平均 int a=3,b=4,c=5,d=6,e=7; int max=a,min=a; if (b>max) max=b; if (c>max) max=c; if (d>max) max=d; if (e>max) max=e; if (b<min) min=b; if (c<min) min=c; if (d<min) min=c; if (e<min) min=e; int avg=(a+b+c+d+e-max-min)/3; printf("%d",avg); return 0; } </pre>	

19	<pre>int main() { int a=4,b=7,c=1,t; if(a>b) { t=a; a=b; b=t; }//兩數交換 if(b>c) { t=b; b=c; c=t; } if(a>b) { t=a; a=b; b=t; } printf("%d,%d,%d ",a,b,c); }</pre>	
----	---	--

迴圈指令

生活上常會遇到重複執行的事件，例如，一天要上七堂課，連續作答選擇題 10 題等等。所以電腦就有所謂的迴圈，幫您完成指定的重複次數。其次，所有的程式語言都有兩種基本迴圈，那就是 for 與 while，其用法與差別請看本章說明。

8-1 for 迴圈

我們小時候常常被老師罰寫某個句子十遍，這件事若讓電腦做，那可就輕鬆了，因為電腦有 for 迴圈運算。例如，以下敘述可將『洪峻維』重複寫 10 次。

```
for (int i=1;i<=10;i++)
    printf("洪峻維\n");
```

for 指令的語法如下：

```
for([計數變數=起始值] ; [迴圈運算式] ; [計數變數的改變量]) {
    [指令區塊1;]
    [break;]
    [continue;]
    [goto (標籤名稱);]
    [指令區塊2;]
}
```

以上語法說明如下：

1. 只要“迴圈運算式”結果為 1(真)，則繼續執行迴圈內的指令區塊。
2. for 運算僅預設執行 1 個敘述，若要執行兩個以上敘述，請加上 {}。請鍵入以下程式，寫出執行結果。

8-2 程式設計實習進階

```
int main() {  
    for (int i=1;i<=10;i++){  
        printf("%d\n",i);  
        printf("%d\n",i);  
    }  
}
```

```
int main() {  
    int i;  
    for ( i=1;i<=10;i++){  
        printf("%d\n",i);  
        printf("%d\n",i);  
    }  
}
```

以上變數*i*的有效範圍僅在迴圈內，所以以下程式，第 2 個 `printf("%d\n",i);` 的 *i* 無效。

```
int main() {  
    for ( int i=1;i<=10;i++){  
        printf("%d\n",i);  
    }  
    printf("%d\n",i);  
}
```

3. 計數變數的改變量若為遞增，請留意終值較大；若為遞減，則終值較小。請鍵入以下程式，並寫出執行結果。

```
int main() {  
    for (int i=1;i<=10;i++){  
        printf("%d\n",i);  
    }  
    for (int i=1;i<=10;i=i+4)  
        printf("%d\n",i);  
    return 0;  
}
```

```
int main() {  
    for (int i=10;i>=1;i--)  
        printf("%d\n",i);  
    for (int i=10;i>=1;i=i-3)  
        printf("%d\n",i);  
    return 0;  
}
```

4. 前面計數變數都是由小而大，使用『<=』。若是由大到小，則要用『>=』，這樣第一筆資料才會得到『true』，才有輸出結果。

```
int main() {  
    for( i=10;i>=1;i--) {  
        printf("%d",i);//10987654321  
    }  
}
```

5. 以下程式每次遞減 3。

```
int main() {  
    for( i=10;i>=1;i-=3) {
```

```
        printf("%d",i);//10741
    }
}
```

6. 關係運算子通常使用『<,<=,>,>=』，若使用『==,!=』，則沒有錯誤訊息，但執行結果不同。請鍵入以下程式，寫出執行結果。

```
int main(){
    for (int i=1;i==5;i++)
        printf("%d",i);
    printf("\n");
    return 0;
}
```

```
int main(){
    for (int i=1;i!=5;i++)
        printf("%d",i);
    printf("\n");
    return 0;
}
```

7. 迴圈運算式也不能寫成：

```
int main() {
    for ( int i=1;i=10;i++)
        printf("%d",i);
}
```

因為 C 關係運算子沒有「=」，這樣是語法錯誤。其次，若寫成：

```
int main() {
    for ( i=1;i==10;i++)
        printf("%d",i);
}
```

則是語法沒錯，但結果不一樣，因為一開始 `1==10` 這件事就 `false`，所以什麼都沒有作就離開迴圈。

8. 只要迴圈運算式為 0(false)，就離開迴圈。請鍵入以下程式，並寫出執行結果。

```
int main(){
    for (int i=1;i>=5;i++)
        printf("%d\n",i);
    for (int i=5;i<=1;i++)
        printf("%d\n",i);
    return 0;
}
```

9. 計數變量也可在迴圈內改變。例如：

```
int main(){
    for (int i=1;i<=10 ;i++){
        printf("%d\n",i);
        i++;
    }
    return 0;
}
```

10. 程式若執行到 break，則會提早離開 for 迴圈。(請鍵入以下程式，寫出執行結果)

```
int main(){
    int i, sum=0;
    for(i=1; i<=6; i++) {
        printf("%d\n",i);
        if(i==4)
            break; //強制離開迴圈
        printf("%d\n",i);
    }
    printf("when out of loop, i=%d",i);
    return 0;
}
```

11. 程式若執行到 continue，則會略過 continue 下面的指令區塊 2，繼續執行下一個計數變量。(請鍵入以下程式，寫出執行結果)

```
int main(){
    int i, sum=0;
    for(i=1; i<=6; i++) {
        printf("%d\n",i);
        if(i==4)
            continue; //略過下一指令，繼續執行下一個i
        printf("%d\n",i);
    }
    printf("when out of loop, i=%d",i);
    return 0;
}
```

12. 程式若執行到 goto，則會略過 goto 下面的指令區塊 2，跳躍至（標籤名稱）所在的位置，繼續執行標籤以下的指令區塊。

```
int main(){
    int i, sum=0;
    for(i=1; i<=6; i++) {
        printf("%d\n",i);
        if(i==4)
            goto aa; //強制離開迴圈
        printf("%d\n",i);
    }
    aa:
    printf("when out of loop, i=%d",i);
    return 0;
}
```

13. 指令區塊內可以放置任何合法的指令，當然也可含 for。for 內有 for，稱為巢狀迴圈（Nested Loop）。例如，以下指令可印出 1 至 10 五次。

```
int main(){
    for (int i=1;i<=5;i++){
        for (int j=1;j<=10;j++)
            printf("%d",j);
        printf("\n");
    }
    return 0;
}
```

14. 運算式只要合理就好，不一定要用計數變數。例如：

```
int main(){
    int i=0,s=0;
    for (i=1;s<=10;i++){
        printf("%d",i);
        s=s+i;
    }
    printf("%d",i);
    printf("%d",s);
    return 0;
}
```

```
int main(){
    int i=0,s=0;
    for (i=1;i<=10 && s<=10;i++)
    {
        printf("%d",i);
        s=s+i;
    }
    printf("%d",i);
    printf("%d",s);
    return 0;
}
```

自我練習

題號	題目	結果
1	<pre> int def(int x){ //技藝競賽程式組107 if(x==1) return 1; else{ int i,B=0,A=1,O,S=0;//字元O for (i=2;i<=x;i++){ S=A+B;O=S; B=A;A=S; } return O;//字元O } } int main(){ int z=def(7); printf("%i",z); } </pre> (A) 5 (B) 8 (C) 13 (D) 21	
2	<pre> int main(){//技藝競賽程式組107 int y1,y2=13,y3=1; for (y1=0;y1<=y2;y3){ y3+=y1; y1+=2; } printf("%d",y3); } </pre> (A) 31 (B) 43 (C) 57 (D) 73	
3	<pre> int main(){//技藝競賽程式組108 int x=0,y=0; for (y=1;y<=20;y++){ int z=y%5; if(z==0) x++; } printf("%3d%3d\n",x,y); } </pre> (A) 0 0 (B) 4 21 (C) 2 11 (D) 3 11	

4	請問下列程式片段執行後，會印出甚麼？（技藝競賽程式組 109） (A): 5 31 (B): 15 31 (C): 10 31 (D): 12 31 <pre> main() { int x = 2, y = 0; for (y = 1; y <= 30; y++) { int z = y % 6; if (z == 0) x+=2; } printf("%3d%3d", x, y); } </pre>	
5	請問下列程式片段執行後，輸出的第 12 個數值是？ (技藝競賽程式組 109) (A): 31 (B): 37 (C): 41 (D): 43 <pre> main() { int p, d; bool flag; for (p = 2; p <= 50; ++p) { flag = 1; for (d = 2; d < p; ++d) if (p%d == 0) flag = 0; if (flag != 0) printf("%i ", p); } } </pre>	
6	//字元輸入練習程式，電腦產生字元，使用者輸入此字元，電腦評判正確字元個數 <pre> int main() { srand(time(NULL)); int r=0,w=0; for (int i=1;i<=10;i++){ int a=97+rand()%26; printf("%c", (char)a); char b; if (b==(char)a){ printf("right\n"); r++; } else{ printf("wrong\n"); w++; } } printf("right=%d\n",r); printf("wrong=%d\n",w); } </pre>	

8-8 程式設計實習進階

7	程式執行如下圖，請填入底線程式。 <pre>int main() { int a=____; for(int i=2; i<=a; i++) printf("%d * %d= %d\n",____); return 0; }</pre> 	
8	<pre>int main() { //判斷是否質數 int a=11;_Bool prime=1; //布林型態 for(int i=2; i<=a-1; i++) { if (a%i==0) { // 留意==, 不是= prime=0; break; //找到1個就不是了 } } if(prime) printf("A",a) ; //1 else printf("B",a); //0 }</pre>	
9	<pre>int main() { //判斷是否互質 int a=10,b=198,t;_Bool prime=1; //布林型態 if (b>a) {t=a;a=b;b=t;} for(int i=2; i<=a; i++) { if (a%i==0 && b%i==0) { prime=0; break; //找到1個就不是了 } } if(prime) printf("%d %d is A",a,b) ; else printf("%d %d is B",a,b); }</pre>	

8-2 while 迴圈

上一節的 for 是用於程式設計階段已知迴圈次數。但有些情況，我們於程式設計階段並不知迴圈的執行次數，此時即可使用 while 指令。且有些迴圈可能一次都不執行，所以 while 指令又分為前測試迴圈（Pre-test loop）與後測試迴圈（Post-test loop）。while 的前測試迴圈語法如下圖左，後測試迴圈如下圖右。

<pre>while(運算式){ 指令區塊; break; continue; }</pre>	<pre>do{ 指令區塊; break; continue; }while (運算式);</pre>
---	---

以上語法說明如下：

1. 不論是前測試迴圈或後測試迴圈，均是運算式值為 1(真)時，繼續執行迴圈，運算式為 0(偽)時，離開迴圈。
2. 前測試與後測試迴圈的差別為，前測試迴圈有可能一次均不執行迴圈，但後測試迴圈至少執行一次。
3. 後測試迴圈的 while (運算式) 後面要加分號(;)，而前測試迴圈的 while 不用加分號。
4. while 指令區塊內亦適用 break 與 continue，前者為強迫提早離開迴圈，後者可略過部份指令，提早進入條件運算式。
5. 以下程式片段可用前測試迴圈解輸出 1 ~ 10。

```
int main(){
    int i=1;
    while ( i<=10){
        printf("%d\n",i);
        i=i+1;
    }
    return 0;
}
```

以下程式片段可用後測試迴圈輸出 1 ~ 10。

```
int main(){
    int i=1;
    do{
```

```

        printf("%d\n",i);
        i=i+1;
    } while (i<=10);
    return 0;
}

```

自我練習

題號	題目	結果
1	<pre> int abc(int x,int y){ //技藝競賽程式組107 int tmp; while(x%y!=0){ tmp=y;y=x%y;x=tmp; } return y; } int main(){ int z=abc(159,52); printf("%i",z); } </pre> (A) 1 (B) 2 (C) 13 (D) 26	
2	<pre> int main(){//技藝競賽程式組107 int k=2,n=0; while(k<65535){ k=k*k; n++; } printf("%d",n); } </pre> (A) 4 (B) 5 (C) 6 (D) 7	
3	<pre> int f(){//技藝競賽程式組107 int p=2; while(p<2000){ p=p*2; } return(p); } int main(){ int p=f(); printf("%d",p); } </pre> (A) 1023 (B) 1024 (C) 2047 (D) 2048	

4	<pre>int main() { //技藝競賽程式組107 int n=0; int i=1; while(i<=100) { n=n+i; i=i+2; } printf("%d",n); }</pre> (A) 2000 (B) 2500 (C) 5000 (D) 10000	
5	<pre>int main() { //技藝競賽程式組108 int x=110,y=20; while(x>120) { y=x-y; x++; } printf("%3d%3d\n",x,y); }</pre> (A) 111 90 (B) 112 21 (C) 110 90 (D) 110 20	
6	<pre>int main() { //技藝競賽程式組108 int x=3,y=6,z=0; do{ printf("Please input z:"); scanf("%d",&z); x=x+z+y; y++; }while(z<10); y*=2; printf("%3d%3d%3d",z,x,y); }</pre> (A) 10 49 20 (B) 10 48 20 (C) 10 45 18 (D) 10 3 6	
7	曉華想要知道三角函數 $\sin(x)$ 在 $x = 0$ 之後遞增的變化情形，寫了如下的 C 語言程式碼，卻發現迴圈內行號 8 和行號 9 的程式碼只執行了一次，下列哪一種修改程式的方式可以讓迴圈內的程式碼多執行幾次？ (提示：$\sin(1)=0.8415$，統測 112) <pre>1 #include <stdio.h> 2 #include <math.h> 3 int x = 100; 4 int main() {</pre>	

	<pre> 5 int x = 0; 6 double y = 0.0; 7 do{ 8 y = 10*sin(x); 9 printf("x=%d, y=%lf\n", x, y); 10 } while(++x <= y); 11 printf("end of program\n"); 12 return 0; 13 } </pre> (A) 把行號 3 中的 $x = 100$ 改為 $x = 0$ (C) 把行號 6 中 y 的初始值改為 -1.0 解：請複習 6-1 節，前置與後置遞增，並鍵入以下程式。 <pre> #include <stdio.h> #include <math.h> int x = 100; int main(){ int x = 0; double y = 0.0; int a,b; do{ y = 10*sin(x); printf("x=%d, y=%lf\n", x, y); //a=++x;//x=x+1;b=x b=x++;//b=x;x=x+1 printf("x=%d, b=%d\n", x, a); } while(b<= y); printf("end of program\n"); return 0; } </pre>	
8	請問下列 C 語言指令何者不為迴圈指令？（技藝競賽修護組 112） (A) switch/case (B) while (C) for (D) do/while。	
9	有關於迴圈指令，下列敘述何者正確？（技藝競賽修護組 110） (A)「for」：執行程式的重複次數不確定時適用之 (B)「while」：先執行後判斷重複程式時適用之 (C)「do while」：先判斷後執行重複程式時適用之 (D)「while」：執行程式的重複次數不確定時適用之。	

10	請問下列 C 程式碼片段會輸出多少個 good 字串？ (技藝競賽修護組 111) <pre>int a = 1; while (a <= 5) printf("good");</pre> (A) 5 (B) 4 (C) 0 (D) 無限多。	
11	請問下列 C 程式碼片段 do/while 迴圈一共會執行幾次？ (技藝競賽修護組 111) <pre>int n=0; do { n += 3; } while(n < 10);</pre> (A) 3 (B) 4 (C) 5 (D) 6。	
12	如果執行下列 C 語言程式，則此重複執行的程式區塊會執行多少次？ (技藝競賽修護組 110) (A) 無窮次 (B) 2 次 (C) 3 次 (D) 4 次。 <pre>01 int a=0; 02 while(a<3){ 03 ; 04 }</pre>	
13	執行下列 C 程式片段，請問最後輸出是？ (技藝競賽電腦修護 106) <pre>#include <stdio.h> void main() { int number = 1061130, result; do { result = number % 10; printf("%i", result); number = number / 10; } while (number != 0); }</pre> (A) 1061130 (B) 0311601 (C) 106113 (D) 311601	
14	下列程式執行後，輸出是？ (技藝競賽程式設計 112) <pre>#include <stdio.h> int main(void) { int x = 1, s = 0; while (x <= 5) { int y = x * x * x; s += y; ++x; } printf("%d\n", s); }</pre> (A) 100 (B) 125 (C) 225 (D) 216	

15	程式片段中，請問執行結果為何？（技藝競賽程式設計 109） (A): 1 2 3 4 (B): 1 2 3 4 5 (C): 2 3 4 (D): 2 3 4 5	
<pre>1 #include <stdio.h> 2 3 int main(){ 4 int a = 1; 5 do{ 6 printf("%d ", a); 7 }while(++a<5); 8 9 return 0; 10 }</pre>		
16	程式片段中，請問執行結果為何？（技藝競賽程式設計 109） (A): 1 2 3 4 (B): 1 2 3 4 5 (C): 2 3 4 (D): 2 3 4 5	
<pre>1 #include <stdio.h> 2 3 int main(){ 4 int a = 1; 5 do{ 6 printf("%d ", a); 7 }while(a++<5); 8 9 return 0; 10 }</pre>		
17	程式片段中，請問執行結果為何？（技藝競賽程式設計 109） (A): 1 2 3 4 (B): 1 2 3 4 5 (C): 2 3 4 (D): 2 3 4 5	
<pre>1 #include <stdio.h> 2 3 int main(){ 4 int a = 1; 5 while(++a<5){ 6 printf("%d ", a); 7 } 8 9 return 0; 10 }</pre>		
18	程式片段中，請問執行結果為何？（技藝競賽程式設計 109） (A): 1 2 3 4 (B): 1 2 3 4 5 (C): 2 3 4 (D): 2 3 4 5	
<pre>1 #include <stdio.h> 2 3 int main(){ 4 int a = 1; 5 while(a++<5){ 6 printf("%d ", a); 7 } 8 9 return 0; 10 }</pre>		

- 19 程式片段中，是利用輾轉相除法來求得 m 與 n 的最大公因數，請問迴圈內的敘述應該為何？（技藝競賽程式設計 109）

```

1  #include <stdio.h>
2
3  int main(){
4      int m, n, r;
5
6      while(scanf("%d%d", &m, &n)!=EOF){
7          r = m % n;
8          while(r!=0){
9
10
11
12
13          }
14          printf("%d\n", n);
15      }
16      return 0;

```

- (A) $r=m\%m; m=n; n=r;$ (B) $r=m\%n; n=r; m=n$
 (C) $n=r; m=n; r=m\%n;$ (D) $m=n; n=r; r=m\%n;$

- 20 程式片段中 8~10 行，while 迴圈該如何撰寫，可以計算輸入的整數 n 每個位數的總和，例如輸入 1234，輸出 10。（技藝競賽程式設計 109）

```

1  #include <stdio.h>
2
3  int main(){
4      int n;
5      int sum = 0;
6      scanf("%d", &n);
7
8      while( ){
9
10
11
12      }
13      printf("%d\n", sum);
14      return 0;
15  }

```

- (A) $\text{while}(n!=0)\{$
 $n /= 10;$
 $\text{sum} += n\%10;$
 $\}$
- (B) $\text{while}(n/10!=0)\{$
 $\text{sum} += n\%10;$
 $n /= 10;$
 $\}$
- (C) $\text{while}(n!=0)\{$
 $\text{sum} = n\%10;$
 $n /= 10;$
 $\}$
- (D) $\text{while}(n!=0)\{$
 $\text{sum} += n\%10;$
 $n /= 10;$
 $\}$

21	<pre> int main() { //商與餘數 int a=8,b=3,q=0; while(a>=b) { a=a-b; q++; } printf("%d\n", q); printf(" %d\n", a); } </pre>	
22	請問下列 C 語言片段程式最後輸出結果為何？ (A) 99 (B) 100 (C) 101 (D) 102。 <pre> int i = 0; do ++i; while (i <= 100); printf("%d", i); </pre>	
23	<pre> int main() { //技藝競賽程式組108 int n=1234,sum=0; while(n!=0) { sum=sum+n%10; n=n/10; } printf("%d",sum); } </pre> (A) 1234 (B) 10 (C) 1 (D) 4	
24	<pre> int main() { //實數商 float a=8,b=3,q=0; a=a*10; while(a>=b) { a=a-b; q++; } q=q/10; a=a/10; printf(" %f\n", q); printf(" %f\n", a); } </pre>	

25	<pre> int main() { int i=3,sum=0,n=0; while(sum<100){ printf("%d\n",i); sum=sum+i; n=n+1; i=i+3; } printf("%d\n",i); printf("%d\n",n); return 0; } </pre>	
26	<pre> int main() { int i=3,sum=0,n=0; for (i=3;sum<100;i=i+3){ printf("%d\n",i); sum=sum+i; n=n+1;//次數 } printf("%d\n",i); printf("%d\n",n); } </pre>	
27	<pre> int main() { //統計加3的次數 int i=3,sum=0,n=0; while(sum<100){ sum=sum+i; n=n+1;//次數 } printf("%d\n",n); } </pre>	
28	以下程式可連續輸入數字，輸入 -1 時結束，請填寫程式空白。 <pre> int main() { int sum=0; int a; scanf("%d",&a); _____{ sum=sum+a; scanf("%d",&a); } printf("%d \n",sum); } </pre>	

29	<pre> int main(){//統計數字和 int i=0,a=251,s=0;int b; while (a>0){ b=a%10; s=s+b; i++; a=a/10; } printf("%d %d ",s,i); } </pre>	
30	<pre> int main(){ int i=0,a=25132,s=0;int b; while (a>0){ b=a%100; s=s+b; i++; a=a/100; } printf("%d %d ",s,i); } </pre>	
31	<pre> int main(){//數字反向輸出 int a=25132,b; while (a>0){ b=a%10; printf("%d",b); a=a/10; } } </pre>	
32	以下程式可連續出現兩個亂數，讓使用者練習心算，直到錯了才停止，電腦輸出答對題數 <pre> int main(){ srand(time(NULL)); int r=0;_Bool right=1; while (____){ int a=1+rand()%9; int b=1+rand()%9; printf("%d +%d=",a,b); int c; scanf(" %d",&c);// ____{ printf("right\n"); r++; } else{ printf("wrong\n"); right=0; } } printf("right=%d\n",r); } </pre>	

33	以下程式，要求輸入密碼，密碼可以錯 2 次，第 3 次就結束，填入空白程式。 <pre> int main(){ int a; int n=0; _____{ printf("Please input password:"); scanf("%d",&a); if (a==1234){ printf("right"); n=0; } else{ n++; printf("password worng %d time\n",n); } } } </pre>	
34	<pre> int main(){//判斷數字是否全偶數 int a,b; int inc; a=22664; int a1=a; inc=1;//true while (a>0){ b=a%10; if (b%2==1){ inc=0; break; } a=a/10; } if(inc){ printf("%d is A\n",a1); }else{ printf("%d is B\n",a1); } return 0; } </pre>	

35	<pre> int main(){//判斷數字是否遞增 int a,b1,b2; int inc; a=12634; int a1=a; b1=a%10; a=a/10; inc=1;//true do{ b2=a%10; a=a/10; if (b2>=b1){ inc=0; break; } b1=b2; }while (a>0); if(inc){ printf("%d is A\n",a1); }else{ printf("%d is B\n",a1); } } </pre>	
36	程式填空。程式，可以連續輸入阿拉伯數字，每次 1 個，當輸入「e」時結束，且組合成一個數字。例如：輸入 4,3,8,e 則輸出 438。 <pre> int main() { char a; int s=0; scanf(" %c",&a); while (_____) { int b=a-48; printf("%d\n",b); _____; scanf(" %c",&a); } printf("%d",s); } </pre>	
37	有關下面 C 程式片段中，若輸入值為 6，最後 ccc 的值為何？ (技藝競賽程式組 111) <pre> int ccc=0, i=1, no, ppp; scanf("%d",&no); ppp=no; while(ppp-i>=0) { ppp=ppp-i; ccc++; i=i+2; } printf("%d\n", ccc); </pre> (A) 2 (B) 3 (C) 5 (D) 7	

38	C 程式碼如下，請問 $i = i * j$ 被執行幾次？（技藝競賽程式組 110） (A) 6 次 (B) 7 次 (C) 8 次 (D) 9 次 <pre>int i=1,j=1; while (i<50000){ j++; i=i*j;}</pre>	
39	右圖程式片段，請問有何問題？ <pre>int main(){ int x = 0; While (x <= 999) { x += 20; } }</pre> (A) $(x \leq 999)$ 要改為 $\{x \leq 999\}$ (B) $(x \leq 999)$ 要改為 $x \leq 999$ (C) $(x \leq 999)$ 要改為 $(x \leq 999);$ (D) While 要改為 while	
40	執行右圖程式碼之後，請問最後 y 的值多少？ (A) 1122 (B) 1683(C) 2449 (D) 3366 <pre>#include <stdio.h> int main(){ int x = 3; int y = 0; while (x <= 100){ y += x; x += 3; } printf("y = %d", y); }</pre>	

8-3 巢狀迴圈

迴圈中又有迴圈，稱為巢狀迴圈。巢狀迴圈在程式設計的領域非常重要，這種運算思維是電腦的強項，也是初學者最感頭疼的單元，本節將使用若干範例引領學生征服此運算思維。

自我練習

題號	題目	結果
1	下列程式執行完畢後所輸出值為何？ (技藝競賽程式組) <pre> int main() { int x = 0, n = 5; for (int i=1; i<=n; i=i+1) for (int j=1; j<=n; j=j+1) { if ((i+j)==2) x = x + 2; if ((i+j)==3) x = x + 3; if ((i+j)==4) x = x + 4; } printf ("%d\n", x); return 0; } </pre> (A) 12 (B) 24 (C) 16 (D) 20	
2	請問以下程式，所輸出的第 3 個結果為何？ (技藝競賽程式組) (A) 456 (B) 23 (C) 1 (D) 78910 <pre> #include <stdio.h> main(void){ int n = 4, a = 1; for (int i = 1; i <= n; i++) { for (int c = 1; c <= i; c++){ printf("%d", a); a++; } printf("\n"); } } </pre>	
3	程式填空，以下程式可以找出 2 到 100 的質數。 <pre> int main() { _Bool prime=1; for(int a=2 ;a<=1000;a++){ _____; for(int i=2;i<=a-1;i++){ if (a%i==0) { prime=0;break; } } if(prime) printf("%5d",a) ; } } </pre>	

4	<pre> int main() { int n=4;int s=0; for (int i=1;i<=n;i++) for (int j=1;j<=i;j++) s=s+j; printf("%d",s); } </pre>	
5	<pre> int main() { //質因數分解 int a=521752; int flag=0; int n=0; for (int i =2;i<=a;i++){ n=0; while(a%i==0){ n+=1; a=a/i; } if(n>0){ if (! flag) flag=1; else printf("*"); printf("%d^%d",i,n); } } } </pre>	
6	程式填空。以下程式可以找出三位數阿姆斯壯數。$abc=a^3+b^3+c^3$ <pre> int main() { int i, j, k; int n=0; /* 個數 */ int sum1, sum2; for(i=1; i<=9; i++){ for(j=0; j<=9; j++){ for(k=0; k<=9; k++) { sum1=100*i+10*j+k; _____ if(sum1==sum2){ n++; printf("%d : %4d\n",i,sum1); } } } } } </pre>	

7	程式填空。以下程式可以找出三位數阿姆斯壯數。abc=a³+b³+c³ <pre> int main() { for (int i=100;i<=999;i++){ _____ _____ _____ int d=pow(a,3)+pow(b,3)+pow(c,3); if (i==d) printf("%d\n",i); } } </pre>	
8	請問下面 C 程式中，printf("\n") 共被執行幾次？ (A) 4 (B) 5 (C) 8 (D) 16 <pre> #include <stdio.h> main(){ for (int i=1; i<=4; i++) { for (int j=1; j<5; j++) printf("*"); printf("\n"); } } </pre>	
9	int main(){//技藝競賽程式組107 int i,j; for (i=0;i<3;i++){ for (j=0;j<=i;j++){ printf("*"); } printf("\n"); } (A)  (B)  (C)  (D) 	

10	程式填空。  <pre> int main() { int i, j; for(i=1; i<=5; i++) { for(j=1; _____; j++) { printf(" "); } for(j=1; _____; j++) { printf("*"); } printf("\n");//跳列且歸位 } } </pre>	
11	程式填空。  <pre> int main() { for(int i=1; i<=6; i++) { for(int j=1; j<=i; j++) { printf(" "); } for(int j=1; _____; j++) { printf("*"); } printf("\n");//跳列且歸位 } return 0; } </pre>	

12	程式填空（APCS 試題）  <pre> int main() { int i, j; for(i=1; i<=5; i++) { for(j=1; j<=6-i; j++) { printf(" "); } for(j=1;_____; j++) { printf("*"); } printf("\n");//跳列且歸位 } } </pre>	
13	程式填空。APCS 試題  <pre> int main() { int i, j; for(i=3; i>=1; i--) { for(j=1;_____; j++) { printf(" "); } for(j=1;_____; j++) { printf("*"); } printf("\n");//跳列且歸位 } } </pre>	

14	程式填空  <pre> int main() { int i, j; for(i=1; i<=6; i++) { for(j=1; ____; j++) { _____ } printf("\n");//跳列且歸位 } } </pre>	
15	程式填空。  <pre> int main() { int i, j; for(i=1; i<=5; i++) { for(j=1; ____; j++) { printf(" "); } for(j=1; j<=i; j++) { _____ } printf("\n");//跳列且歸位 } } </pre>	

16	程式填空。  <pre> int main() { int i, j,s=1; for(i=1; i<=5; i++) { for(j=1; j<=i; j++) { printf("%2d ",s); s++; } printf("\n");//跳列且歸位 } } </pre>	
17	程式填空  <pre> int main() { int i, j; for(i=1; i<=5; i++) { for(j=1; _____; j++) { printf(" "); } for(j=1; j<=i; j++) { _____; } printf("\n");//跳列且歸位 } } </pre>	
18	<pre> int main(){//乘法問題 int a=82,b=36,sum=0; for(int i=1; i<=36; i++) sum=sum+a; printf("%d",sum); } </pre>	

19	<pre> int main(){ int s=0; for (int i=1;i<=10;i++) { s=s+i; i=i+1; } printf("%d",s); } </pre>	
20	<pre> int main(){ int s=0,i; for (i=1;s<=20;i++) { s=s+i; i=i+1; } printf("%d,%d",i,s); return 0; } </pre>	
21	<pre> int main(){//開根號 int i, y;int a=9; for (i=0;i<=a;i=i+1){ printf("%d\n",i); y=i*i; if (y>=a){ break; } } printf("%d",i); } </pre>	
22	<pre> int main(){//開根號 double i, y, a=10, d=0.1; for (i=0;i<=a;i=i+d){ y=i*i; if (y>=a){ break; } } printf("%3.1lf",i); } </pre>	

23	<pre> int main(){//開立方 double i, y, a=27, d=0.1; for (i=0;i<=a;i=i+d){ y=i*i*i; if (y>=a){ break; } } printf("%3.1lf",i); } </pre>	
24	<pre> int main(){ int x,y; int a=10; for (x=-a;x<=a;x=x+1){ //printf("%lf\n",i); y=x*x-4*x-5; printf("%d,%d",x,y); if (y==0) printf(",x=%d\n",x); else printf("\n"); } return 0; } </pre>	
25	<pre> int main(){//解方程式 int x,y, a=10; for (x=-a;x<=a;x=x+1){ y=x*x-4*x-5; if (y==0) printf("%d\n",x); } } </pre>	



26	<pre>int main(){ int x,y, a=10, m,mx=10000; for (x=-a;x<=a;x=x+1){ y=x*x-4*x-5; if (y<mx){ m=x; mx=y; } } printf("%d %d",m,mx); return 0; }</pre>	
27	<pre>int main(){ double x,y, a=10; for (x=-a;x<=a;x=x+0.1){ y=x*x+x-0.75; if (fabs(y)<0.01) printf("%3.1f\n",x); } return 0; }</pre>	
28	<pre>int main(){ int a=27; for (int i=1;i<=a;i=i+1){ if (a%i==0){ printf("%d ",i); } } }</pre>	

29	<pre> int main() { // 1~22 找出所有 2 的個數 int a=0, n=0; int b, b1, b2, b3; for (int i=1; i<=22; i=i+1) { b=i; b1=b %10; if (b1==2) n++; b=b/10; b2=b%10; if (b2==2) n++; b3=b/10; if (b3==2) n++; } printf("%d", n); } </pre>	
30	<pre> int main() { // 輾轉相除法 int a=9, b=21, t, r; do { r=a%b; a=b; b=r; } while (r>0); printf(" %d", a); } </pre>	
31	<pre> int main() { // 輾轉相除法 int a=76, b=48, r; while ((a% b) !=0) { r=a%b; a=b; b=r; } printf("%d", b); } </pre>	

32	程式填空。亂數不為零則繼續。 <pre> int main(){ int s=0,a; time_t t; srand((unsigned)time(&t)); do{ a=rand()%6; printf("%d",a); }_____ } </pre>	
33	程式填空。直到有連續兩個亂數相同為止 <pre> int main(){ int s=0,a,b;time_t t; srand((unsigned)time(&t)); do{ a=rand()%6+1;//需要#include <stdlib.h> b=rand()%6+1; printf("%d,%d ",a,b); }_____ } </pre>	
34	程式填空。直到有連續三個亂數相同為止 <pre> int main(){ int s=0,a=0,a1=0,a2=0; time_t t; srand((unsigned)time(&t)); do{ a2=a1; a1=a; a=rand()%6+1; printf("%d ",a); s=s+a; }_____ } </pre>	

35	程式填空。直到亂數和小於等於 8，則程式結束。 <pre> int main() { int s=0,a,b; time_t t; srand((unsigned)time(&t)); do{ a=rand()%6+1; b=rand()%6+1; printf("%d,%d ",a,b); } _____ } </pre>	
36	程式填空。直到後面的亂數大於前面的亂數。 <pre> int main() { int s=0,a,b; time_t t; //需要宣告#include <time.h> srand((unsigned)time(&t)); do{ a=rand()%6+1;//需要#include <stdlib.h> b=rand()%6+1; printf("%d,%d ",a,b); } _____ } </pre>	
37	程式填空。直到有其中兩個亂數相等為止。 <pre> int main() { int s=0,a,b,c; time_t t; //需要宣告#include <time.h> srand((unsigned)time(&t)); do{ a=rand()%6+1;//需要#include <stdlib.h> b=rand()%6+1; c=rand()%6+1; printf("%d %d %d\n",a,b,c); } } </pre>	

38	程式填空。直到有其中兩個亂數相等為止。 <pre> int main() { int sum,a,b,c,d,t; time_t t1; //需要宣告#include <time.h> srand((unsigned)time(&t1)); do{ a=rand()%6+1; b=rand()%6+1; c=rand()%6+1; d=rand()%6+1; if (a>b){t=a;a=b;b=t;} if (b>c){ t=b;b=c;c=t;} if (c>d){t=c;c=d;d=t;} if (a>b){t=a;a=b;b=t;} if (b>c){ t=b;b=c;c=t;} if (a>b){t=a;a=b;b=t;} printf("%d %d %d %d\n",a,b,c,d); }_____ if (a==b) sum=c+d; else if (b==c) sum=a+d; else sum=a+b; printf("%d",sum); } </pre>	
39	<pre> int main() { //二分猜值法 double y,x,x1,x2,t; y=9;x1=0;x2=y; do { x=(x1+x2)/2; t=x*x; if (t<y) x1=x; //猜的太小，調整下限為x else x2=x; //猜的太大，調整上限為x } while (fabs(x2-x1) >0.01) ; //fabs() 絕對值 printf(" %4.2f ",x); } </pre>	

40	<pre> int main() { //開立方 double y,x,x1,x2,t; y=27;x1=0;x2=y; do { x=(x1+x2)/2; t=x*x*x; if (t<y) x1=x; //猜的太小，調整下限為x else x2=x; //猜的太大，調整上限為x } while (fabs(x2-x1) >0.01) ; printf(" %4.1f ",x); } </pre>	
41	<pre> int main() { //除法 double y,x,z,x1,x2,t; y=27;z=3;x1=0;x2=y; do { x=(x1+x2)/2; t=x*z; if (t<y) x1=x; //猜的太小，調整下限為x else x2=x; //猜的太大，調整上限為x } while (fabs(x2-x1) >0.01) ; printf("%4.1f ",x); } </pre>	
42	<pre> int main() { //進位轉換 int a=18,n=2,r; char t[10]="",b[10]=""; while(a>0) { r=a % n; itoa(r,t,10); //轉字串 strcat(b,t); //字串b=b+t a=a/n; } strrev(b); //反轉 printf("%s",b); return 0; } </pre>	

43	<pre> int main() { //APCS秘密差 int a=263541, n=0, r; while (a>0){ r=a%10; n=n+r; a=a/10; r=a%10; n=n-r; a=a/10; } printf("%d", abs(n)); } </pre>	
44	<pre> int judge(int n){ //APCS完全奇數 int t; do{ t=n %10; if (t%2==0) return(0); //找到偶數就離開 n=n/10; }while (n>0); //通過層層考驗，就是完全奇數 return(1); } int main(){ int n=13256, m=n, p=n, odd; //bool //found big odd=judge(m); while(odd==0) { m=m+1; odd=judge(m); } printf("%d\n", m); //found small odd=judge(p); while(odd==0) { p=p-1; odd=judge(p); } printf("%d\n", p); printf("%d\n", m-p); return 0; } </pre>	

45	<pre> int main(){//APCS 判斷是否遞增 int d=19456,m,n=10,inc=1; do{ m=d %10; d=d/10; if (m>=n){ inc=0; printf("%d\n",m); break; } n=m; }while (d>0); printf("%d\n",inc); } </pre>	
46	<pre> int judge(int d){//APCS遞增數 int inc=1,m,n=10; do{ m=d %10; d=d/10; if (m>=n){ inc=0; return inc; } n=m; }while (d>0); } int main(){ int d=19456,m,n=10,inc=1; for (int i=1;i<=25;i++){ inc=judge(i); if (inc) printf("%d\n",i); } return 0; } </pre>	

47	<pre> int judge(int d){ int inc=1,m,n=10; do{ m=d %10; d=d/10; if (m>=n){ inc=0; return inc; } n=m; }while (d>0); } int main(){ int m,n=10,inc=1; for (int i=20;i<=23;i++){ inc=judge(i); if (inc) printf("%d\n",i); } return 0; } </pre>	
48	<pre> int main() { //積分 double sum=0, dx=0.01, x,y; for(x=0; x<=10; x=x+dx){ y=x; //y=x sum=sum+y*dx; //高度*切割的寬度 } printf("%4.1f\n",sum); } </pre>	
49	<pre> int main() { //sin(θ) 正半週面積 float sum=0, dx=0.01, y; for(float x=0; x<=M_PI; x=x+dx){ y=sin(x); sum=sum+y*dx; //高度*切割的寬度 } printf("%4.1f",sum); } </pre>	

50	<pre> int main() { //圓面積 float sum=0,dx=0.001,y; for(float x=0; x<=1; x=x+dx){ y=sqrt(1-x*x); sum=sum+y*dx; //高度*切割的寬度 } printf("%4.2f",4*sum); } </pre>	
51	<pre> int main() { //計算e int i; double e,t1,t2; for(i=0 ;i<=10;i++) { t1=pow(10,i); //載入math.h t2=1+1/t1; e=pow(t2,t1); } printf("%6.3f\n",e); } </pre>	

課後習題

一、填充題

(請先追蹤程式寫出執行結果，然後鍵入以下程式，核對自己答案。#include <stdio.h> #include <stdlib.h> 請自己補上)

題號	題目	結果
1	<pre> int i, total = 0; //統測111 for(i = 1; i < 8; i += 2) total += i; printf("%d",total); </pre>	
2	<pre> int y, r, a = 30, b = 42; //統測111 r = a%b; while(r != 0) { a = b; b = r; r = a%b; } y = b; printf("%d",y); </pre>	

3	<pre> int main() { int k=4; int m=1; for (int i=1;i<=5;i++){ //cout<<k<<" "<<m; for (int j=1;j<=k;j++){ printf(" "); } for (int j=1;j<=m;j++){ printf("*"); } printf("\n"); k=k-1; //m=m+1; m=2*i+1;//修改處 } return 0; } </pre>	
4	<pre> void main () { //APCS int a=0; int b=1; int i, temp, N=12; for (i=2; i<=N; i=i+1) { temp = b; b=a+b ; a = temp; printf ("%d\n", (b)); } } </pre>	
5	<pre> void main () { //APCS int i = 76; int j = 48,k; while ((i % j) != 0) { k=i%j; i=j; j=k; } printf ("%d\n", j); } </pre>	

6	<pre> void main() { //APCS for (int i=0; i<=10; i=i+1) { printf ("%d ", i); i = i + 1; } printf ("\n"); } </pre>	
7	<pre> void main() { //APCS int a=0, n=1; for (int i=1; i<=n; i=i+1) for (int j=i; j<=n; j=j+1) for (int k=1; k<=n; k=k+1) a = a + 1; printf("%d",a); } //請分別寫出n=1~6,10,100的輸出值。 </pre>	
8	<pre> void main() { //APCS105/10 for (int i=0; i<=3; i=i+1) { for (int j=0; j<i; j=j+1) printf(" "); for (int k=6-2*i; k>0 ; k=k-1) printf("*"); printf("\n"); } } </pre>	
9	<pre> void main() { //APCS10510 int i=2, x=3; int N=65536; while (i <= N) { i = i * i * i; x = x + 1; } printf ("%d %d \n", i, x); } </pre>	

10	以下程式片段無法正確列印 20 次的 "Hi!"，請問下列哪一個修正方式仍無法正確列印 20 次的 "Hi!" ？ //APCS10603 <pre>for (int i=0; i<=100; i=i+5) { printf ("%s\n", "Hi!"); }</pre> (A) 需要將 i<=100 和 i=i+5 分別修正為 i<20 和 i=i+1 (B) 需要將 i=0 修正為 i=5 (C) 需要將 i<=100 修正為 i<100; (D) 需要將 i=0 和 i<=100 分別修正為 i=5 和 i<100	
11	<pre>int main() { //APCS10603 int x = 0, n = 5; for (int i=1; i<=n; i=i+1) for (int j=1; j<=n; j=j+1) { if ((i+j)==2) x = x + 2; if ((i+j)==3) x = x + 3; if ((i+j)==4) x = x + 4; } printf ("%d\n", x); return 0; }</pre>	

MEMO

陣列

前面處理少量的資料，可以個別設定一些單一變數。但是如果資料龐大，例如，若有資料如下：

```
3, 8, 4, 7, 2, 9
```

要求極大、極小、平均等運算，若還是使用 6 個變數儲存，則程式將會很冗長。本章將會介紹另一種資料結構『陣列』，然後配合上一章的 for 或 while 迴圈，就可以以非常精簡的程式，處理以上大批資料。其次，陣列還有維度之分，分別是一維、二維、三維等等，要如何選擇維度就要看您資料運算的方式，若只有同批資料的關係，例如，上例就是一維資料陣列就可以。但是，在一個班級的國、英、數、自然、社會等 5 科成績裡，我們不僅要求縱向的各科平均，還要求橫向的每人平均，此時就需要使用二維陣列，請看本章以下說明。

9-1 一維陣列

處理少量的資料，可以為每一筆資料設定一個變數，但若資料數量很多，例如，若有資料如下：

```
3, 8, 4, 7, 2, 9
```

要求其和、極大、極小呢？是不是指派 6 個變數，當然也可以，但是程式會非常冗長，請看範例 7-3a。為了改善此一問題，就要使用本節的陣列型態了。因為陣列型態，可以使用『陣列索引』配合 for 與 while 迴圈而簡化程式，此即為本章重點。例如：以上資料就可宣告陣列如下：

```
int a[7];
```

9-2 程式設計實習進階

int 是陣列資料型態，陣列型態可為 3-2 節任一型態，a 是陣列變數名稱，變數名稱也應該遵循 4-1 節變數命名規則。然後就可依序指派這些資料到陣列，程式如下：

```
a[1]=3;a[2]=8;a[3]=4;a[4]=7;a[5]=2;a[6]=9;
```

中括號內的 1,2,3 稱為『陣列索引』，簡稱索引。人類索引編號通常從 1 號開始使用，但電腦卻是從 0 號開始，所以宣告

```
int a[7]
```

其實是使用 0,1,2,3,4,5,6 等 7 個位置。索引 0 號可用可不用，陣列索引從 0 開始，有其獨到用法，若配合迴圈與取餘，可以執行一些無限循環的功能。例如：以下程式，可以讓陣列索引在 0 到 6 之間無限循環。

```
void loop() {  
    PORTA=a[i];  
    delay(1000);  
    i=(i+1) %7;//保障i在0,1,2,3,4,5,6循環  
}
```

✂ 陣列的存取

陣列的存取都要靠索引，例如：以下程式可重新指派 a 陣列索引 3 的值。

```
a[3]=1;
```

以下程式可將 a 陣列索引 3 的值指派給變數 b，但請留意兩者資料型態要相符。

```
int b=a[3];  
int b;b=a[3];
```

✂ 陣列宣告與初值設定

單一變數可宣告變數的同時就指派初值，陣列也可以。例如：前面陣列的宣告與初值指派，程式可以簡化如下：

```
int a[7]={0,3,8,4,7,2,9};
```

本例索引 0 不用，也要給予 1 個 0，以上資料 3 才會從索引 1 開始放。事實上，陣

列宣告時，您可以不用指派陣列的個數，所以以下程式就可以。

```
int a[]={0,3,8,4,7,2,9};
```

但是陣列的大小給的太小也不行。例如，以下程式就不行：

```
int a[3]={0,3,8,4,7,2,9};
```

其次，給大一點當然可以，以下程式，資料僅給索引 0~6，索引 7,8,9 沒有給初值，電腦會自動指派為 0。

```
int a[10]={0,3,8,4,7,2,9};
```

陣列的資料結構的最大優點是，陣列結構可配合迴圈（前面單一變數不行），例如，以下程式配合陣列的索引，即可以以精簡的程式，處理大批數據。

```
int a[]={0,3,8,4,7,2,9};
int sum=0;
for(int i=1; i<=6; i++) {
    a[i]=a[i]*2;//每一個都乘以2
    sum+=a[i]; //相加每一元素
}
```

題號	題目	結果
1	下列 C 語言片段程式輸出結果為何？ <pre>int score[10] = {80,90,95,75,100,80,85,95,90,60}; printf("%d", score[5]);</pre> (A) 75 (B) 80 (C) 85 (D) 100	
2	下列程式執行時，輸入 5 後，輸出結果為？（技藝競賽程式組 112） <pre>#include <stdio.h> int fun2(int n){ int a=2, b=1; for (int i=1; i<=n; ++i){ b+=i*(a-1); a<<=1; } return b; } int main(void){ int y; puts("輸入1個正整數："); scanf("%d",&y); printf("%d", fun2(y)); }</pre> (A) 29 (B) 89 (C) 244 (D) 622	

3	若 A[1]、A[2]，和 A[3] 分別為陣列 A[] 的三個元素 (element)，下列那個程式片段可以將 A[1] 和 A[2] 的內容交換？（技藝競賽程式組 112） (A) A[1] = A[2]; A[2] = A[1]; (B) A[3] = A[1]; A[1] = A[2]; A[2] = A[3]; (C) A[2] = A[1]; A[3] = A[2]; A[1] = A[3]; (D) A[2] = A[1]; A[1] = A[2];	
4	請問下列哪一個是 C 語言陣列索引預設的起始值？ （技藝競賽修護組 111） (A) 0 (B) -1 (C) 1 (D) 2。	
5	有一程式如右圖所示，執行後，顯示值為何？（技藝競賽程式組 110） (A) 0 (B) 1 (C) 2 (D) 9 <pre> #include <stdio.h> int f (int a[] , int n) { int index = 0; for (int i=1; i<=n-1; i=i+1) { if (a[i] >= a[index]) { index = i; } } return index;} main(){ int a[9]={1,2,3,4,7,5,9,6,8}; int ret=0; ret=f(a,9); printf("%d",ret); } </pre>	
6	請問下列程式片段執行後，輸出的第 12 個數值是？ （技藝競賽程式組 109） (A) : 88 (B) : 143 (C) : 232 (D) : 376 <pre> main() { const int n = 15; unsigned long long int s[15]; s[0] = 0; s[1] = 1; for (int i = 2; i < n; i++) s[i] = s[i - 2] + s[i - 1] + 1; for (int i = 0; i < n; i++) printf("%llu\n", s[i]); } </pre>	

7	請問以下程式，輸出結果為何？ (技藝競賽程式組 109) (A) :32 60 59 42 42 27 (B) :27 32 42 42 59 60 (C) :60 59 42 42 32 27 (D) :60 59 32 42 42 27 <pre> main() { int x[6] = { 32, 42, 60, 27, 42, 59 }; for (int j = 2; j < 6; j++) { int k = x[j]; int i = j - 1; while ((i > 0) && (x[i] < k)) { x[i + 1] = x[i]; i--; } x[i + 1] = k; } for (int j = 0; j < 6; j++) printf("%i ", x[j]); } </pre>	
8	<pre> int main(){//技藝競賽 float dataB[4]={2.2,2.8,-2.4,-1.8}; int j,result=0; for (j=0;j<4;j++){ result=result+round(dataB[j]);//round() is四捨五入 } printf("%d",result); return 0; } </pre> (A) — 1 (B) 0 (C) 1 (D) 2	
9	<pre> int main() { int a[] = { 70, 60, 90, 40, 50 }; int i, ma, mi, s; ma = a[0]; mi = a[0]; s = a[0]; for (i = 1; i <= 4; i++) { s += a[i]; if (a[i]>ma) ma = a[i]; if (a[i]<mi) mi = a[i]; } printf("%d\n", s / 5); printf("%d\n" , ma); printf("%d\n" , mi); } </pre>	

10	<pre> int main() { //統計111 此題考沒給初值，預設0 int Sum = 0, Array[4] = {1, 2, 3}, i; for(i = 1; i < 4; i++) Sum += Array[i]; printf("%d", Sum); } </pre>	
11	<pre> int main() { int a[10]={-1,0,3,4,-5,8}; int np=0, n0=0, nm=0; for (int i=0;i<=6;i++){ if (a[i]<0) nm++; else if (a[i]==0) n0++; else np++; } printf("%d\n",nm); printf("%d\n",n0); printf("%d\n",np); return 0; } </pre>	
12	<pre> int main() { //次數統計 int a[]={1,3,2,6,4,2,1,3,4,5}; int alen=(int)(sizeof(a) / sizeof(a[0])); int b[7]={0,0,0,0,0,0,0}; for (int i=0;i<=alen-1;i++) b[a[i]]=b[a[i]]+1; printf("%d\n",b[3]); } </pre>	
13	<pre> int main() { //區間人數統計 int a[]={11,33,22,66,44,22,11,33,44,55,88,99,77,66,100}; int alen=(int)(sizeof(a) / sizeof(a[0])); int b[11]={0,0,0,0,0,0,0,0,0,0,0}; for (int i=0;i<=alen-1;i++) b[a[i]/10]=b[a[i]/10]+1; b[9]=b[9]+b[10]; printf("%d",b[9]); } </pre>	



14	以下程式假設輸入 3，寫出輸出結果。 <pre> int main() { int n=5, a[n+1],i,j,b; // 陣列初始化 for(i=1; i<=n; i++) a[i]=0; for(i=1; i<=n; i++){ scanf("%d",&b); a[b]=1; for (j=1;j<=n;j++) if (a[j]==0) printf("%2d",j); } } </pre>	
15	<pre> int main() { //進位轉換 char c[]={'0','1','2','3','4','5','6','7','8','9', ,'A','B','C','D','E','F'}; char d[20]; int a=18, b=16, r, n=0,i; for (i=20 ;i>=0 ;i--) d[i]=' '; //協助清空記憶體 n=0; //索引從0開始 int al=a; while(a>0) { r=a%b; d[n]=c[r]; //將餘數轉為字元，且從索引0開始放 n++; a=a/b; } for (i=6 ;i>=0 ;i--) printf("%c",d[i]); } </pre>	
16	<pre> int main() { //月份轉天數 int a[]={0,31,28,31,30,31,30,31,31,30,31,30,31}; int b1=3,b2=1,c=0; for (int i=1;i<=b1-1;i++) c=c+a[i]; c=c+b2; printf("%d",c); } </pre>	

17	<pre> int main() { //日期的計算 int a[]={0,31,30,31,30,31,30,31,31,30,31,30,31}; int b0=2018,b1=9,b2=30,b3=0; int d0=2018,d1=11,d2=1,d3=0; for (int i=1;i<=b1-1;i++) b3=b3+a[i]; b3=b3+b2; for (int i=1;i<=d1-1;i++) d3=d3+a[i]; d3=d3+d2; printf("%d",d3-b3); } </pre>	
18	程式填空。以下程式，可練習電子琴。 <pre> int main() { int b[]={0,523,587,659,698,784,880,988,1046}; char a; int c; while(1){ //無限迴圈，請看下一章 a= getche(); //輸入後，不用按Enter _____ printf("%d\n",c); _beep(b[c],500); } } </pre>	
19	程式填空。以下程式，可練習單字的打字練習。 <pre> int main() { char b[][10]={"look","at","one","two","tree", "and","day","book","from","go"}; char c[10]; int r=0,w=0; for (int i=0;i<=9;i++){ printf("%s\n",b[i]); scanf("%s",&c); _____{ r=r+1; printf("Right\n"); } else{ w=w+1; printf("Wrong\n"); } } printf("Right=%d,wrong=%d",r,w); } </pre>	



20	<pre> int main() { //求幾月幾日 int a[]={0,31,28,31,30,31,30,31,31,30,31,30,31}; int d=32, m=1; while (d>a[m]){ d=d-a[m]; m++; } printf("%d,%d",m,d); } </pre>	
21	<pre> int main() { //求星期幾 int a[]={0,31,30,31,30,31,30,31,31,30,31,30,31}; char d[][3]={"日","一","二","三","四","五","六"}; int b1=4,b2=2,c=0; for (int i=1;i<=b1-1;i++) c=c+a[i]; c=c+b2-1; printf("%d\n",c); c=(c+3)%7; printf("%s",d[c]); } </pre>	
22	<pre> int main() { //陣列合併 int a[]={3,4,7}; int b[]={4,8}; int c[20]; int alen=(int)(sizeof(a) / sizeof(a[0])); int blen=(int)(sizeof(b) / sizeof(b[0])); for (int i=0;i<=alen-1;i++) c[i]=a[i]; for (int i=0;i<=blen-1;i++) c[i+alen]=b[i]; for (int i=0;i<=alen+blen-1;i++) printf("%d ",c[i]); } </pre>	

23	<pre> int main() { //合併排序法 int a[]={3,4}; int b[]={4,5,6};int c[20]; int alen=(int)(sizeof(a) / sizeof(a[0])); int blen=(int)(sizeof(b) / sizeof(b[0])); int p=0,q=0,r=0; do { if (a[p]<b[q]){ c[r]=a[p];r++; p++; } else{ c[r]=b[q]; r++; q++; } }while (p<alen-1 && q< blen-1); while(p<=alen-1){ c[r]=a[p]; r++; p++; } while(q<=blen-1){ c[r]=b[q];r++; q++; } for (int i=0;i<=alen+blen-1;i++) printf("%d ",c[i]); return 0; } </pre>	
24	<pre> int main() { //交集 int a[]={3,4,7};int b[]={4,6};int c[20]; int c1=0; int d[20]; int alen=(int)(sizeof(a) / sizeof(a[0])); int blen=(int)(sizeof(b) / sizeof(b[0])); for (int i=0;i<=alen-1;i++){ for (int j=0;j<=blen-1;j++){ if (a[i]==b[j]){ c[c1]=a[i]; c1++; } } } for (int i=0;i<=c1-1;i++) printf("%d ",c[i]); } </pre>	

25	<pre> int main() { //聯集 int a[]={3,4,7};int b[]={4,6};int c[20]; int cl=0; int d[20]; int alen=(int)(sizeof(a) / sizeof(a[0])); int blen=(int)(sizeof(b) / sizeof(b[0])); //聯集 for (int i=0;i<=alen-1;i++){ d[i]=a[i]; } int dl=alen;int repeat=0; for (int i=0;i<=blen-1;i++){ for (int j=0;j<=alen-1;j++){ if (b[i]==a[j]){ repeat=1; break; } } if (repeat==0){ d[dl]=b[i]; dl++; } repeat=0; } for (int i=0;i<=dl-1;i++) printf("%d ",d[i]); } </pre>	
26	<pre> int main() { int a[]={3,1,4,1,4,7};int b[20]; int alen=(int)(sizeof(a) / sizeof(a[0])); b[0]=a[0]; int c=0; int repeat=0; for (int i=1;i<=alen-1;i++){ for (int j=0;j<=c;j++){ if (a[i]==b[j]){ repeat=1; break; } } if (repeat==0){ c=c+1; b[c]=a[i]; } repeat=0; } for (int i=0;i<=c;i++) printf("%d ",b[i]); } </pre>	

9-2 多維陣列

含有兩個或兩個以上索引值的陣列稱為多維陣列，一般最常應用的是二維陣列，下面我們就以二維陣列為例，來介紹如何使用多維陣列。

多維陣列變數宣告

資料型態 陣列變數名稱[陣列的大小] [陣列的大小] . . . ;

以下敘述宣告一個二維的整數型態陣列變數。

```
int    a[3][5];
```

你可以把二維陣列想像成一個表格，所以上式 `a` 這個陣列就像一個有 3 列 5 行可以存放 15 個元素的表格。和一維陣列一樣，不管是列或行，其索引值皆由 0 開始。為了讓讀者更清楚二維陣列的資料結構，我們將 `a` 陣列的所有元素以表格表示如下：

	第0行	第1行	第2行	第3行	第4行
第 0 列	<code>a[0][0]</code>	<code>a[0][1]</code>	<code>a[0][2]</code>	<code>a[0][3]</code>	<code>a[0][4]</code>
第 1 列	<code>a[1][0]</code>	<code>a[1][1]</code>	<code>a[1][2]</code>	<code>a[1][3]</code>	<code>a[1][4]</code>
第 2 列	<code>a[2][0]</code>	<code>a[2][1]</code>	<code>a[2][2]</code>	<code>a[2][3]</code>	<code>a[2][4]</code>

多維陣列變數值的存取

以下敘述可以將第 0 列第 1 行這個陣列元素的值設定為 1。

```
a[0][1]=1;
```

以下敘述可以將第 0 列第 1 行這個陣列元素的值指定給變數 `b`。

```
b= a[0][1];
```

多維陣列變數值的初始化

假設有表格資料如下：

10	11	12	13	14
20	21	22	23	24
30	31	32	33	34

若以二維陣列儲存，則其敘述如下：

```
int a[3][5]={ {10, 11, 12,13, 14},
              {20, 21, 22, 23, 24},
              {30, 31, 32, 33, 34} };
```

以上敘述示範了二維陣列變數值的初始化，我們以下表顯示每個陣列元素的初始值。

	第0行	第1行	第2行	第3行	第4行
第 0 列	10	11	12	13	14
第 1 列	20	21	22	23	24
第 2 列	30	31	32	33	34

題號	題目	結果
1	假設某程式使用 C 語言的敘述及其示意圖如下所示，請問那一個沒有指定初始值？ (技藝競賽修護組 110) C 語言敘述：int c[2][3]= {0, 1, 2, 3, 4}; (A) C[0][0] (B)C[1][2] (C) C[1][0] (D) C[0][2]。	
2	int c[2][3]= {0, 1, 2, 3, 4}; printf("%d",c[1][2]); (A) 0 (B) 1 (C) 2 (D) 3	
3	int c[2][3]= {0, 1, 2, 3, 4,5}; printf("%d",c[1][2]); (A) 0 (B) 1 (C) 2 (D) 5	
4	有一 C 程式片段 (如下圖) 能夠執行，下列空白處填入何程式碼會產生錯誤？ (技藝競賽程式組 109) (A):int data[3][3] = { {11 , 12 , 13}, {21, 22, 23}, {31, 32, 33} }; (B):int data[3][3] = { 11 , 12 , 13, 21, 22, 23, 31, 32, 33 }; (C):int data[][3] = { {11 , 12 , 13}, {21, 22, 23}, {31, 32, 33} }; (D):int data[3][] = { {11 , 12 , 13}, {21, 22, 23}, {31, 32, 33} };	

	<pre> #include <stdio.h> int main() { int i, j; for (i=0; i<3; i++) { for (j=0; j<3; j++) printf("%3d", data[i][j]); printf("\n"); } } </pre>	
5	<pre> int main() { // 技藝競賽程式組108 int dataA[3][3] = {{5, 12, 23}, {8, 6, 9}, {9, 4, 5}}; int j, k, value = 0; for (j = 0; j < 3; j++) { for (k = 0; k < 3; k++) { value = value + (dataA[j][k] % (k + 1)); } } printf("%d", value); return 0; } </pre> (A) 4 (B) 5 (C) 8 (D) 9	
6	<pre> int main() { // 二維矩陣相加 int i, j, k; const int m = 2; const int n = 3; int a[2][3] = {{1, 2, 3}, {4, 5, 6}}; int b[2][3] = {{1, 2, 0}, {2, 1, 3}}; int c[2][3]; for (i = 0; i < m; i++) for (j = 0; j < n; j++) c[i][j] = a[i][j] + b[i][j]; for (i = 0; i < m; i++) { for (j = 0; j < n; j++) printf("%4d", c[i][j]); printf("\n"); } } </pre>	

7	<pre> int main() { int a[][3]={1,2,0}, {0,5,6}, {7,8,9}, {10,11,0}}; int n=0;int sum=0; for (int i=0;i<=3;i++){ for (int j=0;j<=2;j++){ if (a[i][j]==0){ n++; } sum=sum+a[i][j]; } } printf("%d\n",n); printf("%d\n",sum); } </pre>	
8	程式填空。以下程式可計算每個人的平均、每科平均。 <pre> int main() { int a[7][7]={0,0,0,0,0,0}, {1,50,60,70,0,0}, {2,30,40,50,0,0}, {3,70,80,90,0,0}, {4,66,77,88,0,0}, {5,22,33,44,0,0}, {0,0,0,0,0,0}}; int i, j, k; const int stdnum=5; //每人平均 for(j=1; j<=stdnum; j++) { for(k=1; k<=3; k++) _____ a[j][4]=a[j][4]/3; } //各科平均 for(j=1; j<=4; j++) { for(k=1; k<=stdnum; k++) _____; a[stdnum+1][j]/=stdnum; } //不及格科數 </pre>	

```

for(j=1; j<=stdnum+1; j++)
for(k=1; k<=3; k++)
    if(a[j][k]<60)
        _____;

//輸出
printf("    No chin  eng math  avg fail\n");
for(i=1; i<=stdnum+1; i++) {
for(j=0; j<6; j++)
    printf("%5d",a[i][j]);
printf("\n");
}
}

```

```

No chin  eng math  avg fail
1  50   60   70   60   1
2  30   40   50   40   3
3  70   80   90   80   0
4  66   77   88   77   0
5  22   33   44   33   3
0  47   58   68   58   2

```

```

9  int main() {
    int a[6][7]={0,0,0,0,0,0,0},
           {1,0,2,4,6,0,0},
           {2,8,0,2,1,0,0},
           {3,7,0,0,9,0,0},
           {0,0,0,0,0,0,0},
           {0,0,0,0,0,0,0}};

    const int stdnum=3;
    for(int i=1; i<=stdnum+2; i++) {
        for(int j=0; j<=6; j++){
            printf("%5d",a[i][j]);
        }
        printf("\n");
    }
    printf("\n");
    for(int i=1; i<=stdnum; i++) {
        for(int j=1; j<=4; j++) {
            a[i][5]=a[i][5]+a[i][j];
            if(a[i][j]==0)
                a[i][6]++;
        }
    }
}

```

	<pre> for (int i=1;i<=4;i++){ for (int j=1;j<=stdnum;j++){ a[stdnum+1][i]=a[stdnum+1][i]+a[j][i]; if (a[j][i]==0) a[stdnum+2][i]++; } } printf("%3d",a[2][5]); printf("%3d",a[2][6]); printf("%3d",a[4][2]); printf("%3d",a[5][2]); } </pre>	
10	大部分程式語言都是以列為主的方式儲存陣列。在一個 8x4 的陣列 (array) A 裡，若每個元素需要兩單位的記憶體大小，且若 A[0][0] 的記憶體位址為 108 (十進制表示)，則 A[1][2] 的記憶體位址為何？ (APCS10503 試題) (A) 120 (B) 124 (C) 128 (D) 以上皆非	
11	有一段 C 程式碼如下，填空處將定義二維陣列 m。請問哪一個定義敘述可以讓程式碼最後列印 max 時的值為 2？ (技藝競賽程式組 111) <u> 填空處 </u> <pre> int max=0; for (int i=0; i<3; i+=1) { int deg=0; for (int j=0; j<3; j+=1){ deg += m[i][j]; } if (deg>max) max=deg; } printf("%d\n", max); </pre> (A) int m[3][3] = {{1, 1, 1}, {0, 1, 1}, {0, 0, 1}}; (B) int m[3][3] = {{1, 0, 0}, {0, 1, 0}, {0, 0, 1}}; (C) int m[3][3] = {{0, 0, 1}, {1, 1, 1}, {1, 0, 1}}; (D) int m[3][3] = {{0, 1, 0}, {1, 0, 1}, {1, 1, 0}};	

9-3 字元陣列

在 C 語言中，基本資料型態並無字串資料型態（C++ 才有字串資料型態）。例如：無法使用

```
string a;
```

若要處理字串，則要使用字元陣列或指標，指標請參看 10-3 節。在 C 語言中，字串是利用簡單的一維字元陣列加上一個字串結束字元 '\0'（NULL）來表示。例如：以下列敘述是宣告並初始化變數 a 的初值為字串 "abc"。其次，請注意字串使用雙引號（" "），字元使用單引號（' '），如果誤用則會造成錯誤。

```
char a[ ]="abc";
```

上式的效果等於

```
char b[ ]={'a', 'b', 'c', '\0'}; // 將字元單獨放入字元陣列中
```

雖然第一種字串宣告方式中的字串 "abc" 的長度為 3，但是編譯器編譯時會自動在字串末端加上一個結束字元（'\0'）表示字串的結束。因此，所宣告的 a 字元陣列在記憶體中實際上是占了 4 個字元組。請鍵入以下程式，並觀察執行結果。

```
int main() {  
    char a[]="abc";//編譯器會自動放\0  
    char b[]={ 'a', 'b', 'c', '\0'};//編譯器不會自動放'\0'  
    char c[]={ 'a', 'b', 'c' };//編譯器不會自動放'\0'  
    printf("%s\n",a);  
    printf("%s\n",b);  
    printf("%s\n",c);  
    return 0;  
}
```

字元與字串

字串既然是字元陣列，所以字元的處理也同陣列索引的處理。例如：

```
int main() {  
    char a[]="abc";//編譯器會自動放\0  
    printf("%s\n",a);  
}
```

```

printf("%c\n",a[0]);
printf("%c\n",a[1]);
printf("%c\n",a[2]);
printf("%c\n",a[3]);
char b=a[0];
printf("%c\n",b);
return 0;
}

```

✂ 字串複製

字串的複製，並不能使用

```
c=a; //C++才可以
```

一定要使用字元陣列複製函式 `strcpy`，請參考 11-3 節。

```
strcpy(c,a);
```

👉 自我練習

1. 請鍵入以下程式，寫出執行結果。

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h> //使用strcmp, strcpy
int main()
{
    char a[]="Horng";
    char b[10];
    strcpy(b,a);
    printf("%s\n",b);
    return 0;
}

```

2. 若宣告一個字元陣列 `char str[20] = "Hello world!"`; 該陣列 `str[12]` 值為何？
(APCS10510)

(A) 未宣告 (B) \0 (C) ! (D) \n

字串輸入與輸出

字串的輸出入和其他資料型態一樣都可以使用 `printf()`、`scanf()`，唯須注意下列事項：

1. 列印格式需要置換為 `%s`。
2. 使用 `scanf()` 時必須使用取址符號 (`&`)，但輸出時就不用再取址，程式如下：

```
char a[4];
scanf("%s", &a);
printf("%s", a);
```

3. 若所輸入的字串中包含空白字元，則必須以讀入字元 (`getchar()`) 的方式將字串輸入到陣列中，並在陣列末端加入一結束字元 `'\0'`。

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    char c, d[10];int i=0;
    while ((c=getchar( )) != EOF && c!='\n'){
        // 本例假設輸入d="abc xyz"
        d[i]=c;
        i++;
    }
    d[i]='\0';
    printf("%s\n", d);
    return 0;
}
```

4. 請鍵入以下程式，寫出執行結果。

```
int main()
{
    char c, d[10];int i=0;
    scanf("%s", &d); // 本例假設輸入"abc xyz"
    printf("%s\n", d);
    return 0;
}
```


☞ 字串陣列

若有連續字串如下：

Hand、Mary、洪煥維、陳惠敏、王一平

請問該如何儲存呢？答案是使用字元二維陣列，請鍵入以下程式，並觀察執行結果。

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    char a[][10]={"Hand","Mary","洪煥維","陳惠敏","王一平"};
    printf("%c\n",a[0][0]); //H
    printf("%s\n",a[0]); //Hand
    printf("%c%c\n",a[2][0],a[2][1]); //洪, 中文佔兩個字元
    printf("%s\n",a[2]); //洪煥維
    printf("\n");
    int i,j;
    int n=5;
    // 輸出全部
    for (i=0;i<=n-1;i++){
        printf("%10s",a[i]);
    }
    printf("\n");
}
```

☞ 程式說明

1. 字串宣告時，僅可省略最高維度的大小。本例預設字串長度最多為 10，程式如下：

```
char a[][10]={"Hand","Mary","洪煥維","陳惠敏","王一平"};
```

☞ 二維字串陣列

若有二維資料如下：

姓名	數學	APCS	平均
Tom	90	80	
John	60	90	
Mary	80	60	

請問如何儲存，如何計算平均。以下是 C 語言程式。請鍵入以下程式，並寫出執行結果。

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    char a[][4][10]={{"Tom","90","80",""},
                     {"John","60","90",""},
                     {"Mary","80","60",""}};
    printf("%c\n",a[0][0][0]);//T
    printf("%c\n",a[0][0][1]);//o
    printf("%s\n",a[0][0]);//Tom
    printf("%s\n",a[0][1]);//90
    printf("%s\n",a[1][0]);//John
    int i,j;
    //輸出
    for (i=0;i<=2;i++){
        for (j=0;j<=3;j++){
            printf("%4s",a[i][j]);
        }
        printf("\n");;
    }
    //計算總分、平均
    int sum=0,avg;
    for (i=0;i<=2;i++){
        sum=0;
        for (j=1;j<=2;j++){
            sum=sum+atoi(a[i][j]);//轉數值，請看11-3
        }
        avg=sum/2;
        itoa(avg,a[i][3],10);//轉字串，請看11-3
    }
    printf("\n");
```

```
//輸出
for (i=0;i<=2;i++){
    for (j=0;j<=3;j++){
        printf("%4s",a[i][j]);
    }
    printf("\n");
}
return 0;
}
```

以下則是 C++ 程式列印：

```
#include <iostream>
#include <cstring>
using namespace std;
int main()
{
    string a[][4]={{"Tom","90","80",""},
                   {"John","60","90",""},
                   {"Mary","80","60",""}};
    cout<<a[0][0][0]<<endl;//T
    cout<<a[0][0][1]<<endl;//o
    cout<<a[1][1][0]<<endl;//6
    cout<<a[0][0]<<endl;//Tom
    cout<<a[0][1]<<endl;//90
    cout<<a[1][0]<<endl;//John
    int i,j;
    //輸出
    for (i=0;i<=2;i++){
        for (j=0;j<=3;j++){
            cout<<a[i][j]<<" ";
        }
        cout<<endl;
    }
    //計算總分、平均
    int sum=0,avg;
    for (i=0;i<=2;i++){
        sum=0;
        for (j=1;j<=2;j++){
            sum=sum+stoi(a[i][j]); //轉數值，請看9_3
        }
    }
}
```

```

        //sum=sum+atoi(a[i][j]); //C style轉數值，請看9_3
    }
    avg=sum/2;
    a[i][3]=to_string(avg); //轉字串，請看9_3
    //itoa(avg,a[i][3],10); //C style轉字串，請看9_3
}
cout<<endl;
//輸出
for (i=0;i<=2;i++){
    for (j=0;j<=3;j++){
        cout<<a[i][j]<<" ";
    }
    cout<<endl;
}
return 0;
}

```

自我練習

題號	題目	結果
1	C 語言如何使用變數 a 儲存字串 Horng。_____	
2	C 語言如何輸出變數 a 的字串 Horng。_____	
3	程式填空。以下程式可以輸出 Horng <pre> int main(){ char a[]="Horng"; char b[10]; _____ printf("%s\n",b); } </pre>	
4	C 語言如何以陣列變數 a 儲存 aa、bb、cc 等 3 筆資料。(假設資料長度不超過 10) _____	
5	<pre> int main(){ char a[][10]={"aa","bb","cc"}; char b[10]; printf("%s\n",a); printf("%s\n",a[0]); strcpy(b,a); printf("%s\n",b); } </pre>	

6	程式填空。以下程式可輸入字串，輸入 0 時，輸出所有輸入。 <pre> int main() { char a[10][10], b[10]; int i=0; scanf("%s", &b); _____{ strcpy(a[i], b); i++; scanf("%s", &b); } int n=i; for (i=0; i<=n; i++) printf("%10s", a[i]); } </pre>	
7	<pre> int main() { //字串排序 char a[10][10]={"cc", "ff", "dd", "kk"}; int n=3; char t[10]; for(int i=0; i<=n-1-1; i++){ for (int j=0; j<=n-i-1-1; j++){ if (strcmp(a[j], a[j+1])>0) { //字串比較，請看11_3 strcpy(t, a[j]); //字串複製，請看11_3 strcpy(a[j], a[j+1]); strcpy(a[j+1], t); } } } for (int i=0; i<=n; i++) printf("%s\n", a[i]); } </pre>	
8	<pre> int main() { char a[]="ABCD"; char b[][10]={"AA1", "BB2", "CC3", "DD4"}; char c[][2][10]={{ "AA", "90"}, {"BB", "80"}, {"CC", "70"} }; cout <<a[0]<<endl; //A cout <<a<<endl; //ABCD cout <<b[0][0]<<endl; //A cout <<b[0]<<endl; //AA1 cout <<c[0][0][0]<<endl; //A cout<<c[0][0]<<endl; //AA cout<<c[0][1]<<endl; //90 } </pre>	

9	程式填空，以下程式可輸入以下資料且輸出資料。 <pre> aa 80 70 bb 90 60 cc 30 99 int main(){ char a[3][4][10]; for (int i=0;i<=2;i++){ _____ } //輸出 for (int i=0;i<=2;i++){ printf("%4s %4s %4s",a[i][0],a[i][1],a[i][2]); printf("\n");; } }</pre>	
10	<pre> int main(){//此題也可以程式填空 char a[][5][10]={{"Tom","50","40","",""}, {"John","60","90","",""}, {"Mary","40","60","",""}, {"aa","30","40","",""}, {"dd","90","55","",""}, {"","","","","",""}, {"","","","","",""}}; int stunum=5,i,j,sum=0,avg; for (i=0;i<=stunum;i++){ sum=0; int n=0; for (j=1;j<=2;j++){ sum=sum+atoi(a[i][j]);//轉數值，請看11_3 if (atoi(a[i][j])<60) n++; } avg=sum/2; itoa(avg,a[i][3],10);//轉字串，請看11_3 itoa(n,a[i][4],10); } for (int i=1;i<=3;i++){ sum=0; int n=0; for (int j=0;j<=stunum-1;j++){ sum=sum+atoi(a[j][i]); } } }</pre>	

	<pre> if (atoi(a[j][i])<60) n++; } sum=sum/stunum; itoa(sum,a[stunum][i],10); itoa(n,a[stunum+1][i],10); } printf("\n"); //输出 for (i=0;i<=6;i++){ for (j=0;j<=4;j++) printf("%4s",a[i][j]); printf("\n"); } return 0; }</pre> <table><tr><td>Tom</td><td>50</td><td>40</td><td>45</td><td>2</td></tr><tr><td>John</td><td>60</td><td>90</td><td>75</td><td>0</td></tr><tr><td>Mary</td><td>40</td><td>60</td><td>50</td><td>1</td></tr><tr><td>aa</td><td>30</td><td>40</td><td>35</td><td>2</td></tr><tr><td>dd</td><td>90</td><td>55</td><td>72</td><td>1</td></tr><tr><td></td><td>54</td><td>57</td><td>55</td><td>2</td></tr><tr><td></td><td>3</td><td>3</td><td>3</td><td></td></tr></table>	Tom	50	40	45	2	John	60	90	75	0	Mary	40	60	50	1	aa	30	40	35	2	dd	90	55	72	1		54	57	55	2		3	3	3		
Tom	50	40	45	2																																	
John	60	90	75	0																																	
Mary	40	60	50	1																																	
aa	30	40	35	2																																	
dd	90	55	72	1																																	
	54	57	55	2																																	
	3	3	3																																		
11	<pre>int main(){ char a[]="ABCD"; char b[][10]={"AA1","BB2","CC3","DD4"}; char c[][2][10]={{ "AA","90"}, {"BB","80"}, {"CC","70"} }; char d[10]; strcpy(d,a); printf("%c\n",a[0]); //A printf("%s\n",a); //ABCD printf("%s\n",d); //ABCD printf("%c\n",b[0][0]); //A printf("%s\n",b[0]); //AA1 strcpy(d,b[1]); printf("%s\n",d); //BB2 printf("%c\n",c[0][0][0]); //A printf("%s\n",c[0][0]); //AA strcpy(d,c[0][1]); printf("%s\n",d); //90 }</pre>																																				

```

12  int main() {
    char a[][5][3]={{"aa","80","90","70",""},
                    {"bb","20","10","80",""},
                    {"cc","90","80","70",""}}; //均視為字串

    //計算總分、平均
    int sum=0,avg;
    int i,j;
    for (i=0;i<=2;i++){
        sum=0;
        for (j=1;j<=3;j++){
            sum=sum+atoi(a[i][j]); //轉數值再計算
        }
        avg=sum/3;
        itoa(avg,a[i][4],10); //計算完再轉字串
    }
    //輸出
    for (i=0;i<=2;i++){
        for (j=0;j<=4;j++){
            printf("%8s",a[i][j]);
            //佔8格，且靠右
        }
        printf("\n");
    }
    return 0;
}

```

9-4 結構

範例9-4a

前面的二維陣列僅可以所有欄位的資料型態通通相同，但是若資料如下，請留意姓名是字串，成績是數值。

座號	姓名	國文	英文	數學	平均	備註
1	aa	80	90	70		
2	bb	20	10	80		
3	cc	90	80	70		
平均						

1. 請用二維陣列儲存以上資料。
2. 計算每人平均。
3. 統計每人不及格科數。
4. 統計各科平均。

執行結果

aa	80	90	70	80
bb	20	10	80	36
cc	90	80	70	80

演算說明

以上資料有數值也有字串，因為陣列的所有欄位的資料型態要求一致，也就是僅能全部是數值，或全部是字串，又因為字串無法以數值表示，所以只能通通以字串儲存。

```
char a[][5][3]={{"aa","80","90","70",""},
                 {"bb","20","10","80",""},
                 {"cc","90","80","70",""}}; //均視為字串
```

但是成績的計算就要先將字串的成績轉為數值，等計算完再轉字串，最後再放回，全部程式如下：

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h> //use strcpy()
#include <stdlib.h> //use atoi()
int main() {
    char a[][5][3]={{"aa","80","90","70",""},
                    {"bb","20","10","80",""},
                    {"cc","90","80","70",""}}; //均視為字串

    //計算總分、平均
    int sum=0,avg;
    int i,j;
    for (i=0;i<=2;i++){
        sum=0;
        for (j=1;j<=3;j++){
            sum=sum+atoi(a[i][j]); //轉數值再計算
        }
    }
}
```

```

        avg=sum/3;
        itoa(avg,a[i][4],10); //計算完再轉字串
    }
    //輸出
    for (i=0;i<=2;i++){
        for (j=0;j<=4;j++){
            printf("%8s",a[i][j]);
            //佔8格，且靠右
        }
        printf("\n");
    }
    return 0;
}

```

☞ 結構(struct)

前一範例通通以字串表示，運算時要轉來轉去，真的很麻煩。另一種變通的辦法，就是使用本節介紹的資料結構 **struct**，本書翻譯為結構。因為一個結構容許包含不同型態的資料，也就是容許包含字串與數值，這樣就解決上例使用陣列的不便，請看以下說明。

☞ 結構型態的宣告

結構型態宣告的語法如下：

```

struct 結構型態名稱 {
    資料型態    欄位名稱1;
    資料型態    欄位名稱2;
    :
    資料型態    欄位名稱N;
};

```

以上語法說明如下：

1. 當宣告結構型態時，必須使用 **struct** 關鍵字。
2. 結構型態名稱的命名規則同識別字命名規則。
3. 在大括弧中的項目是結構的成員，也稱為欄位。欄位的宣告方式同一般變數的宣告方式，必須先宣告資料型態，然後緊跟著欄位名稱。
4. 不同的欄位之間必須以分號（；）隔開。

5. 欄位的資料型態可為前面基本資料型態的任一資料型態。例如，char、int、long、double、陣列等型態。
6. 欄位名稱的命名規則同識別字命名規則。

以下以一個學生的成績記錄為例，來說明以上結構型態的宣告。例如：以下的程式片段可以宣告一個含有姓名、國文、數學及英文成績等欄位的結構型態，而結構型態的名稱為 stuGrade。

```
struct stuGrade
{
    char name[5]; // C 語言字串
    int chi;
    int math;
    int eng;
}; // 請留意要分號
```

☞ 結構型態變數的宣告

結構型態一經宣告後，即為一種使用者自訂的資料型態，其用法同基本資料型態。例如：以下敘述宣告變數 student 為使用者自訂的 stuGrade 結構型態。

```
struct stuGrade student;
```

☞ 結構欄位的存取

結構欄位存取的方法是在結構變數與欄位之間加上點（.）運算子。例如：以下的敘述可設定變數 student 的欄位內容。

```
strcpy(student.name, "洪峻維"); // student.name="洪峻維"; 不行
student.chi=90;
student.math=100;
student.eng=80;
```

☞ 結構欄位值的初始化

如同陣列一般，也可以在宣告結構變數時，同時指派結構初始值。例如：以下敘述可設定變數 student 的初始值。

```
struct stuGrade student={"aa", // name 欄位
                          90, // chi 欄位
```

```
100, // math 欄位  
80 /* eng 欄位 */  
}; //請留意，要分號
```

以上說明的全部程式如下：

```
#include <stdio.h>  
#include <stdlib.h>  
#include <string.h>  
struct stuGrade  
{  
    char name[7]; // C 語言字串  
    int chi;  
    int math;  
    int eng;  
};  
int main()  
{  
    //struct stuGrade student;  
    struct stuGrade student={"aa", // name 欄位  
                               80, // chi 欄位  
                               90, // math 欄位  
                               70 /* eng 欄位 */  
                               }; //請留意，要分號  
  
    printf("%5s", student.name);  
    printf("%4d", student.chi);  
    printf("%4d", student.math);  
    printf("%4d", student.eng);  
    printf("\n");  
    strcpy(student.name, "bb");  
    student.chi=90;  
    student.math=100;  
    student.eng=80;  
    printf("%5s", student.name);  
    printf("%4d", student.chi);  
    printf("%4d", student.math);  
    printf("%4d", student.eng);  
    return 0;  
}
```

以上程式執行結果如下：

```
aa 80 90 70
bb 90 100 80
```

以下以結構重做矩形面積。

```
#include <stdio.h>
#include <stdlib.h>
struct rect{//步驟1：定義結構
    int Length;
    int Width;
    int Area;
};
int main(){
    struct rect a; //步驟2：宣告結構變數
    a.Length=3; //步驟3：使用結構欄位寫程式
    a.Width=4;
    a.Area=a.Length*a.Width;
    printf("%d",a.Area);
}
```

☞ 陣列型態與結構型態綜合使用

於前面陣列，曾提到陣列型態可為任一資料型態，當然也包含了使用者自訂的結構型態。例如：以下的敘述即宣告陣列變數 a 為 stuGrade 的結構型態。

```
struct studGrade a[10];
```

當完成以上宣告，變數 a 即擁有 10 筆 stuGrade 型態的陣列資料。如此，便可以設定一號或二號同學的成績紀錄如下：

```
strcpy(a[0].name," 洪竣維"); // 一號的姓名
a[0].chi=90; // 一號的國文成績
strcpy(a[1].name," 陳惠敏");// 二號的姓名
a[1].chi=80; // 二號的國文成績
```

自我練習

題號	題目	結果
1	<pre> struct point{//以結構重做兩點的距離 int x; int y; }; int main(){ struct point a={0,3}; struct point b={4,0}; printf("%2d\n",a.x); int d=sqrt((pow(a.x-b.x,2)+pow((a.y-b.y),2))); printf("%2d",d); } </pre>	
2	程式填空。以下程式可以計算長方形面積與邊長。 <pre> struct rect{ int width; int length; int area; int edge; }; int main(){ struct rect a; a.width=3; a.length=4; _____ _____ printf("%2d\n",a.area); printf("%2d\n",a. edge); } </pre>	
3	程式填空：以下程式輸出如下： <pre> aa 70 80 90 80 bb 90 80 70 80 cc 40 50 60 50 </pre> <pre> struct stuGrade{ char name[7]; int chi; int math; int eng; int avg; }; int main() { </pre>	

```

struct stuGrade a[3]={{"aa", 70,80,90,0},
                      {"bb", 90,80,70,0},
                      {"cc",40,50,60,0}};

for (int i=0;i<=2;i++ )

    for (int i=0;i<=2;i++){
        printf("%5s",a[i].name);
        printf("%4d",a[i].chi);
        printf("%4d",a[i].math);
        printf("%4d",a[i].eng);
        printf("%4d\n",a[i].avg);
    }
}

```

範例9-4b

請以結構重做範例 9-4a。

執行結果

```

aa 70 80 90 80
bb 90 80 70 80
cc 40 50 60 50

```

程式列印

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
struct stuGrade
{
    char name[7]; // C 語言字串
    int chi;
    int math;
    int eng;
    int avg;
};
int main()
{
    //struct stuGrade a;
    struct stuGrade a[3]={{"aa", 70,80,90,0},
                          {"bb", 90,80,70,0},

```

```

                                {"cc",40,50,60,0}};
for (int i=0;i<=2;i++){
    a[i].avg=(a[i].chi+a[i].eng+a[i].math)/3;
}
for (int i=0;i<=2;i++){
    printf("%5s",a[i].name);
    printf("%4d",a[i].chi);
    printf("%4d",a[i].math);
    printf("%4d",a[i].eng);
    printf("%4d\n",a[i].avg);
}
return 0;
}

```

☞ 欄位陣列

前面所介紹的是將陣列變數宣告為結構型態，那結構欄位的資料型態是否可為陣列呢？答案是肯定的。例如：先前 stuGrade 結構型態的 chi、math 及 eng 欄位均為 int，所以 stuGrade 亦可重新定義如下：

```

struct stuGrade
{
    char name[7];
    int score[4];
};

```

其次，宣告陣列變數 a 為 stuGrade 的結構型態如下：

```

struct stuGrade a[10];

```

則此型態的第一個人的成績，可表示如下：

```

strcpy(a[0].name," 洪峻維");// / 一號的姓名
a[0].score[0]=90; // 一號的第一科成績
a[0].score[1]=70; // 一號的第二科成績
a[0].score[2]=80; // 一號的第三科成績

```


自我練習

題號	題目	結果
1	<pre> struct stuGrade { char name[7]; // C 語言字串 int score[5]; char memo[5]; }; int main(){ struct stuGrade a[4]={{"aa", 30,80,90,0,0,""}, {"bb",90,80,70,0,0,""}, {"cc",30,80,30,0,0,""}, {" ",0,0,0,0,0,0,""}; for (int i=0;i<=2;i++){ for (int j=0;j<=2;j++){ a[i].score[3]=a[i].score[3]+a[i]. score[j]; if (a[i].score[j]<60) a[i].score[4]++; } a[i].score[3]=a[i].score[3]/3; if (a[i].score[3]<60) strcpy(a[i].memo,"Fail"); } for (int i=0;i<=3;i++){ for (int j=0;j<=2;j++){ a[3].score[i]=a[3].score[i]+a[j]. score[i]; a[3].score[i]/=3; } for (int i=0;i<=3;i++){ printf("%5s",a[i].name); for (int j=0;j<=4;j++) printf("%4d",a[i].score[j]); printf("%5s",a[i].memo); printf("\n"); } } } </pre>	

2	關於 C 語言中結構 (structures) 資料型態的敘述，下列何者錯誤？ (統測 111) (A) 結構是一種自行定義的資料型態 (B) 宣告結構資料型態的變數時，只能使用已定義的結構名稱宣告 (C) 定義結構只能使用不同資料型態的變數 (陣列僅能使用相同資料型態，結構可以) (D) 定義結構只能使用 struct 關鍵字	
3	關於 C 程式語言中，結構 (structures) 資料型態的敘述，下列何者錯誤？ (技藝競賽修護組 111) (A) 結構是一種自行定義的資料型態 (B) 定義結構只能使用相同資料型態的變數 (C) 宣告結構資料型態的變數時，只能使用已定義的結構名稱宣告 (D) 定義結構只能使用 struct 關鍵字。	

題號	題目	結果
1	<pre> int main() { // 氣泡排序法 int a[]={0, 99, 88, 77, 66, 55}; int i, j, t; const int N=5; for(i=1; i<=N-1; i++) for(j=1; j<=N-i; j++) if(a[j]>a[j+1]) { t=a[j]; // 交換 a[j]=a[j+1]; a[j+1]=t; } for(i=1; i<=N; i++) printf("%5d", a[i]); } </pre>	
2	<pre> int main() { // 比較兩次 int a[]={0, 99, 88, 77, 66, 55}; int i, j, t; const int N=5; for(i=1; i<=2; i++) for(j=1; j<=N-i; j++) if(a[j]>a[j+1]) { t=a[j]; // 交換 a[j]=a[j+1]; a[j+1]=t; } for(i=1; i<=N; i++) printf("%5d", a[i]); } </pre>	

3	<pre> int main() { //氣泡排序，索引從0 int a[]={ 99, 88, 77, 66}; int i, j, t;const int N=4; for(i=0; i<=N-1-1; i++)//-1 for(j=0; j<=N-i-2; j++)//-2 if(a[j]>a[j+1]) { t=a[j];//交換 a[j]=a[j+1]; a[j+1]=t; } for(i=0; i<=N-1; i++)//-1 printf("%5d" ,a[i]); } </pre>	
4	<pre> int main() { int a[]={0, 99, 88, 77, 66}; int i, j, t;const int N=4; int max,maxid; for(i=1; i<=N-1; i++){ max=a[1];maxid=1; for(j=2; j<=N-i+1; j++) if(a[j]>max){ max=a[j]; maxid=j; } t=a[N-i+1]; a[N-i+1]=a[maxid]; a[maxid]=t; } for(i=1; i<=N; i++) printf("%5d" ,a[i]); } </pre>	
5	<pre> int main() { //兩次 int a[]={0, 99, 88, 77, 66}; int i, j, t;const int N=4; int max,maxid; for(i=1; i<=2; i++){ max=a[1];maxid=1; for(j=2; j<=N-i+1; j++) if(a[j]>max){ </pre>	

	<pre> max=a[j]; maxid=j; } t=a[N-i+1]; a[N-i+1]=a[maxid]; a[maxid]=t; } for(i=1; i<=N; i++) printf("%5d" ,a[i]); } </pre>	
6	<pre> struct stuGrade{//2 char name[7]; // C 語言字串 int chi; int math; int avg; }; int main() { //struct stuGrade a; struct stuGrade a[5]={{ "aa", 8,10,18}, {"bb",9,9,18}, {"cc",7,10,17}, {"dd",10,9,19}, {"ee",9,8,17}}; int N=5; struct stuGrade t; for (int i=0;i<=N-1-1;i++){ for (int j=0;j<=N-i-2;j++){ if (a[j].avg >a[j+1].avg){ t=a[j]; a[j]=a[j+1]; a[j+1]=t; } } } for (int i=0;i<=N-1;i++){ printf("%5s",a[i].name); printf("%4d",a[i].chi); printf("%4d",a[i].math); printf("%4d\n",a[i].avg); } } </pre>	

7	<pre> int main() { //計數排序法 int a[]={0, 77, 66, 99, 44, 55}; int b[6], t; const int N=5; for(int i=1; i<=N; i++) { printf("%5d",a[i]); b[i]=1; //設定每個都是第一名 } printf("\n"); for(int i=1; i<=N-1; i++) for(int j=i+1; j<=N; j++) if(a[i]>a[j]) b[j]++; else b[i]++; for(int i=1; i<=N; i++) printf("%5d",b[i]); } </pre>	
8	<pre> int main() { //計數排序，但2次 int a[]={0, 77, 66, 99, 44, 55}; int b[6], t; const int N=5; for(int i=1; i<=N; i++) { printf("%5d",a[i]); b[i]=1; //設定每個都是第一名 } printf("\n"); for(int i=1; i<=2; i++) for(int j=i+1; j<=N; j++) if(a[i]>a[j]) b[j]++; else b[i]++; for(int i=1; i<=N; i++) printf("%5d",b[i]); } </pre>	
9	<pre> int main() { //成績計算 BY陣列 int a[][5]={0,0,0,0,0, {1,10,10,20,1}, {2,9,9,18,1}, {3,9,10,19,1}, </pre>	

	<pre> {4,10,9,19,1}, {5,9,8,17,1}}; const int N=5; for(int i=1;i<=N;i++) a[i][4]=1;//大家都是第一名 for(int i=1;i<=N-1;i++){ for (int j=i+1;j<=N;j++){ if(a[i][3]>a[j][3]) a[j][4]++;//分數少加一 else if (a[i][3]<a[j][3]) a[i][4]++; //分數相等，同名次 } } for (int i=1;i<=N;i++){ for(int j=0;j<=4;j++) printf("%3d",a[i][j]); printf("\n"); } </pre>	
10	<pre> struct stuGrade{//成績計算BY結構 int no; char name[3]; // C 語言字串 int chi; int math; int total; int order; }; int main() { const int N=5; struct stuGrade a[6]={0,"",0,0,0,0}, {1,"aa", 10,10,20,0}, {2,"bb", 9,9,18,0}, {3,"cc", 7,10,17,0}, {4,"dd",10,9,19,0}, {5,"ee",9,7,16,0}}; for (int i=1;i<=N;i++) a[i].order=1;//設定大家都是第1名 for(int i=1; i<=N-1; i++) for(int j=i+1; j<=N; j++) if(a[i].total>a[j].total) a[j].order++;//分數少的名次加1 else a[i].order++; for(int i=1; i<=N; i++){ printf("%3d %4s %3d %3d %3d %3d",a[i]. no,a[i].name,a[i].chi, a[i].math,a[i].total,a[i].order); printf("\n"); } } </pre>	

🔍 搜尋 (Search)

我們日常生活總是不斷的搜尋，在家裡找東西、到圖書館找書、到超商買東西，都是在搜尋。本單元將介紹若干資料搜尋的方法，分別是循序搜尋與二分搜尋法，分別說明如下：

🔍 循序搜尋法 (Sequential Search)

於序列中，從頭到尾逐一比對搜尋的方式，稱為循序搜尋法，程式如下，其次，若是資料沒有按照某一規則存放，那就只能靠循序搜尋法。

👉 自我練習

題號	題目	結果
1	<pre> int main() { int N=5; int a[]={0,1,2,3,4,5}; int b=3; for (int i=1;i<=N;i++) if (b==a[i]){ printf("%d",i); break; } } </pre>	
2	<pre> #include <string.h> //use strcmp() int main() { //9-5e by array char a[13][2][8] = {{"0", "0"}, {"aa", "1111168"}, {"hh", "2222168"}, {"cc", "3333168"}, {"gg", "4444168"}, {"ff", "5555168"}, {"ii", "6666168"}, {"ee", "7777168"}, {"bb", "8888168"}, {"jj", "9999168"}, {"dd", "1688168"}, {"kk", "2688168"}, {"ll", "3688168"}}; //3688168共7個字元，加上字串結束字元，共8個才夠 int i, found=0; char name[3]; </pre>	

	<pre> printf("input a name: "); scanf("%s", &name);//& for(i=1; i<=12; i++) { if(strcmp(a[i][0],name)==0) { found=1; break; } } if(found==1) printf(" the phone number is %s", a[i][1]); else printf("no data found"); return 0; } </pre>	
3	<pre> struct customer { //9-5e by 結構 char name[3]; char tel[8]; }; int main() { struct customer a[13]={{"0", "0"}, {"aa", "1111168"}, {"hh", "2222168"}, {"cc", "3333168"}, {"gg", "4444168"}, {"ff", "5555168"}, {"ii", "6666168"}, {"ee", "7777168"}, {"bb", "8888168"}, {"jj", "9999168"}, {"dd", "1688168"}, {"kk", "2688168"}, {"ll", "3688168"}}; int i, found=0; char name[3]="ff"; while (strcmp(name,"end")!=0){ for(i=1; i<=12; i++) { if(strcmp(a[i].name,name)==0) { found=1; break; } } if(found==1) printf("%s\n", a[i].tel); else printf("no data found\n"); printf("input a name: "); scanf("%s", &name);//& } } </pre>	

☞ 二分搜尋法 (Binary Search)

前面的循序搜尋法，是採用逐一訪視的方式搜尋所要的資料，但當資料量龐大時，就顯得毫無效率。舉例而言，資料數量若有 1000 筆，則採用循序搜尋法平均搜尋次數為 500 次，但若採用本節所討論的二分搜尋法，則至多不會超過 10 次 ($2^{10}=1024$)。使用二分搜尋法前應先將資料排序。在一個已排序的序列中，首先與序列的中間值比較，若待搜尋資料比中間值大，則待搜尋資料必在此中間值的後面，反之則在前面，所以二分搜尋法每次可將搜尋範圍減半，如此對於資料量龐大時，可提高搜尋效率，請看以下程式：

👉 自我練習

題號	題目	結果
1	一組 129 個已排序的數值資料，若用二元搜尋法找其中某一個特定值，至多需要比對幾次才可找到？ (技藝競賽程式組 111) (A) 7 (B) 8 (C) 9 (D) 10	
2	<pre> int main() { //二分搜尋，寫出搜尋過程 int a[]={0,1,2,3,4,5}; int b=1; int l=1; //下邊界 int u=5; //上邊界 int m; int found=0; while((l<=u) && (found==0)) { m=(int)((l+u)/2); printf("%d %d %d\n",l,u,m); if(b==a[m]){ found=m; break; } else { if(b>a[m]) l=m+1; //調整下邊界 else u=m-1; //調整上邊界 } } if(found==0) printf("no data found"); else printf("%d", found); } </pre>	

3	<pre>int bsearch(int b[], int Key, int low, int high) { int middle; while (low <= high) { middle = (low + high) / 2; if (Key == b[middle]) return middle; else if (Key < b[middle]) high = middle - 1; else low = middle + 1; } return -1; }</pre> 1. 給予一個整數陣列 <code>int n[10]={3,5,7,9,11,14,17,33,44,50}</code>; 呼叫右圖的函式 <code>bsearch(n, 44, 0, 9)</code> 來搜尋 44，請問 while 迴圈內的指令 <code>middle = (low+high)/2</code>; 會執行幾次？ (A) 5 (B) 4 (C) 3 (D) 2 2. 同上題，呼叫函式 <code>bsearch(n, 13, 0, 9)</code> 以搜尋 key 13，while 迴圈內的指令 <code>middle = (low+high)/2</code>; 會執行幾次？ (A) 2 (B) 3 (C) 4 (D) 5 3. 同上題，呼叫函式 <code>bsearch(n, 55, 0, 9)</code> 以搜尋 key 55，while 迴圈內的指令 <code>middle = (low+high)/2</code>; 會執行幾次？ (技藝競賽程式組 111，107) (A) 2 (B) 3 (C) 4 (D) 5	
4	//程式填空，二分搜尋by array <pre>int main() { char a[13][2][8]={{"0", "0"}, {"aa", "1111168"}, {"hh", "2222168"}, {"cc", "3333168"}, {"gg", "4444168"}, {"ff", "5555168"}, {"ii", "6666168"}, {"ee", "7777168"}, {"bb", "8888168"}, {"jj", "9999168"}, {"dd", "1688168"}, {"kk", "2688168"}, {"ll", "3688168"}};</pre>	

```

int i,j, found=0;
char t[9];
const int N=12;
for(i=1; i<=N-1; i++) {
    for(j=1; j<=N-i; j++) {
        if(strcmp(a[j][0], a[j+1][0])>0) {
            strcpy(t,a[j][0]);
            strcpy(a[j][0],a[j+1][0]);
            strcpy(a[j+1][0],t);
            strcpy(t,a[j][1]);
            strcpy(a[j][1],a[j+1][1]);
            strcpy(a[j+1][1],t);
        }
    }
}
for(i=1; i<=12; i++) {
    for(j=0; j<=1; j++) {
        printf("%9s",a[i][j]);
    }
    printf("\n");
}
char name[3];
printf("input a name: ");
scanf("%s", &name); //&
int l=1; //下邊界
int u=12; //上邊界
int m;
while((l<=u) & (found==0)) {
    m=(int)((l+u)/2);
    if(strcmp(name, a[m][0])>0) { // #include <string.h>
        found=1;
        break;
    }
    else {
        if(strcmp(name, a[m][0])>0)
            l=m+1; //調整下邊界
        else
            u=m-1; //調整上邊界
    }
}
if(found==1)
    printf(" the phone number is %s", a[m][1]);
else
    printf("no data found");
return 0;
}

```

```

5 //程式填空，二分搜尋，使用結構
struct customer {
    char name[3];
    char tel[8];
};
int main() {
    struct customer a[13]={{"0", "0"}, {"aa",
"1111168"}, {"hh", "2222168"},
{"cc", "3333168"}, {"gg", "4444168"}, {"ff",
"5555168"}, {"ii", "6666168"},
{"ee", "7777168"}, {"bb", "8888168"}, {"jj",
"9999168"}, {"dd", "1688168"},
{"kk", "2688168"}, {"ll", "3688168"}};
    int i,j, found=0;
    struct customer t;
    const int N=12;
    for(i=1; i<=N-1; i++) {
        for(j=1; j<=N-i; j++) {
            if(strcmp(a[j].name, a[j+1].name)>0) {
                //a[j] a[j+1] 交換
                t=a[j];
                a[j]=a[j+1];
                a[j+1]=t;
            }
        }
    }
    for(i=1; i<=12; i++){
        printf("%4s %9s",a[i].name,a[i].tel);
        printf("\n");
    }
    char name[3];
    printf("input a name: ");
    scanf("%s", &name);//&
    int l=1;//下邊界
    int u=12;//上邊界
    int m;
    while((l<=u) & (found==0)) {
        m=(int)((l+u)/2);
        if(strcmp(name,_____)==0)
    }
    {//#include<string.h>

```

	<pre> found=1; break; } else { if(strcmp(name, a[m].name)>0) l=m+1;//調整下邊界 else u=m-1;//調整上邊界 } } if(found==1) printf(" the phone number is %s", a[m].tel); else printf("no data found"); return 0; } </pre>	
6	程式填空：以下程式可以演奏小毛驢。 <pre> int main(){//小毛驢，進階練習，陣列裡面有陣列 const int b[]={0,523,587,659,698,784,880,988,1046}; const int s=80; const int l=26; const int a[]={1,1,1,1,1,1,3,1,5,1,5,1,5,1,5,1,6, 1,6,1,6,1,8,1,5,4};//音階，時間 while(1){ for (int i=0;i<l/2;i=i+1){ _beep(____,____*60000/(s*2));//音階，延遲時間 } } } </pre>	
7	程式填空：以下程式可以演奏小毛驢。 <pre> int main(){//小毛驢，進階練習，陣列裡面有陣列 const int b[]={0,523,587,659,698,784,880,988,1046}; const int s=80; const int l=13; const int a[][2]={{1,1},{1,1},{1,1},{3,1},{5,1}, {5,1},{5,1},{5,1},{6,1},{6,1}, {6,1},{8,1},{5,4}}; while(1){ for (int i=0;i<=l-1;i=i+1){ _beep(____,____*60000/(s*2));//音階，延遲時間 } } } </pre>	

8	程式填空：以下程式可以演奏小毛驢。 <pre> int main(){//小毛驢，進階練習，陣列裡面有陣列 const int b[]={0,523,587,659,698,784,880,988,1046}; const int s=80; const int l=13; struct aa a[]={1,1,"我",{1,1,"有"},{1,1,"一"}, 3,1,"隻",{5,1,"小"},{5,1,"毛"}, 5,1,"驢",{5,1,"我"},{6,1,"從"}, 6,1,"來",{6,1,"也"},{8,1,"不"}, 5,4,"騎"}}; while(1){ for (int i=0;i<=l-1;i=i+1){ printf("%s",a[i].c); _beep(____,____*60000/(s*2)); //音階,延遲時間 } } } </pre>	
9	程式填空，英文單字記憶練習，填充題 <pre> #include <string>//use strcmp() int main() { //英文單字記憶練習，填充題 char b[100];int c=0,d=0; //將題目用陣列儲存，等待學習檔案後，可將這些資料放在檔案 char a[14][3][100]={{"","",""}, {"delighted", "adj", "高興的"}, {"gear", "n", "服裝"}, {"behave", "v", "表現"}, {"attract", "v", "吸引"}, {"criticism", "n", "批評"}, {"concerned", "adj", "擔心的"}, {"There are four people in my family.", " ", "我們家有四個人"}, {"good", "adj", "好的"}, {"apple", "n", "蘋果"}, {"look", "v", "看見"}, {"book", "n", "書"}, {"television", "n", "電視"}, {"cellphone", "n", "手機"}}; int i,j=0; char e; for (i=1;i<=10;i++){ printf("%s,%s :",____,____); //輸出詞性,中文 j=0; } } </pre>	

	<pre> while (_____) { //可輸入含空白字串 b[j]=e; j++; } b[j]='\0';//記得自己要放字串結尾 if (_____)==0){//與題目比對 printf("Right\n"); c++; } else{ printf("Wrong\n"); d++; } } printf("The Right number is :%d\n",c); printf("The wrong number is :%d\n",d); return 0; } </pre>	
10	程式填空，選擇提練習 <pre> int main(){//選擇題練習 _____={{ "book", "n", "2", "學校", "書", "老師", "校長", { "delighted", "adj", "1", "高興的", "悲傷的", "生氣的", "憤怒的" }, { "gear", "n", "3", "背包", "褲子", "服裝", "鞋子", { "behave", "v", "2", "生氣", "表現", "難過", "看到", { "attract", "v", "4", "打架", "斥責", "排斥", "吸引", { "criticism", "n", "1", "批評", "讚美", "說明", "美人", { "concerned", "adj", "1", "擔心的", "高興的", "傷心的", "生氣的" } } } } } }; _____ _____ _____ char b[2]; //使用者輸入 //t1=int(time.time()) for (int i=0; i<=n-1; i++){ printf("%d . %s,%s : " , __, __, __); // #輸出題號, 英文單字, 詞性 for (int j=1; j<=4; j++){ printf("(%d) %s " , j, _____); //輸出四個選項 } } } </pre>	

	<pre> printf(":"); scanf("%s",&b); //等待使用者輸入答案 if (_____)==0){ //比對 不能使用b==a[i][2] r=r+1; printf("Right\n"); } else{ printf("Wrong\n"); w=w+1; } //t2=int(time.time()) //t=t2-t1 } printf("The Right number is :%d\n",r); printf("The wrong number is :%d\n",w); //print('total time=%d second'% t) } </pre>	
11	<pre> //寫出執行結果 (進階習題) #include <iostream> using namespace std; int main() { int a[]={0,5,7,8,4,5,6,2,5,9}; int i,j,max,n=9; int b[10]; for (i=1;i<=n;i++) b[i]=0; for (i=1;i<=n;i++) b[a[i]]++; max=b[1]; for (i=2;i<=n;i++) if (b[i]> max) max=i; cout<<max<<endl; for (i=1;i<=n-1;i++) for (j=1;j<=n-i;j++) if (a[j]>a[j+1]){ int t=a[j];a[j]=a[j+1];a[j+1]=t;} if (n%2==1) cout<<a[n/2]<<endl; else </pre>	

	<pre> cout<<(a[n/2]+a[n/2+1])/2; int min; min=max=a[1]; for (i=2 ;i<=n;i++){ if (a[i] >max) max=a[i]; if (a[i]<min) min=a[i]; } cout<<max-min; } </pre>	
12	程式填空：以下程式可計算T分數（進階習題） <pre> #include <iostream> #include <cmath> using namespace std; int main(int argc, char** argv) { int a[]={0,5,7,8,4,5,6,2,5,9}; int i,j,max,n=9; float b[10]; float s=0,avg,v,sd; for (i=1 ;i<=n;i++) s=s+_____; avg=s/n; s=0; for (i=1 ;i<=n;i++) s=s+_____; v=s/n; cout<<"變異數 = "<<v<<endl; sd=sqrt(v); cout<<"標準差 = "<<sd<<endl; cout<<"每人Z分數如下："<<endl; for (i=1 ;i<=n;i++){ b[i]=_____; cout<<i<<" "<<ceil(b[i]*10)/10<<endl;//取小數 一位 } return 0; } </pre>	

```

13  int main(int argc, char *argv[]) { //進階習題
    int a1=1567;
    int a2=835;
    int b1[11],b2[11],b3[11];
    int i;
    for ( i=10 ;i>=0 ;i--) {
        b1[i]=0; //協助清空記憶體
        b2[i]=0 ;
        b3[i]=0;
    }
    int n=1; //索引從1開始,
    int a=a1;
    int r;
    //轉為十進位, 放到陣列索引1,2,3,...
    while(a>0) {
        r=a%10;
        b1[n]=r;
        n++;
        a=a/10;
    }
    for (i=4 ;i>=1 ;i--)
        printf("%1d",b1[i]);
    printf("\n");
    n=1; //索引從1開始
    a=a2;
    //轉為十進位, 放到陣列索引1,2,3,...
    while(a>0) {
        r=a%10;
        b2[n]=r;
        n++;
        a=a/10;
    }
    for (i=4 ;i>=1 ;i--)
        printf("%1d",b2[i]);
    printf("\n");
    int t;
    int c=0; //進位
    for (i=1 ;i<=10;i++){
        t=b1[i]+b2[i]+c; //加上一個進位
        c=t/10; //進位
        b3[i]=t%10; //進位不要
    }
    for (i=4 ;i>=1 ;i--)
        printf("%1d",b3[i]);
    printf("\n");
    return 0;
}

```

```

14  int main(int argc, char *argv[]) { // (進階習題)
    int aa = 625;
    int bb[10];
    int c ;
    int i, j ;
    int a,b;
    int n ;
    int d ;
    int f;
    //將數字從最右邊，分成兩個兩個一組
    i = 0;
    do{
        i = i + 1;
        bb[i] = aa % 100;
        aa = aa / 100;
    }while( aa > 0);
    n = i;
    for(i = n;i>=1;i--){
        printf("%d",bb[i]);
    }
    printf("\n");
    //從9,8,7的平方依序找出小於等於最左邊那一組的數字，本例
    是13，並扣掉
    d = bb[n];
    a = 10;
    do{
        a = a - 1;
        f = a * a;
    }while( f > d);
    d = d - f; //d是每次扣剩的
    printf("a=%d ,d=%d\n",a, d);
    //b依序從9,8,7找，並扣掉b(20a+b)
    //a = i;
    n = n - 1;
    for (i=n;i>=1;i--){
        d = d * 100 + bb[i]; //前面剩的是100倍
        b = 10;
        printf("a=%d\n",a);
        printf("d=%d\n",d);
        do{
            b = b - 1;
            f = b*( 20*a + b) ;
        } while (f>d);
        d = d - f; //d是每次扣剩的
        a = a * 10 + b;
    }
    printf("平方根是:%d ",a);
    printf("d=%d",d);
    return 0;
}

```

```

15  #include <iostream>// (進階習題)
    using namespace std;
    int main(int argc, char** argv) {
        //int aa = 3844;
        int aa = 15625;
        short bb[10], c, i, j, a, b, n, d, f;
        //將數字從最右邊，分成兩個三個一組
        i = 0;
        do{
            i = i + 1;
            bb[i] = aa % 1000;
            aa = aa / 1000;
        }while( aa > 0);
        n = i;
        for(int i = n; i >= 1; i--){
            cout << bb[i] << " ";
            cout << endl;
            //從9,8,7的立方依序找出小於等於最左邊那一組的數字，本例
            //是8，並扣掉
            d = bb[n];
            a = 10;
            do{
                a = a - 1;
                f = a * a * a;
            }while( f > d);
            d = d - f; //d是每次扣剩的
            cout << "a=" << a << "    ,d=" << d;
            cout << endl << endl;
            //b依序從9,8,7找，並扣掉b(300aa+30ab+bb)
            a = i;
            n = n - 1;
            for (int i=n; i >= 1; i--){
                d = d * 1000 + bb[i]; //前面剩的是1000倍
                b = 10;
                cout << "a=" << a << endl;
                cout << "d=" << d << endl;
                cout << endl;
                do{
                    b = b - 1;
                    f = b * ( 300 * a * a + 30 * a * b + b * b ) ;
                } while (f > d);
                d = d - f; //d是每次扣剩的
                a = a * 10 + b;
            }
            cout << endl;
            cout << "立方根是: " << a << endl;
            cout << d << endl;
            return 0;
        }
    }

```

16	<pre> #include <time.h>//本例使用Time()函式// (進階習題) int main(int argc, char *argv[]) { int a,b[100];int found=0; srand(time(0)); b[1]= rand()%100; printf("%d ",b[1]); int i; for (i=2;i<=100;i++){ a=rand()%100;//0~99 printf("%d ",a); found=0; for (int j=1;j<=i-1;j++){ if (a==b[j]) { found=1; break; } if (! found) b[i]=a; } if (found==1) break; } printf("\n"); printf("%d ",i); } </pre> 假設輸出結果如下： 15 7 1 28 60 7 請問接下來輸出的值_____	
17	<pre> int main() { int N=5; int a[]={0,1,2,3,4,5}; int b=3; for (int i=1;i<=N;i++) if (b==a[i]){ printf("%d",i); break; } } </pre>	

課後習題

一、填充題

題號	題目	結果
1	<pre> int main() { //APCS10503 int a[100], b[100]; for (int i=1; i<=100; i=i+1) { b[i] = i; } a[0] = 0; for (int i=1; i<=100; i=i+1) { a[i] = b[i] + a[i-1]; } printf ("%d\n", a[50]-a[30]); return 0; } </pre>	
2	<pre> int main() { //APCS10503 int A[5], B[5], i, c; for (i=1; i<=4; i=i+1) { A[i] = 2 + i*4; B[i] = i*5; } c = 0; for (i=1; i<=4; i=i+1) { if (B[i] > A[i]) { c = c + (B[i] % A[i]); } else { c = 1; } } printf ("%d\n", c); return 0; } </pre>	
3	定義 $a[n]$ 為一陣列 (array)，陣列元素的指標為 0 至 $n-1$。若要將陣列中 $a[0]$ 的元素移到 $a[n-1]$，底下程式片段空白處該填入何運算式？ (A) $n+1$ (B) n (C) $n-1$ (D) $n-2$ <pre> int i, hold, n; for (i=0; i<=____; i=i+1) { hold = a[i]; a[i] = a[i+1]; a[i+1] = hold; } </pre>	

4	<pre> int main() { //APCS10510 int a[9] = {1, 3, 5, 7, 9, 8, 6, 4, 2}; int n=9, tmp; for (int i=0; i<n; i=i+1) { tmp = a[i]; a[i] = a[n-i-1]; a[n-i-1] = tmp; } for (int i=0; i<=n/2; i=i+1) printf ("%d %d ", a[i], a[n-i-1]); return 0; } </pre>	
5	下面哪組資料若依序存入陣列中，將無法直接使用二分搜尋法搜尋資料？ (A) a, e, i, o, u (B) 3, 1, 4, 5, 9 (C) 10000, 0, -10000 (D) 1, 10, 10, 10, 100	
6	<pre> int main() { //APCS10510 int i, sum, arr[10]; for (int i=0; i<10; i=i+1) arr[i] = i; sum = 0; for (int i=1; i<9; i=i+1) sum = sum - arr[i-1] + arr[i] + arr[i+1]; printf ("%d", sum); return 0; } </pre>	
7	<pre> int main() { //APCS10603 //int A[]={9,6,7,3,1,2}; int A[]={9,9,9,9,9,9}; int n=6; int p,q; p=q=A[0]; for (int i=1; i<n; i=i+1) { if (A[i]>p) p=A[i]; if (A[i]<q) q=A[i]; } printf("p=%d,q=%d",p,q); //9,9 return 0; } </pre>	

8	<pre> int main() { int M=2,N=3; int A[2][3]={{0,1,2},{3,4,5}}; for (int i=0; i<M; i=i+1){ int rowsum =0; for (int j=0; j<N; j=j+1) { rowsum = rowsum + A[i][j]; } printf("The sum of row %d is %d\n", i,rowsum); } return 0; } </pre>	
9	若 A[][] 是一個 MxN 的整數陣列，底下程式片段用以計算 A 陣列每一列的總和，以下敘述何者正確？ (A) 第一列總和是正確，但其他列總和不一定正確 (B) 程式片段在執行時會產生錯誤 (run-time error) (C) 程式片段中有語法上的錯誤 (D) 程式片段會完成執行並正確印出每一列的總和 <pre> void main () { int rowsum 0; for (int i=0; i<M; i=i+1) for (int j=0; j<N; j=j+1) { rowsum = rowsum + A[i][j]; printf(The sum of row %d is %d. n "", i, } </pre>	
10	以下程式若輸入依序為整數 0, 1, 2, 3, 4, 5, 6, 7, 8, 9，請問 X[] 陣列的元素值依順序為何？ //APCS10603 <pre> int main() { int X[10] = {0}; for (int i=0 ; i<10; i=i+1) scanf("%d" ,&X[(i+2)%10]); for (int i=0 ; i<10; i=i+1) printf("%d,%d\n",i,X[i]); return 0; } </pre>	



11	若 A[1]、A[2]，和 A[3] 分別為陣列 A[] 的三個元素 (element)，下列那個程式片段可以將 A[1] 和 A[2] 的內容交換？ (A) A[1] = A[2]; A[2] = A[1]; (B) A[3] = A[1]; A[1] = A[2]; A[2] = A[3]; (C) A[2] = A[1]; A[3] = A[2]; A[1] = A[3]; (D) 以上皆可	
12	<pre>int main() { int M = 1, N = 101, s = 3; int A [] = {80, 90, 100}; for (int i=0; i<s; i=i+ 1) { if (A [i]>M) M = A [i]; if (A[i]<N) N = A [i]; } printf("Max = %d, Min= %d \n", M, N); return 0; }</pre>	

Chapter 10

指標

指標型態是 C 語言獨有的資料型態，它的優點是指標型態直接使用記憶體位址存取資料，如此可以提升程式的執行效率。指標型態的應用非常廣，可以協助處理陣列與字串，以提高程式的執行效率。

10-1 指標與單一變數

請鍵入以下程式，寫出執行結果。

```
int main() {  
    int *ptr;  
    int a=5;  
    ptr = &a;  
    printf("%d\n", *ptr); //5  
    printf("%d\n", ptr);  
    return 0;  
}
```

☞ 指標型態的宣告

指標型態的宣告比較特殊，它必須藉助一種指標運算子（*），其宣告語法如下：

資料型態 *指標變數；

資料型態可以是 3-2 節的基本資料型態，指標變數可以是一個合法的變數名稱。例如，以下式子即宣告 ptr 是一個整數的指標變數。

```
int *ptr;
```

10-2 程式設計實習進階

經過以上宣告之後，ptr 即可指向任一整數變數位址，例如：

```
int a=5;
ptr = &a;
```

則 ptr 指向變數 a 的位址，& 是取位址運算符號。往後即可使用 *ptr 代表變數 a。例如，以下敘述可印出 *ptr 的值。

```
printf("%d\n", *ptr); //5
```

但是，若硬要印出 ptr 的值，如以下敘述：

```
printf("%d\n", ptr);
```

則沒有意義了。因為 ptr 的值是記憶體位址，此位址會隨著不同的機器與不同的執行次數而有不同的值。所以，在指標型態中，我們並不在意 ptr 所在位址，在意的是 *ptr 的值。

⌘ 運算

指標指向之值也可以用來運算，但運算結果僅能放在變數。例如：

```
int main() {
    int a=5,b=4,c;
    int *p,*q,*r;
    p=&a;
    q=&b;
    printf("%d\n", *p); //5

    c=*p+*q;
    printf("%d\n", c); //9
    r=&c;
    printf("%d\n", *r); //9
    return 0;
}
```

若將運算結果也放在指標，那就不行。例如，以下就不行。

```
*r=*p+*q;
```

自我練習

題號	題目	結果
1	C 語言是使用下列哪一個運算子來取得變數的記憶體位址？ (技藝競賽修護組 111) (A) 「*」 (B) 「&」 (C) 「&&」 (D) 「**」。	
2	<pre>int main() { int a=5,b=4,c; int *p,*q,*r; p=&a; q=&b; *p=10; c=*p+*q; printf("%d\n",c); }</pre>	
3	<pre>int main(){ int a=5,b=4; int *p=&a; int *q=&b; int c=*p * *q; printf("%d",c); return 0; }</pre>	

10-2 指標與陣列

接著我們要談如何使用指標型態存取陣列的值。例如，請鍵入以下程式，寫出執行結果。

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    int a[5]={3, 8, 6, 2, 5};
    int *ptr;    //務必和所指的陣列相同型態
    ptr=a;      // 或
    //ptr = &a[0];
    printf("%d\n",*ptr); //3
    printf("%d\n",ptr[0]); //3
}
```

```

printf("%d\n",ptr[1]);//8
printf("%d\n",*(ptr+1));//8
ptr++;                // ptr=ptr+1
printf("%d\n",*ptr);//8
printf("%d\n",ptr[0]);//8
printf("%d\n",ptr[1]);//6
printf("%d\n", *(ptr+1));//6
printf("%d\n", ptr);
return 0;
}

```

☞ 指標與一維陣列

```

int  a[5]={3, 8, 6, 2, 5};
int  *ptr;    //務必和所指的陣列相同型態

```

宣告 ptr 是一種指標型態，ptr 本身是一個位址，*ptr 才是一個值，而且 *ptr 的型態務必和所指的陣列相同型態。將指標指向一維陣列的程式如下：

```
ptr=a;
```

或

```
ptr = &a[0];
```

將 ptr 指向 a[0] 的位址，& 符號是取址的意思。所以若經以上設定，以下程式都可以取陣列值。

```

printf("%d\n",*ptr);//3
printf("%d\n",ptr[0]);//3
printf("%d\n",ptr[1]);//8
printf("%d\n",*(ptr+1));//8

```

☞ 指標的移動

若要得下一個陣列值，則應將指標向後移一個位置，例如：

```

ptr++;                // ptr=ptr+1
printf("%d\n",*ptr);//8
printf("%d\n",ptr[0]);//8

```

```
printf("%d\n", ptr[1]); //6
printf("%d\n", *(ptr+1)); //6
```

若您強制要印出 ptr，例如：

```
printf("%d\n", ptr);
```

則是得到一個位址，而且這個結果對使用者來說是沒有意義，因為這個位址每次執行均會不同，而且不同的電腦系統也會有不同的值（位址）。重點是我們要得到指標的值 *ptr，而不是指標本身 ptr。

☞ 指標與資料型態的變化

指標是有智慧的，它會隨著指標值所佔記憶體的大小做出正確的移動，例如：

```
double b[5]={3.3, 9.9, 2.2, 4.4, 6.6};
double *p1; //務必和所指的陣列相同型態
p1=b;
p1++;
printf("%d\n", *p1);
```

亦可得到陣列的第 1 個元素 9.9，但若：

```
int a[5]={3, 8, 6, 2, 5};
double *p1;
p1=a;
```

則會得到錯誤結果，因為 *p1 是指向 8 個 byte 的 double 型態，而 *ptr 是指向 2 個 byte 的 int 型態，所以同樣是指向下一個元素 ptr++ 或 p1++，但移動的記憶體數不同。

題號	題目	結果
1	右圖程式片段，輸出結果何者正確？（技藝競賽程式組 110） (A) 3 (B) 4(C) 2 (D) 1 <pre>int x=3; int a[]={1,2,3,4}; int *z; z=a; z=z+x; cout<<*z<<"\n";</pre>	

2	有一 C 程式片段 (如下圖) 執行的結果為何? (技藝競賽程式組 109) (A):'c' (B):'f' (C):'g' (D):'i' <pre> #include <stdio.h> int main() { char data[3][3] = { {'a', 'b', 'c'}, {'d', 'e', 'f'}, {'g', 'h', 'i'} }; char *i; i = &data[1][1]; i=i+2; printf("%3c\n", *i); } </pre>	
3	有一 C 程式片段 (如下圖) 執行的結果為何? (技藝競賽程式組 109) (A): 2 (B): 4 (C): 7 (D): 9 <pre> #include <stdio.h> int main() { int data[10] = { 8, 3, 5, 2, 4, 7, 6, 1, 9, 10 }; int i, *j, *k; j = &data[1]; j = j + 3; k = j + data[*j] % 3; printf("%3d\n", *k); } </pre>	
4	<pre> int main(){// (技藝競賽程式組108) int dataA[3][3]={1,2,3},{4,5,6},{7,8,9}}; int *i=&dataA[1][1],*j; j=i+3;//往後移三個位置 printf("%d ",*i); return 0; } </pre> (A) 4 (B) 5 (C) 7 (D) 8	
5	<pre> int main(){// (技藝競賽程式組108) int dataA[3][3]={1,2,3},{4,5,6},{7,8,9}}; int *i=&dataA[1][1],*j; j=i+3;//往後移三個位置 printf("%d ",*j); return 0; } </pre> (A) 4 (B) 5 (C) 7 (D) 8	
6	以下程式，若第一個輸出是 6618608，則第 2 個輸出是： (技藝競賽程式組 108) <pre> int main(){ int dataA[3][3]={1,2,3},{4,5,6},{7,8,9}}; int *i=&dataA[1][1],*j=&dataA[2][2]; printf("%d %d",i,j); return 0; } </pre> (A) 6618612 (B) 6618616 (C) 6618624 (D) 6618632	

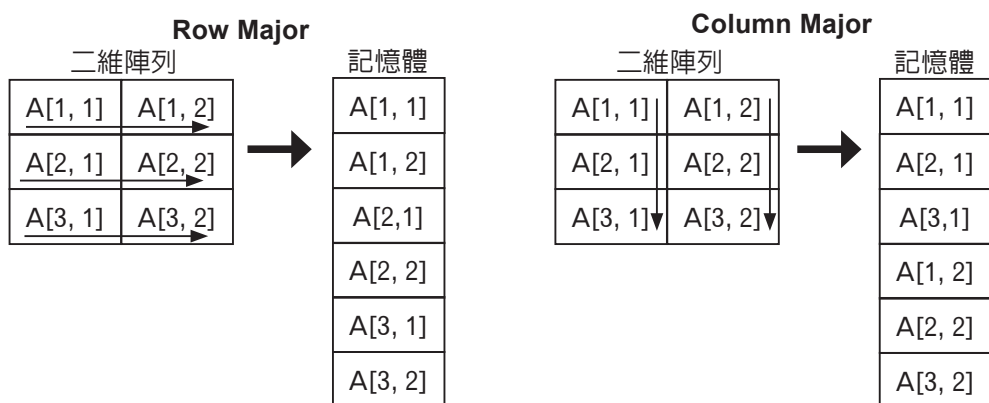
7	<pre> int main(){ int a[]={77, 66, 99, 44, 55}; // 陣列宣告與初始化 int *ptr=a; // 宣告ptr為指標型態，並指向a陣列 int i; int m1=*ptr; // 先設1號為最高分的播臺主 int m2=*ptr; // 先設1號為最低分的播臺主 int s=*ptr; // 先設1號為總分 ptr++; for(i=1; i<=4; i++) { s=*ptr; if(*ptr>m1) // 只要大於播臺主便取代播臺主 m1=*ptr; if(*ptr<m2) m2=*ptr; ptr++; // 指向下一個陣列值 } // 結果輸出 printf("Average: %d\n",s/5); printf("max: %d\n", m1); printf("min: %d\n", m2); return 0; } </pre>	
8	<pre> int main(){ int a[]={-1, 0, 3,4,-5,-8}; // 陣列宣告與初始化 int N=6; int *ptr=a; // 宣告ptr為指標型態，並指向a陣列 int pnum=0; int znum=0; int mnum=0; for(int i=0; i<=N-1; i++) { if(*ptr>0) pnum++; else if (*ptr==0) znum++; else mnum++; ptr++; // 指向下一個陣列值 } // 結果輸出 printf("%d\n",pnum); printf("%d\n", znum); printf("%d\n", mnum); } </pre>	

9	<pre> int main(){//氣泡排序使用指標 int a[]={99, 88, 77, 66, 55}; int *ptr=a; int i, j, t; const int N=5; for(i=1; i<=(N-1); i++) { ptr=a; for(j=1; j<=(N-i); j++) { if(*ptr>*(ptr+1)){ t=*ptr; *ptr=*(ptr+1); *(ptr+1)=t; } ptr++; } } ptr=a; for(i=1; i<=N; i++) { printf("%4d ",*ptr); ptr++; } return 0; } </pre>	
10	<pre> int main(){//僅2次 int a[]={99, 88, 77, 66, 55}; int *ptr=a; int i, j, t; const int N=5; for(i=1; i<=2; i++) { //2 ptr=a; for(j=1; j<=(N-i); j++) { if(*ptr>*(ptr+1)){ t=*ptr; *ptr=*(ptr+1); *(ptr+1)=t; } ptr++; } } ptr=a; for(i=1; i<=N; i++) { printf("%4d ",*ptr); ptr++; } return 0; } </pre>	

11	<pre> int main(){//計數排序法 int a[]={0,99, 88, 77, 66, 55}; int b[]={0,1,1,1,1,1,1}; int i, j, t;const int N=5; int *ptr a=a; int *ptr b=b; for(i=1; i<=(N-1); i++) for(j=i+1; j<=N; j++){ int c=*(ptr a+i); if(*(ptr a+i)>*(ptr a+j)) *(ptr b+j)=*(ptr b+j)+1; else *(ptr b+i)=*(ptr b+j)+1; } for(i=1; i<=N; i++) { printf("%4d ",*(ptr b+i)); } return 0; } </pre>	
12	<pre> int main(){//計數排序法,僅2次 int a[]={0,99, 88, 77, 66, 55}; int b[]={0,1,1,1,1,1,1}; int i, j, t;const int N=5; int *ptr a=a; int *ptr b=b; for(i=1; i<=2; i++)//2 for(j=i+1; j<=N; j++){ int c=*(ptr a+i); if(*(ptr a+i)>*(ptr a+j)) *(ptr b+j)=*(ptr b+j)+1; else *(ptr b+i)=*(ptr b+j)+1; } for(i=1; i<=N; i++) { printf("%4d ",*(ptr b+i)); } return 0; } </pre>	

指標與二維陣列

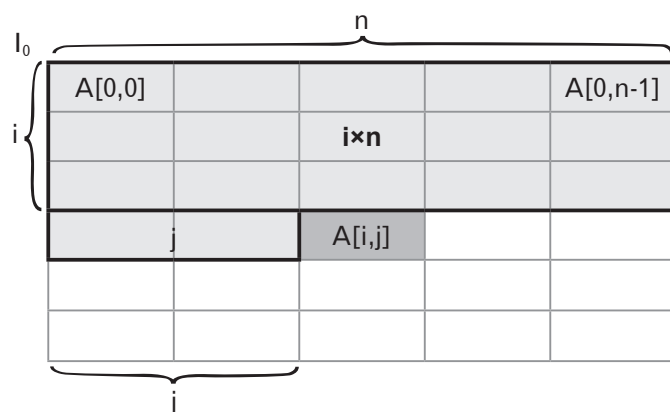
二維陣列在人們眼中通常用於處理二維的行列表格，但電腦內部的記憶體中，是以一維的方式排列，二維轉一維的轉換方式有兩種，分別是 Row-major 與 Column-major，如下表所示：



指標則是採用 Row-major。若第一個元素為 $A[0, 0]$ 位址是 I_0 ，則 $a[i, j]$ 位址公式為：

$$A[i, j] = I_0 + (i \times n + j) \times d$$

(n是每列元素個數，d是每個元素所佔記憶體的數量)



例如，若有資料如下：

0	1	2	3
4	5	6	7
8	9	10	11

先以二維陣列儲存，程式如下：

```
int a[3][4]={0,1,2,3},{4,5,6,7},{8,9,10,11}};
```

二維陣列以指標處理的步驟是先指定第一元素給指標變數，如以下敘述：

```
int *ptr;
ptr = &a[0][0];          /* 不能寫成 ptr=a */
```

若要存取其值，只要使用陣列名稱與適當的位移即可。例如：以下敘述可設定第 i 列第 j 行元素等於 5。

```
*( *(a+i ) + j ) =5;      /* 不能寫成 * ( *ptr+i ) + j = 5 */
```

上式若要以 ptr 表示，則應使用以下敘述將二維排列轉為一維的直線排列，其中 3 是每一維的欄數。

```
*( ptr + 3*i + j ) =5;    /* 不能寫成 * ( a + 3*i + j ) = 5 */
```

以上敘述將二維的列與行轉為一維的直線排列，如下表所示，括號內為二維（列、行），括號外的為一維直線的位移量。

(0,0)0	(0,1)1	(0,2)2	(0,3)3
(1,0)4	(1,1)5	(1,2)6	(1,3)7
(2,0)8	(2,1)9	(2,2)10	(2,3)11

自我練習

題號	題目	結果
1	下列 C 語言片段程式輸出結果為何？ (A) 3 (B) 4 (C) 5 (D) 程式錯誤。 <pre>int main()//技藝競賽112 { int num[3][3][3]={1,2,3,4,5,6,7,8,9}; int *ptr; ptr = &num[0][0][0]; ptr = ptr + 4; printf ("%d", *ptr); printf ("%d",num[0][0][1]);//2 return 0; }</pre>	

2	<pre> int main() { int a[3][4]={{0,1,2,3},{4,5,6,7},{8,9,10,11}}; int *p=&a[0][0]; printf("%d\n",*p); p++; printf("%d\n",*p); p=&a[0][0]; printf("%d\n",*p); printf("%d\n",*(*(a+1) + 2)); printf("%d\n",*(p + 4*1 + 2)); } </pre>	
3	<pre> #include <iostream> using namespace std; int main() { int a[3][4]={{0,1,2,3},{4,5,6,7},{8,9,10,11}}; int *p=&a[0][0]; cout <<*p<<endl; p++; cout<<*p<<endl; p=&a[0][0]; cout <<*p<<endl; cout<<*(*(a+1) + 2)<<endl; cout<<*(p + 4*1 + 2)<<endl; return 0; } </pre>	
4	<pre> int main() { int a[3][4]={{0,1,2,3},{4,5,6,7},{8,9,10,11}}; int *p=&a[0][0]; (*(a+0)+2)=12; printf("%d\n",a[0][2]); printf("%d",*(*(a+0)+2)); return 0; } </pre>	
5	<pre> int main() { int a[3][4]={{0,1,2,3},{4,5,6,7},{8,9,10,11}}; int *p=&a[0][0]; *(p+4*0+2)=15; printf("%d\n",a[0][2]); printf("%d",*(*(a+0)+2)); return 0; } </pre>	

```

6  int main(){// (進階習題)
    int a[7][7]={0,0,0,0,0,0,0},
               {1,50,60,70,0,0,0},
               {2,30,40,50,0,0,0},
               {3,70,80,90,0,0,0},
               {4,66,77,88,0,0,0},
               {5,22,33,44,0,0,0},
               {0,0,0,0,0,0,0}};
    int *ptr=&a[0][0]; // ptr指向陣列
    int i, j, k;
    const int stdnum=5;
    for(j=1; j<=stdnum; j++){ // 計算每人各科成績
        for(k=1; k<=3; k++){
            (*(a+j)+4) += (*(a+j)+k); // a[j][4]+=a[j][k]
        }
        (*(a+j)+4) /= 3; // a[j][4]/=3
    }
    /* 計數排序 */
    for(j=1; j<=stdnum; j++)
        *(ptr+7*j+5)=1; // a[j][5]=1 每人名次先填第一名
        /**(a+j)+5)=1;
    for(j=1; j<=stdnum; j++)
        for(k=j+1; k<=stdnum; k++)
            if( (*(a+j)+4) < (*(a+k)+4) )
                (*(a+j)+5) = (*(a+j)+5)+1; // a[j][5]++
            else
                (*(a+k)+5) = (*(a+k)+5)+1; // a[k][5]++

    /* 計算每人不及格科數 */
    for(j=1; j<=stdnum; j++)
        for(k=1; k<=3; k++)
            if( (*(a+j)+k) < 60) // a[j][k] < 60
                (*(a+j)+6) = (*(a+j)+6)+1; // a[j][6]++

    /* 統計各科平均 */
    for(j=1; j<=4; j++) {
        for(k=1; k<=stdnum; k++)
            (*(a+stdnum+1)+j) += (*(a+k)+j);
    }
    // a[stdnum+1][j]=a[stdnum+1][j]+a[k][j]

```

	<pre> (*(a+stdnum+1)+j) /= stdnum; // a[stdnum+1][j] /= stdnum } //output //cout<<" no chin eng math avg rank fail "<<endl; printf(" no chin eng math avg rank fail \n"); for(i=1; i<=stdnum+1; i++) { for(j=0; j<7; j++){ //cout.width(5); //cout<< (*(a+i)+j); // a[i][j] */ printf("%5d",*(a+i)+j)); } //cout<<endl; printf("\n"); } } } </pre>	
7	<pre> int main() { int a[2][3]={2,1,3},{0,1,0}; int b[2][3]={1,2,3},{1,1,0}; int c[2][3]; int *pa=&a[0][0]; int *pb=&b[0][0]; int *pc=&c[0][0]; for (int i=0;i<2;i++) for (int j=0;j<3;j++) *(pc+i*3+j)=*(pa+i*3+j)+*(pb+i*3+j); for (int i=0;i<2;i++){ for (int j=0;j<3;j++) printf("%d ",*(pc+i*3+j)); printf("\n"); } } </pre>	

8	<pre> int main() { //10-12 方法二 int a[2][3]={ {2,1,3}, {0,1,0} }; int b[2][3]={ {1,2,3}, {1,1,0} }; int c[2][3]; int *pa=&a[0][0]; int *pb=&b[0][0]; int *pc=&c[0][0]; for (int i=0;i<2;i++) for (int j=0;j<3;j++) *(*(c+i)+j)=*(*(a+i)+j)+*(*(b+i)+j); for (int i=0;i<2;i++){ for (int j=0;j<3;j++) printf("%d ",*(*(c+i)+j)); printf("\n"); } } </pre>	
---	---	--

10-3 指標與字串

範例10-3a

請鍵入以下程式，並觀察執行結果。

```

#include <stdio.h>
#include <stdlib.h>
int main( ){
    char s[] = "abcde";
    char *ptr = s;
    printf( "%c\n", *ptr ); //指向一個字元 a
    printf( "%s\n", ptr ); //代表整個字串 abcde
    ++ptr; //指標前進
    printf( "%c\n", *ptr ); //指向一個字元 b
    printf( "%s\n", ptr ); //bcde
    ptr--;
    printf( "%c\n", *ptr ); //指向一個字元 a
}

```

前面我們說明了如何使用字元陣列來處理字串，如以下程式：

```
char s[10]="abcde";
```

現在我們將示範如何使用指標來處理字串。首先須將字串變數宣告為指標，如下：

```
char *ptr=s;
```

正如我們上節所提過的，陣列名稱 `s` 代表的即為 `s` 陣列的基位址 (`&s[0]`)，因此在上列程式宣告中，都是以傳位址的方式進行。指標變數 `ptr` 所儲存的值即為 `s` 陣列的基位址，而非陣列中的資料項，接下來我們便可以利用指標的移動來存取字串。例如：

```
printf( "%c\n", *ptr );//指向一個字元 a
```

`*ptr` 的結果是 `a`。

```
printf( "%s\n", ptr );//代表整個字串 abcde
```

`ptr` 的結果為 `abcde`。

```
++ptr;
printf( "%c\n", *ptr ); //b
```

的結果為 `b`。

```
printf( "%s\n", ptr );//bcde
```

`ptr` 的結果為 `bcde`，代表基位址已經右移一個位置。

```
ptr--;
printf( "%c\n", *ptr ); //a
```

又得到 `a`。

自我練習

1. 請鍵入以下程式，並觀察執行結果。

題號	程式	結果
1	<pre> #include <stdio.h> //統測111 int main() { char Str[] = "abcde"; char *ptr = Str; printf("%s\n", ++ptr); printf("%s\n",ptr); printf("%c\n",ptr); printf("%c\n",*ptr); printf("%s\n",*ptr); return 0; } </pre>	
2	<pre> #include <stdio.h> #include <stdlib.h> int main(){ char *str="I am a good teacher."; char s[20]; int a=0,b=0; while(*str != '\0') { printf("%c",*str); s[a]=*str; a++; if(*str==' ') b++; str++; } printf("\n"); printf("%s\n",str); printf("%c\n",*str); printf("%s\n",s); printf("%d\n",a); printf("%d\n",b); return 0; } </pre>	

3	有一 C 程式片段 (如下圖) 執行的結果為「fine.」, 下列空白處填入何程式碼才正確? (技藝競賽程式組 109) (A): i = &data[1][5]; (B): i = &data[2][6]; (C): i = *data[1][5]; (D): i = data[2][6]; <pre> #include <string.h> #include <stdio.h> int main() { char data[3][20] = {"How are you?", "I am fine.", "See you later." }; char *i; printf("%s", i); } </pre>	
4	<pre> int main() { // 假設輸入「This is a book.」 寫出執行結果。 char in[MAX], *ptr; int c, i=0; printf("input: "); while((c=getchar()) !=EOF && c != '\n') { /* 將每個字元單獨讀入陣列中 */ in[i]=c; i++; } in[i]='\0'; /* 加入字串結束符號 */ ptr=in; /* 將陣列位址傳給指標 */ printf("\n%s", ptr); while(*ptr != '\0') { if(*ptr=='s') { /* 如果輸入的字串中有 's', 則代換為'S' */ *ptr = toupper(*ptr); } if(*ptr==' ') { /* 如果輸入的字串中有空白, 則代換 為換行 */ *ptr = '\n'; } ptr++; } ptr=in; printf("\n%s", ptr); return 0; } </pre>	

5	<pre> int main() { char in[MAX]="This is a book8.", *ptr; int c, i=0; ptr=in; /* 將陣列位址傳給指標 */ int digitnum=0; int uppernum=0; int lowernum=0; int spacenum=0; while(*ptr != '\0') { if(isdigit(*ptr)) digitnum++; if(isupper(*ptr)) uppernum++; if (islower(*ptr)) lowernum++; if (isspace(*ptr)) spacenum++; ptr++; } printf("%d\n",digitnum) ; printf("%d\n",uppernum) ; printf("%d\n",lowernum) ; printf("%d\n",spacenum) ; return 0; } </pre>	
---	--	--

4. 在跑馬燈的設計上，可以由陣列中取出文字，並且顯示於螢幕上。如下片段程式執行完後 ShowMessage 字串為 "ILoveTaiwan"，則 Count 的初始值應為下列哪一個？（統測 112）

```

1  const int Count = ? ;
2  char Dictionary[50] = "IWhLoaorevYTeouTW5aM7iynwyuaTYn?";
3  char ShowMessage[12] ={0};
4  char *P = &Dictionary[0];
5  int Index = 0, Search = 0;
6  for(Index =0; Index < 11 ; Index++)
7  {
8      ShowMessage[Index] = P[Search];
9      Search += Count;
10 }

```

- (A) 0 (B) 1 (C) 2 (D) 3

解：請鍵入以下程式，觀察 "IWhLoaorevYTeouTW5aM7iynwyuaTYn?"，發現有底線的字元，間隔是 3，所以 Count 為 3。

```
#include <stdio.h>
#include <math.h>
int main(){
    const int Count = 3 ;
    char Dictionary[50] = "IWhLoaorevYTeouTW5aM7iynwyuaTYn?";
    char ShowMessage[12] ={0};
    char *P = &Dictionary[0];
    int Index = 0, Search = 0;
    for(Index =0; Index < 11 ; Index++){
        ShowMessage[Index] = P[Search];
        Search += Count;
    }
    for (int i=0;i<=11;i++){
        printf("%c",ShowMessage[i]);
    }
    return 0;
}
```

5. 有關指標的程式設計，如下片段程式可以在【1】的位置加入哪一行程式碼，編譯時不會產生任何錯誤或警告訊息？（統測 112）

```
1  int *p;
2  char MyName[] = {'A'};
3  int MyAge;
4  double MyWeight;
5  【1】
```

- (A) p = MyName; (B) p = &MyName[0]; (C) p = MyWeight; (D) p = &MyAge;

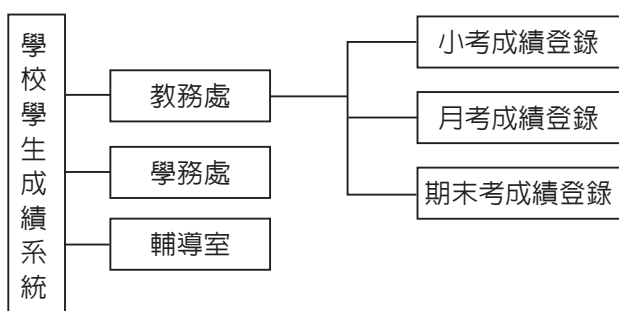
解：請鍵入以下程式。

```
#include <stdio.h>
int main(){
    int *p;
    char MyName[] = {'A'};
    int MyAge;
    double MyWeight;
    p=&MyAge; //指標型態務必和所指型態相同 P10-2
    char *pMyName=MyName;
    //char *pMyName=&MyName[0]; //P10-4
    double *pMyWeight=&MyWeight;
    return 0;
}
```

公用函式及自訂函式

概念

現今工業化的產品通常以模組開發，例如，一部電腦分成主機、螢幕、硬碟、鍵盤、滑鼠等模組，產品的設計與維修都是模組化，彼此獨立，也就是鍵盤壞了，就換鍵盤，滑鼠壞了就換滑鼠。軟體工業是發展較晚的新興產業，也是遵循此道理，希望將問題模組化，每個模組也和硬體系統一樣，是可組合、可分解和可更換的單元。模組化程式設計是指解決一個複雜問題時由上而下逐層把軟體系統劃分成若干模組。每個模組完成一個特定的子功能，所有的模組的功能與模組之間的介面要先定義，並可按某種方法組裝起來，完成整個系統所要求的功能。例如，要撰寫學校學生成績管理系統，就可先分成三個模組，分別是教務處的考試成績模組、學務處的操行系統模組、輔導處的輔導評量系統模組，這些子系統都可完成特定功能，且資料交換介面也已經定義且公開。教務處模組則再分為小考成績登錄模組、月考成績登錄模組、期末考成績登錄模組等系統，如下圖所示：



公用函式(Common functions)

以上每一軟體系統也應遵循模組化程式設計的原則，將使用者經常使用的一些功能撰寫成函式 (Functions)，這些由軟體系統廠商所開發的函式，稱為公用函式，公用函式於本書 11-1~11-4 節介紹。同樣的道理，使用者也可以將常用功能寫成函式，此稱為自訂函式 (Self-defined function)，自訂函式於本書 11-5 節介紹。

11-1 數值函式(Mathematical Functions)

C 語言中，有關數值運算的內儲函式大部分是被集中宣告在 `math.h` 標頭檔中，程式設計者要使用這些函式，要先在程式開頭以 `#include <math.h>` 載入這些函式庫函數的功能不是統測與 APCS 考試重點，所以，以下僅介紹 `abs()` 函數的說明，其餘請自行線上查詢。

1. abs

語法：
`int abs (int x);`
`double fabs (double x);`

說明：傳回 `x` 的絕對值。

例如：1. `abs(-3)= 3`

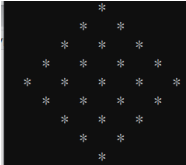
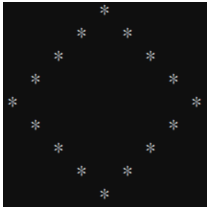
2. `abs(- 4.6)= 4` 不會 4.6，而是 4，這很容易錯，請小心。

若要傳回浮點數，請用 `fabs`。例如：

1. `fabs(-4.6)=4.6`

自我練習

題號	題目	結果
1	<pre>int main(){//技藝競賽程式組107 double a=9.0,b=8.0,c=9.0; printf("%.2f",sqrt(a+b*c)); }</pre> (A) 8.00 (B) 9.00 (C) 10.00 (D) 81.00	
2	在 C 程式需要使用數學函式，請問需要含括哪一個標頭檔？ (技藝競賽修護組 111) (A) <code><stdio.h></code> (B) <code><math.h></code> (C) <code><stdlib.h></code> (D) <code><string.h></code> 。	
3	請問 C 語言的亂數函式需要使用下列哪一個標頭檔？ (技藝競賽修護組 111) (A) <code><stdlib.h></code> (B) <code><math.h></code> (C) <code><stdio.h></code> (D) <code><string.h></code> 。	

4	程式填空。輸出結果如下：  <pre> int main(){ for(int i=4; i>=-4; i--) { for(int j=0; j<=____; j++) printf("* "); printf("\n"); } return 0; } </pre>	
5	程式填空。輸出結果如下：  <pre> int main(){ for(int i=4; i>=-4; i--) { for(int j=0; j<=____; j++) printf(" "); for(int j=0; j<=____; j++) printf("* "); printf("\n"); } } </pre>	
6	程式填空。輸出結果如下： 	

	<pre> int main(){ for(int i=4; i>=-4; i--) { for(int j=0; j<=abs(i); j++) printf(" "); for(int j=0; j<=4-abs(i); j++) if (_____) printf("* "); else printf(" "); printf("\n"); } } </pre>	
7	<pre> int main(){ double x=4,y=3; double r,s; r=(sqrt(x*x+y*y)); s=(atan2(y,x)*180/M_PI); printf("%4.0f∠%4.1f",r,s); return 0; } </pre>	

11-2 字元函式(Character Function)

在 C 語言中，字元的輸出入，除了可使用 `scanf()` 與 `printf()` 外，亦可使用 `getchar()` 與 `putchar()`，此二函式都是包含於 `stdio.h` 標頭檔。

1. `getchar()`

`getchar()` 的作用是用來讀取鍵盤輸入的一個字元，如以下敘述，

```

int b;
b = getchar();

```

可從鍵盤讀取一個字元，其與 `scanf("%c",&b)` 最大的不同是，`getchar()` 可輸入含有空白的字串。其次，與 `getchar()` 接近的函式還有 `getche()` 與 `getch()`，其差別是 `getche` 不用按『Enter』，且與 `getchar()` 相同，有輸出使用者所按鍵字元；`getch()` 不用按『Enter』，也沒有輸出按鍵內容。例如，請個別執行以下敘述，並比較其差別。

```
int b= getchar( );//有輸出，要Enter，include <stdio.h>
int b=getche( );//有輸出，不要Enter，include <conio.h>
int b=getch();//無輸出，不要Enter，include <conio.h>
```

2. putchar ()

putchar() 是將一個字元寫到 (輸出) 螢幕上。假設在上例中，我們透過鍵盤輸入一字元 'B'，'B' 將會被儲存在字元變數 b 中，這時便可以使用 putchar() 函式將其輸出至螢幕上，如下：

```
putchar(b );
```

結果會得到 'B'。

範例11-2a

請寫一個程式，可以輸入含有空白的字串。

執行結果

```
Gwosheng Horng
Hello Gwosheng Horng
```

程式列印

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    char c, d[20];
    int i=0;
    while ((c=getchar( )) !='\n'){
        d[i]=c;
        i++;
    }
    d[i]='\0';//要自己放字串結束字元
    printf("Hello %s\n", d);
    return 0;
}
```

自我練習

1. 同上範例，但將每一字元的 ASCII 碼的數值加 1 後輸出。

ctype.h

ctype.h 存有一些字元處理函式，分別說明如下：

1. isalpha(c)

判斷字元 c 是否為字母。若為真，則傳回非零值；否則傳回 "0"，例如：

```
printf("%d", isalpha(97)); //大寫A~Z的ASCII是65~90,小寫97~122
```

其結果為真，所以傳回非零值。又例如：

```
int a;  
a = isalpha(96);  
printf("%d", a);
```

其結果為 0。

2. isupper(c)

判斷字元 c 是否為大寫字母。若為真，則傳回非零值；否則傳回 "0"，例如：

```
printf("%d", isupper(65));
```

其結果為真，所以傳回非零值。又例如：

```
int a;  
a = isupper(96);  
printf("%d", a);
```

其結果為 0。

3. islower(c)

判斷字元 c 是否為小寫字母。若為真，則傳回非零值；否則傳回 "0"，例如：

```
printf("%d", islower(97));
```

其結果為真，所以傳回非零值。又例如：

```
int a;  
a = islower(65);  
printf("%d", a);
```

其結果為 0。

4. isdigit(c)

判斷字元 c 是否為數字字元（ASCII 48 ~57）。若為真，則傳回非零值；否則傳回 "0"，例如：

```
printf("%d", isdigit(48));
```

其結果為真，所以傳回非零值。又例如：

```
int a;  
a = isdigit(47);  
printf("%d", a);
```

其結果為 0。

5. isspace(c)

判斷字元 c 是否為空白字元（null）。若為真，則傳回非零值；否則傳回 "0"，例如：

```
printf("%d", isspace(32));
```

其結果為真，所以傳回非零值。又例如：

```
int a;  
a = isspace(33);  
printf("%d", a);
```

其結果為 0。

6. tolower(c)

傳回字元 c 所對應的小寫字母。例如：

```
printf("%c", tolower('A'));
```

其結果為 'a'。又例如：

```
int a;  
a = tolower(65);  
printf("%d", a);
```

其結果為 97。

7. toupper(c)

傳回 c 所對應的大寫字母。例如：

```
printf("%c", toupper('a'));
```

其結果為 'A'。又例如：

```
int a;
a = toupper(97);
printf("%d", a);
```

其結果為 65。

8. toascii(c)

傳回 c 所對應的 ASCII 碼。例如：

```
printf("%d", toascii('a'));
```

其結果為 97。

自我練習

1. 請自行執行以上函式，並觀察執行結果。

題號	題目	結果
1	<pre>int main(){ char c, s[50]; int n=0, i, alph=0, upper=0, digit=0, space=0; while((c=getchar())!='\n') { s[n]=c; n++; } for(i=0; i<n; i++) { if(isalpha(s[i])) alph++; if(isupper(s[i])) upper++; if(isdigit(s[i])) digit++; if(isspace(s[i])) space++; } printf("alphabet:%d\n", alph); printf("upper:%d\n", upper); printf("digit:%d\n", digit); printf("space:%d\n", space); }</pre>	

```
2  int main(){
    char a[]="Abc234";
    int b;
    char c;
    printf("%s",a);
    for (int i=0;i<=5;i++){
        printf("%c %d\n",a[i],a[i]);
    }
    for (int i=0;i<=5;i++){
        b=a[i];
        c=a[i];
        if (islower(b)){
            c=c+5;
            a[i]=c;
        }
        if (isupper(b)){
            c=c-6;
            a[i]=c;
        }
        if (isdigit(b)){
            c=c-2;
            a[i]=c;
        }
    }
    printf("%s",a);
}
```

11-3 字串函式(String Function)

C 的字串處理函式都放在 `string.h`，使用前應該先使用 `# include<string.h>`，本單元介紹常用字串函式如下：

1. `strlen`

語法：`size_t strlen( const char *str );`

說明：計算 `s` 字串之長度，不含 `'\0'`。

```
char s1[]="ABCD";
printf("%d",strlen(s1)); //4
```

2. strcmp

語法： `int strcmp( const char *s1, const char *s2 );`

說明： 比較 s1 與 s2 後，傳回一整數值。依字串順序，從第 0,1,2,3 個字元逐一比較，若 s1 小於 s2，傳回值為負數；若 s1 等於 s2，傳回值為 0；若 s1 大於 s2，傳回值為正數。（比較的依據是 ASCII 碼，「A」是 65，「B」是 66，「a」是 97，「b」是 98，順序較後面的英文字母其 ASCII 碼所對應數值較大，且小寫英文字母的 ASCII 碼數值大於大寫英文字母的 ASCII 碼數值）

```
char s1[]="ABCD";
char s2[]="ABCD";
char s3[]="FF";
char s4[]="AA";
printf("%d\n",strcmp(s1,s2)); //0，所有字元要完全相同才會0
printf("%d\n",strcmp(s1,s3)); //-1，s1-s3<0，本例第0個字元就分勝負了
printf("%d\n",strcmp(s1,s4)); //1，s1-s4>0，本例第1個字元才分勝負
```

3. strncmp

語法： `int strncmp( const char *lhs, const char *rhs, size_t count );`

說明： 依照指定 count 的個數，比較 lhs 與 rhs 字串，傳回 lhs-rhs 的差額。字串完全相同傳回 0，不完全相同時，從前面開始比較，只要有字元不同，就傳回其字元的差額，後面字元就不管了。

```
char s1[]="ABCD";
char s2[]="ABED";
char s3[]="FF";
char s4[]="AA";
printf("%d\n",strncmp(s1,s2,2)); //0，只取前面2個，所以相同
printf("%d\n",strncmp(s1,s3,2)); //-1，第0個字元就不同，傳回其差
printf("%d\n",strncmp(s1,s4,2)); //1，第1個字元才不同，傳回其差
```

4. strcpy

語法： `char *strcpy( char *dest, const char *src );`

說明： 將字串 src 的內容複製到字串 dest。原字串 dest 的內容已被字串 src 覆蓋。

```
char s1[]="ABCD";
char s2[]="1234";
```



```
strcpy(s1,s2); //s1=s2
printf("%s\n",s1); //1234
```

5. strncpy

語法： `char *strncpy( char *dest, const char *src, size_t count );`

說明： 將字串 `src` 的內容指定長度 `count` 複製到字串 `dest`。原字串 `dest` 的內容相對長度被字串 `src` 覆蓋。

```
char s1[]="ABCD";
char s2[]="1234";
strncpy(s1,s2,2);
printf("%s\n",s1); //12CD
```

例如：

```
char s1[]="ABCD";
char s2[]="1234";
strncpy(s1+1,s2,2); //可以指定從第幾個開始被取代
printf("%s\n",s1); //A12D
```

6. strcat

語法： `char *strcat( char *dest, const char *src );`

說明： 將字串 `src` 的內容連接到字串 `dest` 之後，傳回 `dest` 的內容。

```
char s1[]="ABCD";
char s2[]="1234";
strcat(s1,s2);
printf("%s\n",s1); //ABCD1234
```

7. strncat

語法： `char *strncat( char *dest, const char *src, size_t count );`

說明： 將字串 `src` 的內容取指定長度 `count` 連接到字串 `dest` 之後，傳回 `dest` 的內容。

```
char s1[]="ABCD";
char s2[]="1234";
strncat(s1,s2,2);
printf("%s\n",s1); //ABCD12
```

※ 8. strchr

語法： `char *strchr( const char *str, int ch );`

說明： 根據指定字元，將指標指到第一個出現在字串 `str` 中的相同字元，並傳回指標值，也就是字元搜尋。

```
char s1[]="ABCD";
printf("%d\n",strchr(s1,'B'));//傳回整個記憶體位址，這沒意義
//要減去相對s1的位址
printf("%d\n",strchr(s1,'B')-s1);//1
printf("%s\n",strchr(s1,'B'));//BCD
```

※ 9. strstr

語法： `char *strstr( const char* str, const char* substr );`

說明： 前面 `strchr` 是字元搜尋，`strstr` 則是子字串搜尋，傳回 `substr` 在 `str` 的起始位置。

```
char s1[]="ABCD";
char s2[]="BC";
printf("%d\n",strstr(s1,s2));//傳回整個記憶體位址，沒意義
printf("%d\n",strstr(s1,s2)-s1);//1
```

10. strrev

說明： 字串反轉。

```
char s1[]="ABCD";
strrev(s1);
printf("%s\n",s1);//DCBA
```

※ 11. strtok

說明： 以某一個字元分割字串為若干子字串函式。例如，翻譯軟體總要先將句子分為單字。

語法： `char *strtok( char *str, const char *delim );`

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main()
{
```

```

char a[]="How many people in your family? These are four
        people in my family.";
char c[]=" .?";//分割的字元有三種
char *d;//指標型態
d=strtok(a,c);//分割完，剩餘放NULL
printf("%s\n",d);
do{
    d=strtok(NULL,c);//下次從NULL繼續分割
    printf("%s\n",d);
}while (d!=NULL);
return 0;
}

```

執行結果如下：

```

How
many
people
in
your
family
These
are
four
people
in
my
family

```

範例11-3a

假設有資料如下：

Mary、Hand、洪煥維、陳惠敏、王一平

請寫一個程式，可以將以上資料排序輸出。排序的方式為由小而大，英文先輸出，英文按字母，中文按筆畫的多寡，都是由小而大。

執行結果

Before Sort:	Mary	Hand	洪煥維	陳惠敏	王一平
After Sort:	Hand	Mary	王一平	洪煥維	陳惠敏

👉 程式列印

1. C 語言沒有 string 型態，字串使用字元陣列，如以下程式的 `char a[][10]`。
2. C 語言沒有字串比較運算子 (`>`, `>=`, `<`, `<=`, `==`, `!=`)，而是要用本單元 `strcmp()` 函式。
3. C 語言字串不能用指派運算子 (`=`)，而是要用本單元 `strcpy()` 函式。

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main()
{
    char a[][10]={"Mary", "Hand", "洪煥維", "陳惠敏", "王一平"};
    int i, j;
    int n=5;
    // 輸出全部
    printf("Before Sort:\n");
    for (i=0; i<=n-1; i++){
        printf("%10s", a[i]);
    }
    printf("\n");
    //排序, 由小而大, 英文按字母, 中文按筆畫的多寡
    char t[10];
    for(i=0; i<=n-1-1; i++){
        for (j=0; j<=n-i-1-1; j++){
            if (strcmp(a[j], a[j+1])>0){ //字串比較, 請看11_3
                strcpy(t, a[j]); //字串複製, 請看11_3
                strcpy(a[j], a[j+1]);
                strcpy(a[j+1], t);
            }
        }
    }
    printf("After Sort:\n");
    for (i=0; i<=n-1; i++){
        printf("%10s", a[i]);
    }
    return 0;
}
```

自我練習

題號	題目	結果
1	<pre>int main(){//技藝競賽程式組 char *dataA[3]={"C++_is_a_widely","used_ computer","language"}; int dataB[3]={3,2,3}; char Strs[50]="",substr[10]=""; int j,k; for (j=0;j<3;j++){ strncpy(substr,dataA[j],dataB[j]%(j+2));//複 製n個字元 strcat(Strs,substr);//字串串接 } printf("%s",Strs); return 0; }</pre> (A) C++uslan (B) Cul (C) C++uselang (D) Cuslan	
2	<pre>int main(){ char s1[]="I am a super teacher."; char s2[]="am"; if(strstr(s1, s2)!=NULL) printf("found"); else printf("can not find"); return 0; }</pre>	
3	<pre>int main(){// (進階習題) char a[]="This is a book."; int alen=15; char b[]=" is"; int blen=3; int c; c=strstr(a,b)-a; printf("%d",c); printf("\n"); for (int i=c+blen;i<=alen;i++){ a[i-blen]=a[i]; } printf("%s",a); }</pre>	

```

4  int main(){// (進階習題)
    char s1[]="ABCDEFGH";
    char s2[10];
    //left
    int n=2;//從左邊取長度2個
    int i;
    for (i=0;i<n;i++){
        s2[i]=s1[i];
    }
    s2[n]='\0';//要幫忙放字串結束字元
    printf("%s\n",s2);
    //mid
    int m=2;n=4;//從索引m開始取4個
    for (i=m;i<m+n;i++){
        s2[i-2]=s1[i];
    }
    s2[m+n-2]='\0';//要幫忙放字串結束字元
    printf("%s\n",s2);
}

```

字串與數值轉換

字串與數值轉換都放在 `<stdlib.h>`，使用前必須在程式前加上

```
#include <stdlib.h>
```

1. atoi: 將字串轉整數。例如：

```

char a[10]="12345";
int b=atoi(a);
b++;
printf("%d\n",b);//12346

```

結果是 12346。

2. atof: 將字串轉為浮點數。例如：

```

char c[10]="123.45";
float d=atof(c);
d=d+5;
printf("%6.2f\n",d);//128.45

```

結果是 128.45。

3. itoa: 將數值轉為指定進位的字串。例如：

```
int num=127;
char f[10] ;
itoa (num,f,10); //轉為十進位字串
printf("%s\n",f); //127
printf("%d\n",strlen(f)); //3
```

又例如：

```
itoa (num,f,2); //轉二進位字串
printf("%s\n",f); //01111111
```

結果是 1111111。又例如：

```
itoa (num,f,16);
printf("%s\n",f); //7F
```

結果是 7F。

自我練習

1. 請鍵入以下程式，寫出執行結果。

```
int main( ){
    int num=127;
    char f[10] ;
    itoa (num,f,10); //轉為十進位字串
    printf("%s\n",f); //127
    printf("%d\n",strlen(f)); //3
    char *ptr;
    itoa (num,ptr,10); //轉為十進位字串
    printf("%s\n",ptr); //127
    printf("%d\n",strlen(ptr)); //3
    printf("%c\n",*ptr); //1
    ptr++;
    printf("%c\n",*ptr); //2
}
```

11-4 日期與時間函式(Data Function/Time Function)

題號	題目	結果
	<pre> int main() { char t1[]="18:50:00"; char t2[]="19:20:00"; long a1,a2; printf("%d\n",t1[0]); //輸出ASCII a1=(t1[0]-48)*10+(t1[1]-48); printf("%d\n",a1); //18 a1=((t1[0]-48)*10+(t1[1]-48))*60+(t1[3]- 48)*10+(t1[4]-48); //18*60+50 a2=((t2[0]-48)*10+(t2[1]-48))*60+(t2[3]- 48)*10+(t2[4]-48); //19*60+20 printf("%d\n",a2-a1); //30 return 0; } </pre>	

11-5 自訂函式(Self-defining function)

前面已經介紹一些公用函式，那在程式設計時，常會遇到某些程式片段需要在同一個程式或不同一個程式重複出現多次，如果這些程式片段都分別在每個地方寫一次，那是一件非常浪費時間的事，且會使程式變得冗長而不易閱讀；更糟的是，萬一要調整此程式片段的部分功能，更要至不同的地方修改，造成程式維護困難，此時可透過自訂函式解決以上困難。

自訂函式的使用方式是將此常用的程式片段賦予一個名稱，當程式設計者需要使用此程式片段時，只要使用此名稱即可呼叫此程式片段，此程式片段即稱為自訂函式，又稱為副程式（Subroutine）。其次，自訂函式完成之後可交由不同的程式呼叫使用，以節省程式設計者的時間。例如：已在前面章節所介紹過的數值函式、字元函式及字串函式等等，這些稱為公用內儲函式，若公用內儲函式無程式設計者要求的功能，此時就可使用本章的自訂函式。其次，自訂函式同樣可讓別的使用者使用。

✎ 函式之原型宣告與呼叫

在本章之前已經介紹了基本資料型態、陣列型態，本節將介紹函式型態。同理，使用函式前一樣要宣告函式型態，但函式的宣告較前面幾種複雜，它包含有函式的傳回型態、函式名稱、參數型態與參數名稱，所以另稱為函式原型 (Prototype)，而不稱為函式宣告。(雖然若說函式宣告，別人也知道指的是函式原型宣告，但較少人使用函式宣告)。函式的使用步驟是，步驟一：函式原型宣告，步驟二：實作函式本體，步驟三：呼叫函式。分別說明如下：

✎ 步驟一：函式原型宣告

函式原型宣告之簡要語法如下：

```
傳回值型態    函式名稱 ([ 參數型態1 參數名稱1, ...]....);
```

以上語法說明如下：

1. 傳回值型態為函式所要傳回的資料型態，可以是任一基本資料型態 (int、char 及 double 等)、結構及指標等，若無傳回值，則用 void 表示。
2. 函式名稱的命名同 4-1 節的識別字命名規則。
3. 參數型態與參數名稱必須成對出現。參數為主程式與函式傳遞資料或回傳執行結果的窗口，若無資料傳遞則括號內空白即可。
4. 函式原型宣告的位置，亦影響其有效範圍，其有效範圍同一般變數的宣告。
5. 以下程式片段為函式原型宣告，其意義是函式傳回一個整數型態，函數名稱為 add，且傳遞兩個整數型態給函式本體。

```
int add(int a, int b);
```

其次，函式原型宣告的變數名稱可省略，例如：以上敘述亦可簡寫如下：

```
int add(int , int );
```

✎ 步驟二：實作函式本體

實作函式本體的簡要語法如下：

```
傳回值型態    函式名稱  ( [ 參數 1, 參數 2...] ){
    [ 敘述區塊; ]
    [ return ( 運算式); ]
    [ 敘述區塊; ]
}
```

以上語法說明如下：

1. 傳回值型態與函式名稱須與函式原型宣告相同。
2. 每個函式可放在 `main` 函式前面或後面均可，若放在 `main()` 前面則函式原型宣告還可省略。其次，每個函式的地位均相同，所以不可以在函式中再定義其它函式。
3. 函式原則上從左大括號『{』執行到右大括號『}』，但若中途有特殊原因要離開函式，可用 `return` 提早離開。
4. 參數 (Parameter) 有些書亦稱為引數 (Argument)。函式呼叫的參數名稱與實作函式本體的參數名稱可不同，但其資料具有傳遞性，也就是函式呼叫的參數將會傳遞給函式本體，至於函式本體的運算結果是否回傳至主程式，那就得依參數的傳遞方式了。
5. 以下程式片段，可將所傳來的兩個參數相加並傳回。

```
int add(int a, int b){
    int c;
    c=a+b;
    return (c);
}
```

步驟三：呼叫函式

呼叫函式的簡要語法如下：

```
[ 傳回值 =]  函式名稱([ 參數 1, 參數 2, ...]);
```

以上語法說明如下：

1. 函式若無傳回值 (void)，則函式名稱前的傳回值可省略。
2. 主程式呼叫函式與函式原型宣告的參數稱為實際參數 (Actual Parameter)，函式本體的參數稱為形式參數 (Formal Parameter)，且兩者的名稱可以不同。實際參數會將參數值傳給形式參數，而形式參數是否將值傳回，則要看參數傳遞方式，參數的傳遞方式有傳值、傳址及傳參考。
3. 以下程式片段可呼叫 `add` 函式，傳回值皆為 8：

```
c=add(6,2);
```

或

```
m=6; n=2;
c=add(m,n);
```

以上三個步驟，全部程式如下：

```
#include <stdio.h>
#include <stdlib.h>
//1.函數原型宣告
int add(int a, int b);
int main()
{
    int m=6,n=2;
    int c;
    //3.呼叫函式
    c=add(m,n);
    printf("%d\n",c);
    c=add(6,2);
    printf("%d\n",c);
    return 0;
}
//2.實作函式實體
int add(int a, int b){//a,b分別按照順序接收m,n
    int c;
    c=a+b;
    return (c);
}
```

若將函數實體放在 main() 前面，可省略函數原型宣告，程式如下：

```
#include <stdio.h>
#include <stdlib.h>
//函式放在前面，可省略函式原型宣告
//1.實作函式實體
int add(int a, int b){
    int c;
    c=a+b;
    return (c);
}
int main()
{
    int m=6,n=2;
    int c;
    //2.呼叫函式
```

```

    c=add(m,n);
    printf("%d\n",c);
    c=add(6,2);
    printf("%d\n",c);
    return 0;
}

```

範例11-5a

試寫一程式，計算 C_n^m 的值。

提示： $C_n^m = \frac{m!}{n!(m-n)!}$ 。

👉 程式列印

1. 以下程式未使用函式，階乘共需重複寫三次，若要修改函式內容，則要到處修改，若沒有修改到，則造成程式的不一致。

```

#include <stdio.h>
#include <stdlib.h>
int main()
{
    int m=5, n=3, m1, m2, m3, m4, i;
    m1=1;
    for(i=1; i<=m; i++)
        m1=m1*i;
    m2=1;
    for(i=1; i<=n; i++)
        m2=m2*i;
    m3=1;
    for(i=1; i<=m-n; i++)
        m3=m3*i;
    m4=m1/(m2*m3);
    printf("%d",m4);
    return 0;
}

```

2. 以下以函式改寫如下，萬一要修改函式，也是在同一個地方修改。

```
#include <stdio.h>
#include <stdlib.h>
// 1. 實作函式實體
int sum(int m){
    int m1=1;
    int i;
    for(i=1; i<=m; i++)
        m1=m1*i;
    return(m1);
}
int main()
{
    int m=5, n=3, m1, m2, m3, m4;
    m1=sum(m); // 1. 函式呼叫
    m2=sum(n); // 1. 函式呼叫
    m3=sum(m-n); // 1. 函式呼叫
    m4=m1/(m2*m3);
    printf("%d",m4);
    return 0;
}}
```

自我練習

1. 試寫一程式，使用函式完成 $1! + 2! + 3! + 4! + \dots + n!$ (n 可由使用者輸入)

參數的傳遞

C 語言參數的傳遞有兩種情形，第一種是傳值 (Call by value)、第二種是傳址 (Call by address)，分別說明如下：

傳值呼叫

參數的傳遞若是傳值，則是一種單向的傳遞方式，參數僅能由主程式傳至函式本體，函式本體並不能藉由參數傳回值。也就是使用傳值參數至多能用函式名稱傳回一個值，若要傳回兩個以上的值，就要使用傳址或傳參考。

以傳值方式傳遞參數時，被傳遞的變數值通常只是被複製到其對應的參數，在函式執行過程，主程式中的變數之值並不會受到影響，其優點是保護主程式的值不會隨便被更動。例如：範例 11-5b 的 x, y 雖在函式有更動其內容，但並不回傳主程式。

範例11-5b

示範傳值呼叫。

```
void swap(int, int); //函式原型宣告
int main() {
    int x=3, y=4;
    swap(x, y);
    printf("3:x=%d,y=%d\n", x, y); //3,4
    return 0;
}
void swap(int p, int q) { //p接收x, q接收y
    int t;
    printf("1:p=%d,q=%d\n", p, q); //3,4
    t=p; p=q; q=t; //交換
    printf("2:p=%d,q=%d\n", p, q); //4,3
}
```

以上傳址呼叫，函式原型宣告、函式呼叫、函式原型如下表：

函式原型宣告	void swap(int, int);
函式呼叫	swap(x, y);
函式原型	void swap(int p, int q)

☛ 傳址呼叫

傳址呼叫參數是一種雙向的參數傳遞，因為傳址是共用記憶體位址，主程式不僅可以將參數傳至函式本體，函式本體內若參數的內容有所變動，亦可將變動的結果傳回主程式。

範例 11-5c 是傳址呼叫的寫法，x,y 在函式的更動，都會回傳主程式。

範例11-5c

示範傳址呼叫。

```
void swap(int *, int *); //函式原型宣告
int main() {
    int x=3, y=4;
    swap(&x, &y);
    printf("3:x=%d,y=%d\n", x, y); //4,3
}
```

```

    return 0;
}
void swap(int *p, int *q){//p接收x,q接收y
    int t;
    printf("1:p=%d,q=%d\n",*p,*q);//3,4
    t=*p; *p=*q; *q=t;//交換
    printf("2:p=%d,q=%d\n",*p,*q);//4,3
}

```

以上傳址呼叫，函式原型宣告、函式呼叫、函式原型如下表：

函式原型宣告	void swap(int *, int *);
函式呼叫	swap(&x, &y);
函式原型	void swap(int *p, int *q)

☞ 陣列的傳遞

陣列的名稱是指向陣列的開始位址，所以呼叫函式時，只要將陣列名稱傳給函式即可，且是一種傳址呼叫，陣列的執行結果將可回傳主程式。若為確保陣列在函式中不被修改，則可在函式本體與函式原型宣告的陣列型態前加 **const**。呼叫函式的語法如下：

函式名稱（陣列名稱）；

以下式子是傳遞 a 陣列至函式本體。

```

int a[5], n=5;
getarray(a, n); //請留意不要中括號

```

陣列函式本體的語法如下：

```

傳回值型態 函式名稱（陣列型態 陣列名稱[ ]）{
    敘述區塊；
}

```

以下敘述是函式本體的實例。

```

void getarray (int a[], int n){
}

```

函式原型的宣告同函式本體的實作，但同一般變數，可省略陣列名稱如下：

傳回值型態 函式名稱 ([陣列型態]) ;

以下敘述是傳遞陣列的函式原型宣告實例。

```
void getarray (int [], int);
```

範例11-5b

請完成以下陣列處理函式。

```
Before sort:
 99  88  77  66  55
After sort:
 55  66  77  88  99
```

👉 程式列印

```
#include <stdio.h>
#include <stdlib.h>
void print(const int a[],int n){
    int i;
    for (i=1;i<=n;i++){
        printf("%5d",a[i]);
    }
    printf("\n");
}
void swap(int *p, int *q){//傳址呼叫，p,q值才能傳回
    int t;
    t=*p; *p=*q; *q=t;
}
void sortarray(int a[],int N){
    int i,j;
    for(i=1; i<=N-1; i++)
        for(j=1; j<=N-i; j++)
            if(a[j]>a[j+1])
                swap(&a[j], &a[j+1]);
}
int main()
{
```



```

int a[]={0, 99, 88, 77, 66, 55};
int i, j;
const int N=5;
printf("Before sort:\n");
print(a,N);
//Sort
sortarray(a,N);
//output
printf("After sort:\n");
print (a,N);
}

```

自我練習

- 若要利用 C 語言寫一個 BMI 函式，此一函式接收傳入兩個整數資料，經計算後回傳的數值必須有小數點後至少兩位數精確度，BMI 函式的原型宣告應為下列何者？
(111 年統測)
(A) double BMI (); (B) void BMI (int, int);
(C) int BMI (int, int); (D) float BMI (int, int);
- 假設有資料如下：
33,99,88,66,55,22,88
請完成以下函式。
(1) 列印陣列。
(2) 求極大值。
(3) 將全部元素乘以 2。
(4) 刪除重複值。
- 曉華寫了下列一段 C 語言程式，想要測試程式執行時如何透過作業系統的終端機 (Console) 指令取得參數 (Arguments)，但發現無法成功進行編譯，應採取下列哪一個方案來解決這個問題？
(統測 112)

```

1
2  #include <stdio.h>
3  //void sub(int i, char *s);
4  int main(int argc, char *argv[]) {
5      sub(argc, argv[2]);
6      return 0;
7  }

```

```

8
9 void sub(int i, char *s){
10     printf("total %d arguments, and the 2nd one is %s\n", i, s);
11 }

```

- (A) 將行號 4 中 `main( int argc, char *argv[] )` 改為 `main()`
 - (B) 去掉行號 3 最前面的註解標記 `//`
 - (C) 將行號 1 的空白行刪除
 - (D) 在行號 1 新增 `#include <stdlib.h>`
- (1) 先鍵入以程式，程式的「`main(int argc, char *argv[])`」可用於接收執行執行程式所鍵入參式。

```

#include <stdio.h>
int main(int argc, char *argv[]) {
    printf("We have %d arguments:\n", argc);
    for (int i = 0; i < argc; ++i) {
        printf("[%d] %s\n", i, argv[i]);
    }
    return 0;
}

```

- (2) 鍵入以下程式

```

#include <stdio.h>
void sub(int i, char *s);
int main(int argc, char *argv[]) {
    sub(argc, argv[2]); //請留意，僅傳遞argv[2]
    return 0;
}
void sub(int i, char *s){
    printf("total %d arguments, and the 2nd one is %s\n", i, s);
}

```

僅輸出第 2 個

- (3) 請寫一個程式，可以執行程式且輸入邊長 a,b，計算其面積。

```

#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[]) {
    printf("We have %d arguments:\n", argc);
}

```

```

for (int i = 0; i < argc; ++i) {
    printf("[%d] %s\n", i, argv[i]);
}
int a=atoi(argv[1]);
int b=atoi(argv[2]);
int c=a*b;
printf("%d",c);
return 0;
}

```

題號	題目	結果
4	如下所示為主式呼叫副程式的 C 程式碼，則其顯示 xyz 的結果如何？ (技藝競賽程式組 112) <pre> main() { int x=8,y=4,z=5; SubProgram(&x,&y,&z); printf("x=%d y=%d z=%d\n",x,y,z); } </pre> <pre> SubProgram(int *x, int y, int *z) { *x = *x + y - *z; y += *x - *z; *z += *x - y; } </pre> x=7 y=6 z=6 (B) x=8 y=4 z=5 (C) x=7 y=4 z=6 (D) x=8 y=6 z=6 (A) x=7 y=6 z=6 (B) x=8 y=4 z=5 (C) x=7 y=4 z=6 (D) x=8 y=6 z=6	
5	<pre> SubProgram(int x,int *y,int *z){ //(技藝競賽程式組108) x=x+*y-*z; *y=x-*z-1; *z=x-*y; } int main(){ int x=12,y=8,z=4; SubProgram(x,&y,&z); printf("x=%d,y=%d,z=%d",x,y,z); return 0; } </pre> (A) x=12,y=8,z=4 (B) x=16,y=11,z=5 (C) x=12,y=11,z=5 (D) x=16,y=11,z=4	
6	以下哪一種傳遞的參數，不受參數變數內容變更的影響？ (技藝競賽程式組 112) (A) 值 (value) (B) 局部變數 (localvariable) (C) 參考 (reference) (D) 全域變數 (global variable)	

7	當 C 函式使用下列哪一個關鍵字開頭時，表示此函式沒有傳回值？ (技藝競賽修護組 111) (A) return (B) int (C) static (D) void。	
8	下圖程式執行結果為何？ (技藝競賽程式組 110) (A) 10 20 10 (B) 10 10 10 (C) 10 10 20 (D) 10 11 10 <pre> #include<stdio.h> int s = 1; void add(int); int main() { int s = 10; printf("%d ", s); add(s); printf("%d ", s); return 0; } void add(int a){ s = s + a; printf("%d ", s); } </pre>	
9	下圖程式執行結果為何？ (技藝競賽程式組 110) (A) 10 20 (B) 11 21 (C) 11 11 (D) 21 21 <pre> #include<stdio.h> void add(int); int main() { int s = 10; add(s); add(s); return 0; } void add(int a){ static int s = 1; s = s + a; printf("%d ", s); } </pre>	
10	下圖程式執行結果為何？ (技藝競賽程式組 110) (A) 10 20 (B) 11 21 (C) 11 11 (D) 21 21 <pre> #include<stdio.h> void add(int); int main() { int s = 10; add(s); add(s); return 0; } void add(int a){ int s = 1; s = s + a; printf("%d ", s); } </pre>	

11	下圖程式，如果輸入整數 1234，執行結果為何？ (A) 1234 (B) 4231 (C) 1324 (D) 4321 (技藝競賽程式組 110) <pre> #include <stdio.h> void printDigit(int n); int main(){ int n; scanf("%d", &n); printDigit(n); return 0; } void printDigit(int n){ printf("%d", n%10); if(n>10){ printDigit(n/10); } } </pre>	
12	請問下列程式執行後，輸出的第 12 個數值是？ (A) 2 3 (B) 40 3 (C) 2 40 (D) 40 2 <pre> #include <stdio.h> void funC(int* o, int q) { int p = 5; q = 3 + p; *o = p * q; } void main(){ int f=2, s=3; funC(&s, f); printf("%4d%4d", f, s); } </pre>	
13	請問下列程式執行後，輸入 "Happy Birthday"，輸出結果為？ (技藝競賽程式組 110) (A) 14 (B) Happy (C) 8 (D) Birthday <pre> size_t work(const char *s) { size_t x; //int for (x=0; *s!='\0'; ++s) ++x; return x; } int main() { char str[80]="Happy birthday"; printf("%d", work(str)); } </pre>	

變數與常數的有效範圍

在一個大型程式裏，一個程式是由一或數個函式組合而成。這些函式通常是由許多人合力完成，爲了避免彼此變數互相干擾，例如張三設 `stunum=15`，李四又設 `stunum=20`，則必然造成無法預期的錯誤，所以爲了防止變數互相干擾，才有變數的有效範圍（又稱爲變數生命週期），茲將變數的有效範圍敘述如下：

任一變數的宣告，若無特殊聲明，均屬於區域變數 (Local Variable)，其有效範圍僅止於該變數所在的程式區塊。例如，若宣告於函式中，則它的有效範圍僅止於該函式，其它的函式均無法取得該區域變數的值；若宣告在 `main()` 的前面，才是全域變數 (Global Variable)。例如，以下程式，全域的 `a` 有效範圍是 `main()`、`setup()`，`c` 的有效範圍則是 `main()`。其次，`main()` 與 `setup()` 內雖然都有變數 `b`，但此 `b` 在不同函式只是同名而已，它們各自佔有不同的記憶體，且其值不會互相干擾。就如同不同家庭會有相同的人名，但他們一定可配到不同身份證號碼，兩者不會互相干擾。

```
#include <stdio.h>
#include <stdlib.h>
int a=6;//全域變數
void setup(){
    int b=1;//區域變數
}
int main(int argc, char *argv[]) {
    int b=2; //區域變數
    int c=3; //區域變數
    printf("%d\n",a);//使用全域
    return 0;
}
```

自我練習

1. 請鍵入以下程式，並觀察執行結果。

程式	執行結果
<pre>#include <stdio.h> #include <stdlib.h> int a=6; void setup(){ int b=1; a=3;</pre>	

```

        printf("b=%d\n",b);
    }
    int main(int argc, char *argv[]) {
        int b=2;
        int c=3;
        printf("a=%d\n",a); //使用全域
        printf("b=%d\n",b);
        setup();
        printf("b=%d\n",b);
        printf("a=%d\n",a);
        return 0;
    }

```

以下程式，又於 setup() 內重複宣告 int a；那也合理，那此 a 的生命，僅止於 setup()，不會影響全域變數。

```

#include <stdio.h>
#include <stdlib.h>
int a=6;
void setup(){
    int a=3;
    printf("b=%d\n",a); //使用區域
}
int main(int argc, char *argv[]) {
    printf("a=%d\n",a); //使用全域
    setup();
    printf("a=%d\n",a);    //使用全域
    return 0;
}

```

3. 寫出程式執行結果。(技藝競賽程式組 107)

```

int s=1; //全域變數
void add(int);
int main(){
    printf("%d,",s);
    add(s);
    add(s);
    printf("%d,",s);
    s++;
}

```

```

    printf("%d,",s);
}
void add(int a){
    int s=6;
    for (;a>=0;a--){
        printf("%d,",s);
        s++;
        printf("%d,",s);
    }
}

```

- (A) 1,6,7,7,8,6,7,7,8,1,2 (B) 1,6,7,7,8,8,9,9,10,1,2
 (C) 1,6,7,7,8,8,9,9,10,10,11 (D) 1,1,2,2,3,3,4,4,5,5,6

4. 閱讀下文，回答第 40 - 42 題

曉華想要了解 C 語言程式區域變數 (Local variable) 和全域變數 (Global variable) 的數值變化情形，撰寫了下列的程式。

```

1  #include <stdio.h>
2  int sum=1, x=10;
3  int inc(int xin){
4      int sum=2;
5      sum = sum + xin;
6      xin++;
7      return (sum);
8  }
9  int main(){
10     int sum = 3;
11     sum=inc(x);
12     printf("%d, %d", sum, x);
13     return 0;
14 }

```

解：本例請鍵入以下程式，觀察執行結果。

```

#include <stdio.h>
int sum=1, x=10;
int inc(int xin){
    //xin=10
    int sum=2;
    sum = sum + xin;//2+10
}

```



```

    xin++; //11 但不會傳回
    return (sum); //12
}
int main(){
    int sum = 3;
    sum=inc(x); //x=10,但不會被修改
    printf("%d, %d", sum, x); //12 10
    return 0;
}

```

40. 下列何者為程式執行結果？（答案是 D）
 (A) 11, 11 (B) 13, 10 (C) 12, 11 (D) 12, 10
41. 在執行到行號 12 的時候，想要讓 x 的值隨著行號 6 中 xin 的值更新，下列修改程式的方式何者正確？（答案是 A）
 (A) 行號 11 的 x 改為 &x，並將函式 inc() 中所有的 xin 全部改為 *xin
 (B) 行號 11 的 x 改為 *x，並將函式 inc() 中所有的 xin 全部改為 &xin
 (C) 行號 11 的 x 改為 &x，並將函式 inc() 中所有的 xin 全部改為 &xin
 (D) 行號 11 的 x 改為 *x，並將函式 inc() 中所有的 xin 全部改為 *xin

解：本例請鍵入以下程式，觀察執行結果。

```

#include <stdlib.h>
#include <stdio.h>
int sum=1, x=10; //全域變數
int inc(int *xin){
    int sum=2; //區域
    printf("%d\n", *xin); //10
    sum = sum + *xin;
    *xin++; //不行
    (*xin)++; //OK
    printf("%d\n", *xin); //11
    return (sum);
}
int main(){
    int sum = 3; //區域
    sum=inc(&x); //傳址呼叫
    printf("%d, %d\n", sum, x); //12 11
    return 0;
}

```

42. 關於行號 2、行號 4、以及行號 10 的變數 sum 的敘述，下列何者正確？（答案是 A）
- (A) 行號 2 的 sum 是全域變數，行號 4 的 sum 是區域變數
 - (B) 行號 2 的 sum 是區域變數，行號 4 的 sum 是全域變數
 - (C) 行號 2 的 sum 和行號 10 的 sum 都是區域變數
 - (D) 行號 2 的 sum 和行號 10 的 sum 都是全域變數

11-6 遞迴

（備註：遞迴不是程式設計實習課綱，111 與 112 也沒考，但 APCS 有列入考試範圍，是 APCS 學科術科的重點，技藝競賽也是必考）

遞迴 (Recursive) 是一種分而治之的經典方法，因為在分治法裡，常常遇到先處理一個或一部份，剩下部分的方法竟然和原方法相同，既然和原方法相同，那可否呼叫或使用原函式？答案是可以的，這樣重複呼叫原函式，就稱為遞迴（此一演算法在高中數學有介紹，所以當然是 APCS 考試重點）。其次，『遞迴』可以實現程式設計抽象化的目的，其程式寫法也真的很抽象，要能有『您傻瓜，他聰明』的體會與意境，跟著『畫老虎、畫蘭花』，才能豁然開朗，因為遞迴不用很精密的寫程式，只要跟著演算法寫程式，所以其實更適合文科學生練習，因為有很多人不容易融入，而且很重要，所以我舉很多實例如下，希望大家能享受頓悟的樂趣。

階乘

自然數的階乘運算定義如下：

$$N! = 1 * 2 * 3 \cdots * N$$

若以函式表示，則其敘述如下：

$$f(x) = 1 * 2 * 3 \cdots * x$$

依照以上定義，可以以循序 for 迴圈撰寫如下：

```
int main()
{
    int s=1;
    int i;
    for (i=1;i<=6;i++){
        s=s*i;
    }
}
```

```
printf("%d",s);
return 0;
}
```

遞迴則是另一種運算思維，它是一種反向思考，強調分而治之，也就是將一批大資料，先處理一筆或一批（也可說去掉一筆或一批），但是剩下的資料，也可用原資料的方法處理。例如，本例階乘問題

$$f(x)=1*2*3\cdots*x$$

若以遞迴來探討，則從大的開始分解，以 $x=6$ 為例

$$f(6)=6*5*4*3*2*1$$

我們嘗試將 6 提出，剩下的竟也和原方法相同的道理，所以

$$\begin{aligned} f(x) &= x * f(x-1) \\ &= x * (x-1) * f(x-2) \\ &= x * (x-1) * (x-2) * f(x-3) \end{aligned}$$

只要 $x>1$ 則繼續分解，直到 $x=1$ 時 $f(x)=1$ 。其次，遞迴一定要有結束回去的條件，這就是遞迴的第二個重點，以上運算思維可以以數學式表示如下：

$$f(x) = \begin{cases} x * f(x-1), & x > 1 \\ 1, & x = 1 \end{cases}$$

以上是數學語言，若要以程式語言表示，則程式如下：

```
int f(int x){
    int p;
    if(x>=1)
        p=x*f(x-1); //抽一個出來，剩下也是用原方法
    else
        p=1; //x=1，p也是1
    return (p);
}
int main()
{
    int f(int x);
    int x=6, d;
    d=f(x);
}
```

```
    printf("%d",d);
    return 0;
}
```

以上 f() 也可簡化如下：

```
int f(int x){
    if(x>=1)
        return(x*f(x-1));
    else
        return(1);
}
```

爲了讓大家確實瞭解遞迴的運作，請在上一 f() 函式加上

```
int p=0;
printf("p=%d,x=%d\n",p,x);
```

則程式執行結果如下：也就是遞迴結束，p 才有值。

```
p=0,x=6
p=0,x=5
p=0,x=4
p=0,x=3
p=0,x=2
p=0,x=1
p=0,x=0
720
```

遞迴就是這樣，把握兩個重點，如何去分，如何回去，照著這兩個重點去『唬爛』，結果就會跑出來。『抽一個出來，剩下也是用原方法，其值是 $x*f(x-1)$ 』，『最後 $x=1$ 時，函式值爲 1』。所以文科學生應該更適合用遞迴寫程式。請繼續看以下範例。

費式數列

又例如，費式數列定義如下：

```
f1=1:f2=1
n>=3, fn=fn-1+fn-2  n>=3
```

依照以上定義，所以費式數列前 8 項如下：

```
1,1,2,3,5,8,13,21
```

依照以上定義，以循序法 for 撰寫如下：

```
int main()
{
    int f[100];
    f[1]=1;f[2]=1;
    int n=5;
    for (int i=3;i<=n;i++){
        f[i]=f[i-1]+f[i-2];
    }
    printf("%d",f[n]);
    return 0;
}
```

費式數列與遞迴

遞迴就是反向思考，從大的分解，強調分而治之，所以，

```
n>2 時 f(x)=f(x-1)+f(x-2)
n<=2 時 f(x)=1
```

x 以 5 為例

```
f(5)=f(4)+f(3)
    =f(3)+f(2)+f(2)+f(1)
    =f(2)+f(1)+3
    =5
```

以上是數學語言，若要以程式語言表示，則程式如下：

```
int f(int x){
    if (x>=3)
        return (f(x-1)+f(x-2));
    else
        return(1);
}
int main()
{
    int n=5;
    printf("%d",f(n));
    return 0;
}
```

以上程式，也是把握兩個重點，如何去分，如何回去，『 $x \geq 3$ 』，就分解為 $x-1$ 與 $x-2$ ，且用原方法』，『直到 $x=2$ 或 $x=1$ ，此函式就等於 1』。以上程式也是看著演算法跟著『唬爛』，就能得到結果。這都是程式抽象化的意境，程式編譯器要能達到『您傻瓜，他聰明』，人人都能用簡單的指令解決困難、複雜的問題。

二分法與遞迴

二分搜尋法當然也適用分治的遞迴法，也是把握兩個重點、如何去分、如何回去，就可寫出程式，本題程式如下：

```
#include <iostream>
#include <cmath>
using namespace std;
float f(float x,float &x1,float &x2){//目標,下界,上界
    float y,t;
    if (fabs(x1-x2)<0.001)
        return x1;
    else {
        y=(x1+x2)/2;
        t=y*y;
        if (t<x)
            f(x,y,x2); //猜的太小，調整下界，然後呼叫原方法
        else
            f(x,x1,y); //猜的太大，調整上界
    }
}
int main(int argc, char** argv) {
    float x=9,x1=0,x2=9;
    cout<<f(x,x1,x2); //目標,下界,上界
    return 0;
}
```

函數運算 (APCS 108年2月)

範例11-6a

定義 $f(x)=x+1$, $g(x,y)=x+y$, $h(x,y,z)=x+y-z$ ，輸入是簡化的，請看以下測試資料。其次，評分依照 3 個等級給分。等級 1 例如， $f\ 1$ ，代表 $f(1)$ ，並求其值，答案是 2； $f\ f\ 1$ ，代表 $f(f(1))$ ，並求其值，答案是 3，這樣可得 10 分；等級 2 例如， $h\ f\ 1\ g\ 2\ 3\ 4$ 代表 $h(f(1),g(2,3),4)$ ，並求其值，答案是 3，這樣可得 40 分；等級 3 例如， $h\ f\ 100\ g\ 20\ 300\ 4000$ ，數字長度不限，這樣可得滿分。

👉 參考解答

爲了充分理解這題，我將程式循序漸進分成以下 7 個程式。其次，我僅用到本書有提的指令與函數，這樣可減輕讀者的負擔。也就是高中生程式設計要把重點放在資料結構與演算法，盡量少用龐大函式，這樣學習才不會失焦。

1. 程式 1，能解測資 $f\ f\ 1$ 。
2. 程式 2，能解測資 $h\ f\ 1\ g\ 2\ 3\ 4$ 。
3. 程式 3，要能解出 $h\ f\ 100\ g\ 2\ 3\ 40$ 。
4. 循序法。

👉 程式列印

1. 程式 1，能解測資 $f\ f\ 1$ ，這題要抓住一個重點，取到 f ，表示是 $f(x)$ 那就繼續取一個，且回去；若取到數值就回去。

```
#include <ctype.h> //use isdigit()
int sum(char a[], int & p) {
    char b=a[p];
    if (isdigit(b)) {
        int x=(int)b-48;
        return(x); //遇到數值，抓數值回去
    }
    else {
        p=p+2;
        int x=(sum(a,p)); //遇到f要繼續取一個，且將結果傳回去
        return(x+1); //並回結果
    }
}
```

```
}  
int main()  
{  
    printf("Hello world!\n");  
    //char a[]="f 5";  
    //char a[]="f f 5";  
    char a[]="f f f 5";  
    int p=0;  
    int b=sum(a,p);  
    printf("b=%d",b);  
    return 0;  
}
```

2. 程式 2，能解測資 h f l g 2 3 4 。

```
#include <ctype.h>  
int sum(char a[],int & p){  
    char b=a[p];  
    if (isdigit(b)){  
        int x=(int)b-48;  
        return(x);  
    }  
    else {  
        if (b=='f'){  
            p=p+2;  
            int x=sum(a,p);  
            return(x+1);  
        }  
        else if (b=='g') {  
            p=p+2;  
            int x=sum(a,p);  
            p=p+2;  
            int y=sum(a,p);  
            return(x+y);  
        }  
        else if (b=='h'){  
            p=p+2;  
            int x=sum(a,p);  
            p=p+2;  
            int y=sum(a,p);
```



```

        p=p+2;
        int z=sum(a,p);
        return(x+y-z);
    }
}
int main()
{
    //char a[]="f 5";//6
    //char a[]="f f 5";//7
    //char a[]="g g 1 2 g 1 0";//4
    //char a[]="h h 1 2 3 h 1 1 1 h 1 1 0";//-1
    char a[]="h f 1 g 2 3 4";//3
    //char a[]="h g 1 2 h 1 2 1 f 1";//3
    //char a[]="g f h 1 4 f 5 f 2";
    //g f h 1 4 6 f 2//g f -1 3//g 0 3//3
    int p=0;
    int b=sum(a,p);
    printf("%d",b);
    return 0;
}

```

3. 程式 3，要能解出 h f 100 g 2 3 40。

```

#include <ctype.h>//use isdigit()
int sum(char a[],int & p){
    char b=a[p];
    if (isdigit(b)){
        int x=(int)b-48;
        while(isdigit(a[p+1])){
            p=p+1;
            b=a[p];
            int y=int(b)-48;
            x=10*x+y;
        }
        return(x);
    }
    else {
        if (b=='f'){
            p=p+2;

```

```
        int x=sum(a,p);
        return(x+1);
    }
    else if (b=='g') {
        p=p+2;
        int x=sum(a,p);
        p=p+2;
        int y=sum(a,p);
        return(x+y);
    }
    else if (b=='h'){
        p=p+2;
        int x=sum(a,p);
        p=p+2;
        int y=sum(a,p);
        p=p+2;
        int z=sum(a,p);
        return(x+y-z);
    }
}

int main()
{
    //char a[]="f 5";//6
    //char a[]="f f 5";//7
    //char a[]="g g 1 2 g 1 0";//4
    //char a[]="h h 1 2 3 h 1 1 1 h 1 1 0";//-1
    //char a[]="h f 1 g 2 3 4";//3
    char a[]="h f 100 g 2 3 40";//66
    //char a[]="h g 1 2 h 1 2 1 f 1";//3
    //char a[]="g f h 1 4 f 5 f 2";
    //g f h 1 4 6 f 2//g f -1 3//g 0 3//3
    int p=0;
    int b=sum(a,p);
    printf("%d",b);
    return 0;
}
```

4. 習慣了遞迴，您就會體會遞迴較輕鬆，若還是不習慣使用遞迴，那就要很精確的用循序法寫程式，演算法如下：

(1) 計算字串長度。

(2) 依字串長度，從字串尾巴每次取 1 個字元。

(2-1) 若此字元是運算元，則進堆疊。

(2-2) 否則，若是函式運算子，則依照函數取出所需引數的數量、運算、再將運算結果放進堆疊。

(3) 運算完畢，則堆疊僅剩一個，那就是答案。根據以上演算，程式撰寫如下：

```
#include <ctype.h>
void push (int a[],int d,int &top){
    //傳址，雙向傳遞
    top++; //堆疊頂端指標
    a[top]=d;
}
int pop (int a[],int &top){
    int b;
    b=a[top];
    top--;
    return( b);
}
int main()
{
    //char a[]="f 5";//6
    //char a[]="f f 5";//7
    //char a[]="g g 1 2 g 1 0";//4
    //char a[]="h h 1 2 3 h 1 1 1 h 1 1 0";//-1
    //char a[]="h f 1 g 2 3 4";//3
    //char a[]="h g 1 2 h 1 2 1 f 1";//3
    char a[]="g f h 1 4 f 5 f 2";
    //g f h 1 4 6 f 2//g f -1 3//g 0 3//3
    int i=0,len=0;
    char d;
    //計算字串長度
    do {
        d=a[i];
        i++;
    }
    while (d!='\0');
```

```
len=i-1;
printf("The String len=%d\n",len);
char b;
int s[20];
int top=-1;
int x,y,z;
//從尾巴開始抓
for (int i=len;i>=0;i--){
    b=a[i];
    if (isdigit(b))
        push (s,(int)(b-'0'),top);
    else
        switch( b){
            case 'f':
                x=pop(s,top);
                b=x+1;
                push (s,b,top);
                break;
            case 'g':
                x=pop(s,top);
                y=pop(s,top);
                b=x+y;
                push (s,b,top);
                break;
            case 'h':
                x=pop(s,top);
                y=pop(s,top);
                z=pop(s,top);
                b=x+y-z;
                push (s,b,top);
                break;
        }
}
b=pop(s,top);
printf("%d",b);
return 0;
}
```

遞迴的觀念題

遞迴不但是實作題的必考，觀念題每年也都必考 3~4 題。遞迴的觀念題主要分為三類型，第一類是純遞迴的追蹤。例如：

請問以 `a(13,15)` 呼叫右側 `a()` 函式，函式執行完後其回傳值為何？ 105/03/07

- (A) 90
- (B) 103
- (C) 93
- (D) 60

```
int a(int n, int m) {
    if (n < 10) {
        if (m < 10) {
            return n + m ;
        }
        else {
            return a(n, m-2) + m ;
        }
    }
    else {
        return a(n-1, m) + n ;
    }
}
```

參考解答

使用人腦直接推演如下：

$$\begin{aligned}
 a(13,15) &= a(12,15) + 13 = a(11,15) + 12 + 13 = a(10,15) + 11 + 25 \\
 &= a(9,15) + 10 + 36 = a(9,13) + 15 + 46 = a(9,11) + 13 + 61 \\
 &= a(9,9) + 11 + 74 = 18 + 85 = 103
 \end{aligned}$$

若使用電腦協助推演，則程式如下，請觀察執行結果，考試時，沒電腦，也是要這樣推演。

```
int a(int n, int m) {
    printf("n=%d,m=%d\n",n,m);
    //略。。。
}

int main(int argc, char** argv) {
    printf("%d",a(13,15));
    return 0;
}
```

又例如：給定右側 `g()` 函式，`g(13)` 回傳值為何？

105/03/10

- (A) 16
- (B) 18
- (C) 19
- (D) 22

```
int g(int a) {
    if (a > 1) {
        return g(a - 2) + 3;
    }
    return a;
}
```

參考解答

使用人腦直接推演如下：

$$\begin{aligned}
 g(13) &= g(11) + 3 = g(9) + 3 + 3 = g(7) + 3 + 6 = g(5) + 3 + 9 \\
 &= g(3) + 3 + 12 = g(1) + 3 + 15 = 1 + 18 = 19
 \end{aligned}$$

第二類型是函式還沒執行完，又呼叫函式，那還沒被執行的程式敘述，就要自己畫一個堆疊，幫它放進堆疊，等到遇到 `return` 時，才將這些堆疊的內容通通輸出。例如：給定右側函式 `f1()` 及 `f2()`。`f1(1)` 運算過程中，以下敘述何者為錯？

105/03/12

- (A) 印出的數字最大的是 4
- (B) `f1` 一共被呼叫二次
- (C) `f2` 一共被呼叫三次
- (D) 數字 2 被印出兩次

```
void f1 (int m) {
    if (m > 3) {
        printf ("%d\n", m);
        return;
    }
    else {
        printf ("%d\n", m);
        f2(m+2);
        printf ("%d\n", m);
    }
}

void f2 (int n) {
```

```
    if (n > 3) {
        printf ("%d\n", n);
        return;
    }
    else {
        printf ("%d\n", n);
        f1(n-1);
        printf ("%d\n", n);
    }
}
```

參考解答

這是第二種典型類型，有遞迴，且函式還沒作完，又去呼叫別的函式，那函式還沒作的程式片段，要先放入堆疊，等會兒遇到 `return` 時，要將全部堆疊由上而下清空。

```
f1(1)=1+f2(3)+ printf("%d\n",m);
```

此時

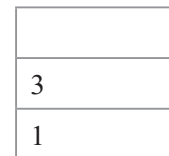
```
printf("%d\n",m);
```

沒有被執行，但又呼叫 `f2`，所以要將其結果放到堆疊，如下圖：(要等到 `return` 時，才能輸出)



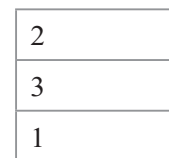
堆疊

```
=1+3+f1(2)+
```



堆疊

```
=1+3+2+f2(4)+
```



堆疊

```
=1+3+2+4+return (要清光堆疊)
```

```
=1+3+2+4+2+3+1
```

第三類型是數字很大，那就要尋找規律性。例如：右側程式輸出為何？ 105/03/14

- (A) bar: 6
bar: 1
bar: 8
- (B) bar: 6
foo: 1
bar: 3
- (C) bar: 1
foo: 1
bar: 8
- (D) bar: 6
foo: 1
foo: 3

```
void foo (int i) {
    if (i <= 5) {
        printf ("foo: %d\n", i);
    }
    else {
        bar(i - 10);
    }
}

void bar (int i) {
    if (i <= 10) {
        printf ("bar: %d\n", i);
    }
    else {
        foo(i - 5);
    }
}

void main() {
    foo(15106);
    bar(3091);
    foo(6693);
}
```

參考解答

這題數字有點大，那就是要找規律性，我們發現只要這兩個函式大於 15，通通可以一直扣掉。所以

```
15106/15=1007...1
15106-15*1006=16
foo(15106)=foo(16)=bar(6)=bar:6
bar(3091)=bar(16)=foo(11)=bar(1)=bar:1
foo(6693)=foo(18)=bar(8)=bar:8
```

其次，遞迴追蹤的過程，要仔細觀察數字是否已經重複，若重複就可直接根據前面的結果填入答案，這樣才能省時間。例如，右側 g(4) 函式呼叫執行後，回傳值為何？ 105/03/24

- (A) 6
(B) 11
(C) 13
(D) 14


```

int f (int n) {
    if (n > 3) {
        return 1;
    }
    else if (n == 2) {
        return (3 + f(n+1));
    }
    else {
        return (1 + f(n+1));
    }
}

int g(int n) {
    int j = 0;
    for (int i=1; i<=n-1; i=i+1) {
        j = j + f(i);
    }
    return j;
}

```

參考解答

又是遞迴問題，使用人腦追蹤如下：

```

f(1) = 1 + f(2) = 1 + 3 + f(3)
      = 4 + 1 + f(4) = 5 + 1 = 6
f(2) = 3 + f(3) = 3 + 1 + f(4) = 4 + 1 = 5，或看上面，已經重複，等於5。
f(3) = 看上面，已經重複，等於 2。所以
j = f(1) + f(2) + f(3) = 13 。

```

課後習題

本章的另一重點是變數的有效範圍，請鍵入以下程式，並觀察執行結果。

題號	題目	結果
1	<pre> const double PI=3.14159;//全域常數 int a;//全域a int main(int argc, char** argv) { int b, c; a=5;//全域a c=5;//main 內 c for (int a =2 ;a<=3;a++){//好習慣 //for 內a,僅for內有效,不會改變全域的a } } </pre>	

	<pre> //反過來說,若未宣告a,那就慘了,會影響全域a } if (c>2){ int a; a=1;//if 內a,僅if內有效 } return 0; } void aa(){ int c, d;//c,d 僅aa內有效 c=3;//aa內的c,僅aa內有效 } void bb(){ a=8;//使用全域a } </pre>	
2	<pre> #include <stdio.h> #include <stdlib.h> int a=1; void aa(){ printf("1.a=%d\n",a); int a=2; printf("2.a=%d\n",a); } void bb(){ printf("4.a=%d\n",a); a=3; } int main(int argc, char** argv) { aa(); printf("3.a=%d\n",a); bb(); printf("5.a=%d\n",a); return 0; } </pre>	
3	<pre> double Func(int w,int h){//111年統測 double A; A=w*h/2.0; return A; } </pre>	

	<pre> void main () { int a=5,b=3; double result=0; result=Func(a,b); printf("%lf",result); } </pre>	
4	<pre> int f() { //APCS10503 int p = 2; while (p < 2000) { p = 2 * p; } return p; } void main () { int a=f(); printf("%d",a); } </pre>	
5	<pre> int f(int n){ //APCS10503 int p=0; int i=n; while(i>=1){ p=10-2*i; printf("%d",p); i=i-1; } } void main() { f(4); } </pre>	
6	給定函式 F()，執行 F() 時哪一行程式碼可能永遠不會被執行到？ //APCS10603 (A) a = a + 5; (B) a = a + 2; (C) a = 5; (D) 每一行都執行得到 <pre> void F (int a) { while (a < 10) a = a + 5; printf("%d\n",a); } </pre>	

	<pre> if (a < 12) a = a + 2; printf("%d\n",a); if (a <= 11) a = 5; printf("%d\n",a); } int main() { F(1); return 0; } </pre>	
7	<pre> int main(){//APCS10603 int n=5; int i,j; int s=0; for (i=1;i<=n;i++){ for(j=i;j<=n;j++){ s=s+1; } } printf("s=%d",s); return 0; } </pre>	
8	<pre> int f (int a[], int n) { int index = 0; for (int i=1; i<=n-1; i=i+1) { if (a[i] >= a[index]) { index = i; } } return index; } int main() { int a[10]={1,3,9,2,5,8,4,9,6,7}; int k=f(a,10); printf("%d\n",k);// return 0; } </pre>	

9	<pre> int s=1; // 全域變數 void add (int a) { int s=6; //區域變數，其有效範圍僅止於此 //不會回傳影響全域變數 printf("2:a=%d",a); //我增加的 for(; a>=0; a=a-1) { //a用上面 printf("%d,,\n", s); s++; printf("%d,,, \n", s); } } int main () { printf("1:%d, \n", s); add(s); printf("3:%d, \n", s); s=9; printf("4:%d, \n", s); return 0; } </pre>	
10	<pre> int main(){ char d[]="1234",e[]="1243"; int a=0,b=0,i,j; for (i=0;i<=3;i++){ if (d[i]==e[i]) a++; for (j=0;j<=3;j++){ if (d[i]==e[j]) b++; } } b=b-a; printf("%d A %d B ",a,b); } </pre>	



Chapter 12

物件導向程式設計

人類之所以會是萬物之靈，其中一個主要原因是人類可以在錯誤中成長。物件導向的程式設計（Object Oriented Programming, OOP）也是人們在程式語言中逐漸累積的成果，這個觀念在 1970 年代就已提出，只是當時未受到任何程式語言支持。現在，OOP 則已是所有程式語言的標準配備，爲了說明 OOP 大行其道的原因，筆者將程式語言的發展分爲 3 個時期，分別爲非程序導向 (Non-procedure-oriented language)、程序導向 (Procedure-oriented language) 及物件導向 (Object-oriented programming)，三者分別說明如下。

非程序導向

早期的程式語言，並沒有內儲副程式（又稱程式庫）。當開發新的應用程式時，如果某一功能與之前寫過的程式相近，則會將此段已完成的程式整段複製，並稍加修改即可重新加以利用。但是，這些程式的分身包括本尊，自從複製出來以後就開始以各自的方式發展，結果造成各版本的差異越來越大，這些程式很難分辨誰複製誰，彼此之間也難再共用某些程式碼，當遇到錯誤或欲新增功能時，更是很難逐一修改所有程式。

程序導向

爲了解決以上程式共用問題，各編譯器廠商便開始提供一些常用的函式，較有規模的軟體設計公司亦會將常用的函式集中在一個函式庫（Function library），旗下的軟體產品一律呼叫這些標準的函式庫，而不是從函式庫複製出來修改，此即爲程序導向的程式設計。程序導向與非程序導向相比，的確解決了程式共用的問題，但是，人們並不以此爲自滿，有些問題還是不夠順暢。例如：有些函式庫會隨著人們需求的增加而有不同版本，當某些函式功能增加時，只好重新取函式庫的函式加以修改，並賦予新的名稱。如此日積月累，我們的函式庫已有許多函式。這些函式有的功能相近、有的是前後版本不同、有的函式裡的變數來龍去脈

不明，造成使用者的混亂。爲了突破以上瓶頸，於是有物件導向的發展，以解決以上程序導向的不足。

物件導向

程序導向中的函式，存有許多解決問題的函式（函式在目前的物件導向中稱爲方法），它偏重在方法的解決。但是，人類的生活方式不僅是單純的行爲描述，更存在著屬性 (Attributes) 的記載。例如：當在描述一個人時，通常描述如下：「他的名字是洪峻維，身高 172、體重 70，具有滾進、游泳及跑步的能力」；又例如：描述一輛車子時，其描述如：「它的名字是 SENTRA，排氣量是 1600c.c.，耗油量是每公里 0.1 公升，且具有每小時 120 公里的移動能力」。以上人與車即稱爲「物件」(Object)，名字、身高、排氣量則稱爲「屬性」，而滾進、游泳、跑步、移動則稱爲「方法」(Method)。既然真實世界是以物件描述物種，程式設計亦不應侷限在狹隘的函式，僅偏重解決問題的方法，而是應以物件的宏觀角度撰寫程式。所以，基於物件導向的新觀念，程式開發工具即制訂一種新的型態，此型態稱爲類別 (Class)。每一個類別都有屬於自己的屬性與方法，也就是我們已將眾多的函式依照類別存放，如此可解決目前日益龐大的函式命名與函式取用的困擾。例如：程序導向的時代，關於開門的函式即有電梯開門、汽車開門、房子開門等數種開門的方法，如此徒增命名與取用的困擾。但在物件導向的領域中，開門這個方法是附在相對應的類別。例如：於電梯類別有電梯的開門，汽車類別有汽車的開門，房子類別有房子的開門方法，各類別的方法名稱都叫「開門」，撰寫程式時，也是「電梯.開門」，「汽車.開門」，或是「房子.開門」(物件與方法、屬性之間以點(.)運算子連結)，如此既可簡化程式的撰寫，亦可減少程式出錯的機會。但是，在程序導向的領域中，所有的函式都集中，就有可能用錯方法。例如：用開電梯門的方法去開汽車的門，結果當然是錯的。

物件導向的程式設計裡，先定義類別，其次，類別的變數即稱爲物件，物件亦稱爲類別的實現或類別的樣例化 (Instance)。也就是說，類別就像是一個酒瓶或機器人的模型，在這些模型裡面已設計好他所具備的屬性與方法，當您需要酒瓶或機器人時，只要依照這些模型樣例一個或多個酒瓶或機器人即可。其次，物件導向亦提出三個理念，分別是物件的封裝 (Encapsulation)、繼承 (Inheritance) 及多型 (Polymorphism)，以解決程序導向的不足。以上觀念，將在本章說明。

12-1 C++ 程式設計

由於物件導向理念的需求與衝擊，在 1980 年代有物件導向 C++ 語言的誕生，由於 C 語言的執行速度很好且使用率很高，所以 C++ 就繼承 C，且加入物件導向的類別等元素。既然是繼承，表示原來 C 的指令都可以使用。例如：請在 Code::Blocks 新增 C++ 程式，C++ 程式架構如下：

```
#include <iostream>
using namespace std;
int main(){
    cout << "Hello world!" << endl;
    return 0;
}
```

先執行以上程式，若有輸出「"Hello world!"」，請將「cout << "Hello world!" << endl;」刪除，接著，鍵入 2-4a 程式如下，並執行。

```
#include <iostream>
using namespace std;
int main(){
    int a=15,b=8,c;
    c=a*b;
    printf("面積=%d\n",c);
    return 0;
}
```

也可以在 C++ 直接開啓或鍵入範例 2-4a，程式如下，所有 C 語言標頭檔、輸入與指令都可在 C++ 環境執行，此稱為「繼承」。

```
#include <stdio.h>
#include <stdlib.h>
int main(){
    int a=15,b=8,c;
    c=a*b;
    printf("面積=%d\n",c);
    return 0;
}
```

以下程式則以 C++ 指令，重做範例 2-4a 的面積計算。

```
#include <iostream> //載入輸出入標頭檔
using namespace std; //指派使用std命名空間
int main()
{
    int a,b,c;
    cin>>a>>b; //輸入a b
    cout<<a<<endl; //輸出a
    cout<<b<<endl; //輸出b
    c=a*b;
    cout<<c;
    return 0;
}
```

標頭檔

本例因為使用 cout 物件，所以要預先載入其函式庫

```
#include <iostream>
```

C++ 新增的輸出入物件為 cout 與 cin，其函數庫為 iostream 則用來取代

```
#include <stdio.h>
#include <stdlib.h>
```

函式庫的 scanf() 與 printf() 函式。C 語言的函式庫均為 .h，例如：數學函式庫為 math.h。為了方便區分，C++ 新增函式的函式庫，則均沒有 .h，例如：C++ 新增 cmath 用來包含與取代 math.h。

命名空間

using namespace std 為使用命名空間 std，本例若不指定命名空間，也就是不寫 using namespace std; 則每次鍵入 cout、cin 時，都要指定命名空間 std::，程式如下：

```
#include <iostream>
int main() {
    int a;
    std::cout<<"input a:";
```

```
std::cin>>a;
std::cout<<a;
return 0;
}
```

cout

cout 物件可輸出指定任何文數字與變數內容，而且不用去管變數的資料型態。其語法如下：

```
cout <<字串或變數 [<<字串或變數] [<<endl] ;
```

語法中的中括號 "[]" 表示可省略，例如：

```
int a=5;
cout<<a;//5
```

又例如：

```
int a=5;
cout <<"a="<<a<<endl;
```

可輸出 a=5。其中第一個用雙引號表示字串內容，將會以字串方式輸出，第二個 a，則是變數，將會輸出對應變數的內容。endl(endline) 代表此列的輸出結束，跳離此列，往後的輸出移至下一列的開頭。例如：

```
cout <<"Good"<<endl;
cout <<"Good";
```

可輸出：

```
Good
Good
```

但是，

```
cout <<"Good";
cout <<"Good";
```

則輸出：

```
GoodGood
```

範例12-1a

示範輸出以上結果。

👉 程式列印

```
#include <iostream>
using namespace std;
int main() {
    cout <<"Good"<<endl;
    int a=5;
    cout <<"a="<<a<<endl;
    return 0;
}
```

👉 自我練習

1. 請於 C++ 環境完成以上程式。
2. 請嘗試將「using namespace std;」去掉，請問程式要如何修改。

🔗 cin

cin 物件可輸入文數字至指定的變數。其語法如下：

```
cin>>變數 [>>變數] ;
```

例如：

```
char a;
cin >>a;
cout <<"a="<<a<<endl;
```

可輸入字元至變數 a。又例如：

```
int b;
cin >>b;
cout<<"b="<<b<<endl;
```

可輸入整數至變數 b。又例如：

```
char c[5];
cin >>c;
cout<<"c="<<c<<endl;
```

可輸入字串至變數 c。以下程式則可連續輸入 3 個變數，輸入時，變數之間請以空白隔開，例如，輸入 a 5 Horng，則三個變數變數 a,b,c 分別得到 a, 5, Horng。

```
cin >>a>>b>>c;
```

例如，以下是使用 C++ 的 cout、cin 重做範例 7-1c 的程式。

```
#include <iostream>
#include <time.h> // 使用time()
#include <stdlib.h> // 使用rand()
using namespace std;
int main()
{
    srand(time(NULL));
    int a=97+rand()%26;
    cout<<a<<endl;//輸出a
    cout<<(char)a;
    char b;
    cin>>b;//輸入b
    if ((int)b==a)//資料型態相同才能比較
        cout<<"right" //輸出
    else
        cout<<"wrong";
    return 0;
}
```

自我練習

1. 請鍵入以下程式，寫出執行結果。

```
#include <iostream>
using namespace std;
int main() {
    char a;
    cin >>a;
```

```

    cout <<"a="<<a<<endl;
    int b;
    cin >>b;
    cout<<"b="<<b<<endl;
    char c[5];
    cin >>c;
    cout<<"c="<<c<<endl;
    cin >>a>>b>>c;
    cout <<"a="<<a<<endl;
    cout <<"b="<<b<<endl;
    cout <<"c="<<c<<endl;
    return 0;
}

```

題號	題目	結果
	下列 C/C++ 程式片段之敘述，何者正確？（技藝競賽程式組 109） (A)：找出輸入數值最小值 (B)：找出輸入數值最大值 (C)：輸入三個變數 (D)：輸出結果為 the output is:c <pre> int a,b,c; cin>>a; cin>>b; c=a; if (b>c) c=b; cout<<"the output is:"<<c; </pre>	

string 型態

C 語言資料型態並沒有字串，僅使用字元陣列當作字串；C 語言也不能用比較運算子，而是要用比較函式 `strcmp()`。C++ 則新增 `string` 型態，且可以使用「=」當作指派運算子、使用「+」當作字串合併運算子、使用「>,>=,<=,<,<=,<=,<=,<=,<=,<=」當作比較運算子。此即為物件導向的「多型」理念——使用相同的運算子，編譯器在編譯階段，再載入對應的函式，這樣可以減輕開發者的負擔。

自我練習

1. 請鍵入以下程式，並寫出執行結果。

```

#include <iostream>
using namespace std;
int main() {

```

```

string a="aa",b="bb",c;
cout<<a<<endl;
cout<<a[0]<<endl;
c=a+b;//合併、指派運算子
cout<<c<<endl;
cout<<(a>b)<<endl;//比較運算子
return 0;
}

```

2. 請鍵入以下程式，並寫出執行結果。

```

#include <iostream>
using namespace std;
int main() {
    string a[]={"ab","cd","ef"};
    cout<<a[0]<<endl;
    cout<<a[0][0]<<endl;
    cout<<a[1]<<endl;
    return 0;
}

```

3. 以下是字串以一維陣列儲存，然後排序、輸出的操作。C++ 的程式裡，字串可用 string 型態，可用比較運算子(>)，可用字串指派運算子(=)，同範例 11-3a 的字串排序，C++ 程式如下：

```

#include <iostream>
using namespace std;
int main() {
    string a[]={"Hand","Mary","洪峻維","陳惠敏","王一平"};;
    cout<<a[0][0]<<endl;//H
    cout<<a[0][1]<<endl;//a
    cout<<a[1]<<endl;//Mary
    int n=5;
    string t;
    int i,j;
    for(i=0;i<=n-1-1;i++){
        for (j=0;j<=n-2-i;j++){
            if (a[j]>a[j+1]){//字串比較
                t=a[j];//字串複製
                a[j]=a[j+1];
                a[j+1]=t;
            }
        }
    }
}

```

```

        a[j+1]=t;
    }
}
}
for (i=0;i<=n-1;i++)
    cout<<a[i]<<endl;
return 0;
}

```

4. 請寫一程式，可以輸入指定數量 N，然後輸入 N 筆姓名，且排序後輸出。
5. 鍵入以下程式，並寫出執行結果。

```

#include <iostream>//C++
using namespace std;
int main()
{
    string a="ABCD";
    string b[]={"AA1","BB2","CC3","DD4"};
    string c[][2]={{ "AA", "90"}, {"BB", "80"}, {"CC", "70"} };
    string d;
    d=a;
    cout <<a[0]<<endl;//A
    cout <<a<<endl;//ABCD
    cout <<d<<endl;//ABCD
    cout <<b[0][0]<<endl;//A
    cout <<b[0]<<endl;//AA1
    d=b[1];
    cout<<d<<endl;//BB1
    cout <<c[0][0][0]<<endl;//A
    cout<<c[0][0]<<endl;//AA
    d=c[0][1];
    cout<<d<<endl;//90
    return 0;
}

```

6. 以下程式是字串以二維陣列儲存，然後計算總分、計算平均、輸出的操作。

```

#include <iostream>
#include <cstring>//使用C++字串函式
using namespace std;

```



```
int main()
{
    string a[][4]={{"Tom","90","80",""},
                   {"John","60","90",""},
                   {"Mary","80","60",""}};

    cout<<a[0][0][0]<<endl;//T
    cout<<a[0][0][1]<<endl;//o
    cout<<a[1][1][0]<<endl;//6
    cout<<a[0][0]<<endl;//Tom
    cout<<a[0][1]<<endl;//90
    cout<<a[1][0]<<endl;//John
    int i,j;
    //輸出
    for (i=0;i<=2;i++){
        for (j=0;j<=3;j++){
            cout<<a[i][j]<<" ";
        }
        cout<<endl;
    }
    //計算總分、平均
    int sum=0,avg;
    for (i=0;i<=2;i++){
        sum=0;
        for (j=1;j<=2;j++){
            sum=sum+stoi(a[i][j]); //C++轉數值
            //sum=sum+atoi(a[i][j]); //C style轉數值，請看11_3
        }
        avg=sum/2;
        a[i][3]=to_string(avg); //C++轉字串
        //itoa(avg,a[i][3],10); //C style轉字串，請看11_3
    }
    cout<<endl;
    //輸出
    for (i=0;i<=2;i++){
        for (j=0;j<=3;j++){
            cout<<a[i][j]<<" ";
        }
        cout<<endl;
    }
    return 0;
}
```

☞ 函式傳參考呼叫

C 語言有傳值與傳址，請看本書範例 11-5b、11-5c，但是當傳值想改為傳址或傳址想改為傳值，函式裡的變數必須到處修改，造成程式開發者的負擔，C++ 爲了改善此一不便，所以有傳參考。

範例12-1b

示範傳參考呼叫。以傳參考重做範例 11-5b 與 11-5c。

👉 程式列印

現在希望參數是雙向傳遞，若使用傳址，則函式內容必須到處修改變數，但若使用傳參考，則只要改變函式宣告就行。例如，以下程式，分別以傳值、傳址，傳參考的方式重做兩數交換函式，請讀者仔細比較函式原型宣告、主程式與函式寫法的差異。

```
#include <iostream> //C++
using namespace std;
void swap1(int , int ); //傳值
void swap2(int *, int *); //傳址
void swap3(int &, int &); //傳參考
int main() {
    int x=3, y=4;
    swap1(x, y); //傳值
    cout<<"1:x="<<x<<" ,y="<<y<<endl; //3,4
    x=3;y=4;
    swap2(&x,&y); //傳址
    cout<<"2:x="<<x<<" ,y="<<y<<endl; //4,3
    x=3;y=4;
    swap3(x,y); //傳參考，謹函式宣告更動
    //請留意傳參考的主程式也一樣
    cout<<"3:x="<<x<<" ,y="<<y<<endl; //4,3
    return 0;
}
void swap1(int p, int q){ //傳值
    int t;
    t=p; p=q; q=t; //交換
}
void swap2(int *p, int *q){ //傳址
    //請留意，函式內變數要更動
    int t;
```

```

        t=*p; *p=*q; *q=t;//交換
    }
    void swap3(int &p, int &q){ //傳參考，謹函式宣告更動
        //請留意，函式內變數不用更動
        int t;
        t=p; p=q; q=t;//交換
    }

```

以上傳參考呼叫，函式原型宣告、函式呼叫、函式原型如下表：

函式原型宣告	void swap(int &, int &);
函式呼叫	swap(x, y);
函式原型	void swap(int &p, int &q)

✂ 函式多載 (Function overloading)

於函式的設計裡，有時會有兩種以上的函式功能接近，例如只是參數型態或參數個數不同，是否可以使用相同的函式名稱，以減少程式設計者命名的困擾？答案是肯定的。因為物件導向的程式設計均允許程式設計者在參數型態與參數個數不同時，使用相同的函式名稱，這些相同的函式即稱為函式多載。以下我們分別實作 3 個 add 函式，其參數型態與數目均不同，但都共用一個函式名稱，待執行階段即可依參數的型態與數目執行對應的函式。

1. 兩個整數

```

int add(int a1,int b1) {
    int sum=a1+b1;
    return(sum);
}

```

2. 三個整數

```

int add(int a1,int b1,int c1) {
    int sum=a1+b1+c1;
    return(sum);
}

```

3. 兩個整數，一個實數

```
double add(int a1,int b1,double c1) {  
    double sum;  
    sum=a1+b1+c1;  
    return(sum);  
}
```

其次，於主程式撰寫如下：

```
#include <iostream> //載入輸出入標頭檔  
using namespace std; //指派使用std命名空間  
int add(int,int);  
int add(int,int,int);  
double add(int,int,double);  
int main() {  
    int a=3,b=4,e;  
    double c=3.4,d;  
    e=add(a,b);  
    cout<<e<<endl;  
    e=add(a,b,b);  
    cout<<e<<endl;  
    d=add(a,b,c);  
    cout<<d<<endl;  
    return 0;  
}  
int add(int a1,int b1) {  
    int sum=a1+b1;  
    return(sum);  
}  
int add(int a1,int b1,int c1) {  
    int sum=a1+b1+c1;  
    return(sum);  
}  
double add(int a1,int b1,double c1) {  
    double sum;  
    sum=a1+b1+c1;  
    return(sum);  
}
```

 補充說明

C 語言就無函式多載性質。例如：傳回整數的絕對值函數「int abs()」與傳回浮點數的絕對值函數「double fabs()」就不同，請複習 11_1 節。但是 C++ 的 abs() 函式就無此問題了，因為 abs(-3)=-3，abs(-3.4)=3.4。

 自我練習

題號	題目	結果
	<pre>int area(int m){ int n=m*m; return(n); } double area(double m){ double n=m*m*3; return(n); } int area(int m,int n){ int p=m*n; return(p); } int main() { int a=2,b=3,c; double d=2,e; c=area(a); cout<<c<<endl; e=area(d); cout<<e<<endl; c=area(a,b); cout<<c<<endl; return 0; }</pre>	

 12-2 類別

真實世界是以物件描述物種，程式設計亦不應侷限在狹隘解決問題的函式，而是應以物件的宏觀角度撰寫程式。所以，基於物件導向的新觀念，程式開發工具即製訂一種新的型態，此型態稱為類別。每一類別都有屬於自己的屬性、函式。其次，下一節還要介紹繼承，讓程式設計者能繼承舊有類別開發新類別。物件導向的程式設計步驟分別是設計類別、樣例類別（類別的樣例就是物件）、使用物件。分別說明如下：

✎ 步驟一：設計類別

類別成員有屬性、函式。類別的簡易語法如下：

```
class 類別名稱 {
    private:
        屬性成員;
        函式成員;
    protected:
        屬性成員;
        函式成員;
    public :
        屬性成員;
        函式成員;
};
```

基於資料封裝等級，類別成員分為三級，分別是 `private`、`protected` 及 `public`。成員若是 `private`，則該成員僅能在類別內存取；`public` 則是公開的，類別與類別物件皆可存取。`protected` 則是折衷的，可供類別、類別物件與衍生類別與衍生類別物件存取。例如：以下一個簡易的類別實作，可判斷成績是否及格，屬性成員有兩個，分別是 `score`、`result`，函式成員有一個 `Check()`，其封裝等級都是公開的，也就是類別內，與類別物件都可存取。

```
class Pass {
    public :
        int score;
        char result ='F';
        void Check() {
            result='F';
            if (score>= 60)
                result='P';
        }
};
```

✎ 步驟二：樣例類別

類別的變數即稱為物件，物件亦稱為類別的實現或類別的樣例化 (Instance)。也就是說，類別就像是一個酒瓶或機器人的模型，在這些模型裏面已設計好它所具備的屬性與函式，當您需要酒瓶或機器人時，只要依照這些模型樣例一個或多個酒瓶或機器人即可。類別樣例的簡易語法如下：

```
類別名稱 物件名稱;
```

例如，以下程式 pa 樣例 Pass：

```
Pass pa;
```

與變數的宣告相同，其實就是類別變數，只是類別的變數稱為物件了。

☛ 步驟三：使用物件。

使用物件的簡易語法是

```
物件.成員；
```

但僅限於封裝等級是 protected 與 public，因為 private 僅限類別內存取。例如，以下程式可設定 score 成員。

```
pa.score=66;
```

以下程式可評判其及格與否。

```
pa.Check();
```

以下程式可取的評判結果。

```
char a=pa.result;
```

範例12-2a

示範以上程式。

☞ 執行結果

```
88 is P
33 is F
```

☞ 程式列印

```
#include <iostream>
using namespace std;
class Pass {
public :
    int score;
    char result ='F';
```

```

        void Check() {
            result='F';
            if (score>= 60)
                result='P';
        }
    };
int main() {
    Pass pa;
    pa.score=88;
    pa.Check();
    cout <<pa.score <<" is "<<pa.result<<endl;
    pa.score=33;
    pa.Check();
    cout <<pa.score <<" is "<<pa.result<<endl;
    return 0;
}

```

👉 自我練習

1. 請設計一個類別，此類別有四個方法，分別是兩個整數相加、相減、相乘、相除。請問您至少要有哪些資料成員？
2. 請設計一個類別，此類別有一個方法，可處理自來水水費計算的計算。假設自來水費率如下：
 - (1) 100 度以下，每度 3 元。
 - (2) 超過 100 度的部分，每度 5 元。
 請問您至少要有哪些資料成員？

※🔒 資料封裝

軟體科技的開發與其它的工業，如汽車、電視、收音機等機械、電子產品相較，可說是起步較晚的領域，在汽車、電視及收音機等產品上，我們發現這些產品很重視物件的封裝。例如，以電視機而言，我們發現內部有許多零件與開關，對於電視製造商而言，它們使用機殼將這些零件與開關 " 封裝 " 起來，才能避免使用者錯誤使用，只留下部份必要的按鈕(如開關鈕或選台鈕)與螢幕讓使用者欣賞節目。軟體程式的開發何嘗不應如此呢？所以對於類別的規劃，我們亦應重視所有函式、欄位及屬性封裝，使得這些成員有不同的封裝等級，以避免主程式與類別庫之間的干擾。就如同電視機或汽車的一些開關與旋鈕，有些是開放給一般的使用者調整，有的是讓維修工程師調整，有些則是出廠調校後，永遠不讓任何人調整。

基於資料的安全性，有些必須先檢查其是否安全與有效，再放入，此時就要透過函式的存取，先給予檢驗再放入。例如，上例的 `score`，我們必須先檢查其成績範圍是否正確，才能進行下一步的評判，所以我們先宣告其資料封裝等級為 `private`，並且使用 `setScore()` 函數輸入資料，且先檢查其範圍是否正確，若正確再放入 `score`，請看以下程式：

```
class Pass {
    private :
        int score;
        char result ='F';
    public:
        void setScore(int a){
            if (a>=0 && a<=100)
                score=a;
            else
                cout <<a <<"is error";
        }
        int getScore(){
            return(score);
        }
        char getResult(){
            return (result);
        }
        void Check()    {
            result='F';
            if (score>= 60)
                result='P';
        }
};
```

其次，既然 `score` 與 `result` 已經封裝為 `private`，那就無法用物件直接存取，也就是您無法使用 `pa.score=88` 設定成績，而必須使用 `public` 的 `getScore()` 與 `getResult()` 取得 `private` 的 `score` 與 `result` 資料。

範例12-2b

示範以上程式。

 執行結果

```
88 is P
33 is F
133 is error
33 is F
```

 程式列印

```
#include <iostream>
using namespace std;
class Pass {
    private :
        int score;
        char result ='F';
    public:
        void setScore(int a){
            if (a>=0 && a<=100)
                score=a;
            else
                cout <<a <<"is error"<<endl;
        }
        int getScore(){
            return(score);
        }
        char getResult(){
            return (result);
        }
        void Check() {
            result='F';
            if (score>= 60)
                result='P';
        }
};

int main() {
    Pass pa;
    pa.setScore(88);
    pa.Check();
}
```

```

    cout <<pa.getScore() <<" is "<<pa.getResult()<<endl;
    pa.setScore(33);
    pa.Check();
    cout <<pa.getScore() <<" is "<<pa.getResult()<<endl;
    pa.setScore(133);
    pa.Check();
    cout <<pa.getScore() <<" is "<<pa.getResult()<<endl;
    return 0;
}

```

自我練習

1. 請將上一範自我練習 1 的兩數相除改為私有成員，且當除數不為 0 時，才可相除。
2. 請將上一範例的實習的用水度數改為私有成員，當使用者輸入使用度數時，先檢查是否為正數或小於 200，若為正數且小於 200，才可放入類別的私有成員，並計算其水費。
3. 小文設計了一個計算矩形面積的類別 CalculateArea，其宣告如下，則下列哪一個成員函式（Member Function）定義正確？（統測 112）

```

1  class CalculateArea{
2  private:
3      double Length, Width, Area;
4  public:
5      void SetPara(double, double);
6      double GetLength();
7      double GetWidth();
8      double GetArea();
9  };

```

- (A) void CalculateArea:: SetPara(double L, double W){Length = L; Width = W;}
- (B) double GetLength() {return Length;}
- (C) CalculateArea:: GetWidth() {return Width;}
- (D) double CalculateArea:: GetArea() { Area = Length * Width;}

解：

- (1) 我們教材是將函式成員實作放在類別內，此考題則放在類別外，類別外的語法是「類別名稱 :: 成員名稱」，「::」為範疇預算子，全部程式如下：

```
#include <iostream>
using namespace std;
// 宣告類別
class CalculateArea{
// 宣告 private 成員，private僅能在類別內存取
private:
    double Length, Width, Area;
// 宣告 public 成員，public才可在主程式存取
public:
    void SetPara(double, double);
    double GetLength();
    double GetWidth();
    double GetArea();
};
// 實作類別成員函數
void CalculateArea::SetPara(double L, double W){
    Length = L; Width = W;
}
double CalculateArea::GetLength( ){ //題目缺底線
    return Length;
}
double CalculateArea:: GetWidth( ){ //題目缺底線
    return Width;
}
double CalculateArea:: GetArea( ){//題目缺底線
    Area = Length * Width;
    return(Area);
}
// 程式執行的 main()
int main(void) {
    // 宣告並建立 CalculateArea 型態的物件 a
    CalculateArea a;
    a.SetPara(2,3);//指派私有成員Length = L; Width = W;
    cout << a.GetArea() << endl;
    cout << endl;
    return 0;
}
```

(2) 以下則將類別實作放在類別內，這樣就與教材相同。

```
#include <iostream>
using namespace std;
// 宣告類別
class CalculateArea{
private:
    double Length, Width, Area;
// 宣告 public 成員
public:
    void SetPara(double L, double W){
        Length = L; Width = W;
    }
    double GetLength(){
        return Length;
    }
    double GetWidth(){
        return Width;
    };
    double GetArea(){
        Area = Length * Width;
        return(Area);
    }
}; //類別結束要分號
// 程式執行的 main()
int main(void) {
    // 宣告並建立 CalculateArea 型態的物件 a
    CalculateArea a;
    a.SetPara(2,3);
    cout << a.GetArea() << endl;
    cout << endl;
    return 0;
}
```

(3) 自我練習。請練習撰寫建構子。

(4) 自我練習。請撰寫一個新類別，繼承此類別，但增加計算立方體體積、表面積的方法。

4. 在沒有使用存取修飾字 (private 、 public 或 protected) 的情況下，下列敘述何者正確？ (統測 112)

- (A) class 內的所有成員變數都可在 main () 主程式內存取
- (B) struct 宣告的物件不可以做為 class 的成員
- (C) 任何函式內的 struct 型態變數中的成員，在該函式的範圍內都可以被存取
- (D) class 內的成員函式不可存取同一個 class 內的成員變數

解：

- (A) 沒有修飾字 (private 、 public 或 protected)，都是 private，private 僅能在類別內存取，無法給 main() 主程式存取，請鍵入以下程式。

```
#include <iostream>
using namespace std;
// 宣告類別
class CalculateArea{
//沒有修飾字 ( private、public 或 protected )，都是private
    double Length, Width, Area;
    void SetPara(double L, double W){
        Length = L; Width = W;
    }
}; //類別結束要分號
// 程式執行的 main()
int main(void) {
    // 宣告並建立 CalculateArea 型態的物件 a
    CalculateArea a;
    a.Length=2; // //無法存取
    a.SetPara(2,3); //無法存取
    cout << a.GetArea() << endl;
    cout << endl;
    return 0;
}
```

- (B) 可以，請鍵入以下程式。

```
#include <iostream>
using namespace std;
//宣告結構
struct rect{
    double Length, Width, Area;
};
```

```
// 宣告類別
class CalculateArea{
    struct rect a;
public:
    void SetPara(double L, double W){
        a.Length = L; a.Width = W;
    }
    double GetLength(){
        return a.Length;
    }
    double GetWidth(){
        return a.Width;
    };
    double GetArea(){
        a.Area = a.Length * a.Width;
        return(a.Area);
    }
}; //類別結束要分號

int main(void) {
    // 宣告並建立 CalculateArea 型態的物件 a
    CalculateArea a;
    a.SetPara(2,3);
    cout << a.GetArea() << endl;
    cout << endl;
    return 0;
}
```

(C) 正確 (D) 可以

※✎ 建構子 (Constructor)

建立變數的同時可以設定初值，例如，`int a=2;` 那建立物件的同時是否可以設定初值呢？答案是肯定的，因為類別有一函式成員非常特殊，此一函式成員名稱與類別名稱相同，此稱為建構子。C++ 對於建構子的規定如下：

1. 建構子是特殊的函式，函數名稱必須與類別名稱相同，但不能有傳回值。
2. 上一節提到函數可以有多載 (Overloading)，那建構子可以嗎？答案也是肯定的，因為建構子也是函式。例如，我們於上例 Psas 類別，新增兩個 Pass 函式如下：

```

class Pass {
    private :
        int score;
        char result ='F';
    public:
        Pass(int a){ //函式名稱與類別名稱相同
            if (a>=0 && a<=100)
                score=a;
            else
                cout <<a <<"is error";
        }
        Pass(){
            score=0;
        }
        void setScore(int a){
            if (a>=0 && a<=100)
                score=a;
            else
                cout <<a <<"is error"<<endl;
        }
        int getScore(){
            return(score);
        }
        char getResult(){
            return (result);
        }
        void Check() {
            result='F';
            if (score>= 60)
                result='P';
        }
};

```

主程式撰寫如下：

```

int main() {
    Pass pa1(33); //建立函式一併設定初值，且使用Pass(int)建構子
    pa1.Check();
    cout <<pa1.getScore() <<" is "<<pa1.getResult()<<endl;
}

```



```

    Pass pa2; //建立函式未設定初值，則使用Pass()建構子，預設初值為0分
    pa2.Check();
    cout <<pa2.getScore() <<" is " <<pa2.getResult() <<endl;

    Pass pa3; //建立函式未設定初值，則使用Pass()建構子，預設初值為0分
    pa3.setScore(88);
    pa3.Check();
    cout <<pa3.getScore() <<" is " <<pa3.getResult() <<endl;

    return 0;
}

```

範例12-2c

示範以上程式。

👉 執行結果

```

33 is F
0 is F
88 is P

```

👉 自我練習

1. 請將上一範例的自我練習加上建構子。

※ 解構子 (Destructor)

C++ 爲了提高記憶體使用效率，還有解構子的設計，可以將建構子所佔用記憶體刪除。此解構子的名稱亦與類別名稱相同，但是在名稱前加上波浪符號 (~)。例如：~Pass(){}，則可於物件使用完畢，先行歸還建構子所佔用記憶體。C++ 對於解構子的規定如下：

1. 解構子的名稱亦與類別名稱相同，但是在名稱前加上波浪符號 (~)。
2. 解構子不可以有傳回值。
3. 解構子不得有參數。例如：上面 Pass 類別新增解構子如下：

```
~Pass(){} //解構子, 解構子不需有任何程式，系統會自動刪除已使用的記憶體
```

則於物件使用完畢，可執行此解構子如下：

```
pa2.~Pass();
```

則系統將會自動刪除已使用的記憶體。

範例12-2d

示範以上程式。

```
#include <iostream>
using namespace std;
class Pass {
    private :
        int score;
        char result ='F';
    public:
        Pass(int a){ //函式名稱與類別名稱相同
            if (a>=0 && a<=100)
                score=a;
            else
                cout <<a <<"is error";
        }
        Pass(){
            score=0;
        }
        ~Pass(){};
        void setScore(int a){
            if (a>=0 && a<=100)
                score=a;
            else
                cout <<a <<"is error";
        }
        int getScore(){
            return(score);
        }
        char getResult(){
            return (result);
        }
        void Check() {
```

```

        result='F';
        if (score>= 60)
            result='P';
    }
};

int main() {
    Pass pa1(33); //建立函式一併設定初值，且使用Pass(int)建構子
    pa1.Check();
    cout <<pa1.getScore() <<" is "<<pa1.getResult()<<endl;

    Pass pa2; //建立函式未設定初值，則使用Pass()建構子，預設初值為0分
    pa2.Check();
    cout <<pa2.getScore() <<" is "<<pa2.getResult()<<endl;

    Pass pa3; //建立函式未設定初值，則使用Pass()建構子，預設初值為0分
    pa3.setScore(88);
    pa3.Check();
    cout <<pa3.getScore() <<" is "<<pa3.getResult()<<endl;
    pa2.~Pass();
    return 0;
}

```

自我練習

1. 請將上一範例的自我練習加上建構子與解構子。

※ 靜態 (static) 成員

類別成員前加上 static，則稱此成員為靜態成員，靜態成員可不用樣例而直接使用。例如，於前面 Pass 類別新增資料靜態成員 d，與函式成員 Checkgrade()，此兩成員前均加上 static 如下，則此兩成員均是靜態成員。

```

class Pass {
public:
    static const int d=60;
    static char Checkgrade(int c){
        if (c>=d)
            return 'P';
        else

```

```

        return 'F';
    }
};

```

其次，靜態成員於時前，不用事先樣例，直接使用範圍運算子 (::) 就可直接使用。例如：

```

int main() {
    char a;
    int b;
    b=Pass::d;    //直接使用類別資料成員d
    cout <<b<<endl;

    a=Pass::Checkgrade(88); //直接使用類別函式成員
    cout <<a<<endl;

    a=Pass::Checkgrade(22);    //直接使用類別函式成員
    cout <<a<<endl;
    return 0;
}

```

範例12-2e

示範以上程式。

執行結果

```

60
P
F

```

程式列印

```

#include <iostream>
using namespace std;
class Pass {
public:
    static const int d=60;
    static char Checkgrade(int c){
        if (c>=d)
            return 'P';

```

```

        else
            return 'F';
    }
};

int main() {
    char a;
    int b;
    b=Pass::d;    //直接使用類別資料成員d
    cout <<b<<endl;

    a=Pass::Checkgrade(88); //直接使用類別函式成員
    cout <<a<<endl;

    a=Pass::Checkgrade(22);    //直接使用類別函式成員
    cout <<a<<endl;
    return 0;
}

```

👉 自我練習

1. 請設計一個計算面積的類別，此類別有常數 π ，及正方形、長方形、圓面積計算函式，且均可不用樣例此類別，即可使用該常數或該函式。

※ 12-3 繼承

🔗 繼承(Inherit)

任何新產品的開發，均不是無中生有，而是從舊有的產品中繼承某些特性，再加入新的零件或修改部份零件而成一項新的產品。例如，SENTRA180 正是繼承 NEW SENTRA 而來，只是排氣量提高了、內裝豪華了，但是輪胎、方向盤及座椅還是用原來 SENTRA 的東西，這就是繼承的道理，使得新產品的開發得以縮短時程。軟體的開發何嘗不應如此？繼承的另一優點是同一函式得以讓數個新舊版本同時存在。因為當你有新產品時，你不可能同時讓你的新舊客戶同時更新，所以您必須讓這些不同的版本同時存在，以滿足不同的產品需求，就像是您修車更換零件時，師傅總會強調這是哪一版本的汽車，以便決定訂購哪一版本的零件。C++ 的類別繼承簡要語法如下：

```
class 衍生類別 : 繼承方式 基礎類別{;
```

繼承方式有 public、protected 及 private，此三種繼承對基礎類別的影響如下表：

基礎類別成員封裝方式	繼承方法	衍生類別成員封裝方式
private	public	無法存取
protected	public	protected
public	public	public
private	protected	無法存取
protected	protected	protected
public	protected	protected
private	private	無法存取
protected	private	private
public	private	private

例如，以下程式，衍生類別 Pass2 繼承基礎類別 Pass，繼承的方式為 public，原來基礎類別的資料封裝方式 private 改為無法存取，protected 及 public 不變。

```
class Pass2:public Pass{
};
```

所以主程式樣例 Pass2 物件之後，即可使用基礎類別的所有成員。例如：

```
Pass2 pa;
pa.setScore(66);
pa.Check();
cout <<pa.getScore() <<" is "<<pa.getResult()<<endl;
```

範例12-3a

示範以上類別繼承。

執行結果

```
66 is P
33 is F
```

👉 程式列印

```
#include <iostream>
using namespace std;
class Pass {
    protected :
        int score;
        char result ='F';
    public:
        Pass(int a){ //函式名稱與類別名稱相同
            if (a>=0 && a<=100)
                score=a;
            else
                cout <<a <<"is error";
        }
        Pass(){
            score=0;
        }
        void setScore(int a){
            if (a>=0 && a<=100)
                score=a;
            else
                cout <<a <<"is error";
        }
        int getScore(){
            return(score);
        }
        char getResult(){
            return (result);
        }
        void Check() {
            result='F';
            if (score>= 60)
                result='P';
        }
};

class Pass2:public Pass{
};

int main() {
    Pass2 pa;
```

```

    pa.setScore(66);
    pa.Check();
    cout <<pa.getScore() <<" is "<<pa.getResult()<<endl;
    pa.setScore(33);
    pa.Check();
    cout <<pa.getScore() <<" is "<<pa.getResult()<<endl;
    return 0;
}

```

自我練習

1. 同範例 12-2a 自我練習 1，請設計一個類別繼承此類別，且新增一個計算兩數相除的餘數的方法。
2. 請設計一個類別，此類別繼承上一單元的水費計算，但新增計算電費的方法。電費假設未超過 50 度，基本費 100 元，超過時每度 3 元。

※ 多型

多型 (Polymorphism) 有些書譯成同名異式，它的原文是希臘文，意思是說，一個名稱有多種表現的方式，目的是滿足資料的抽象化 (Data Abstraction)。例如，你有一個僕人專門幫你開門，那麼不論這門是內推，外拉或向旁邊推，你都是下同一指令「開門」，然後你的僕人即會依照門的結構而完成開門的動作。以上的重點是，使用者不用去描述細節，編譯器會依照實際的狀態，解決問題。物件導向的程式設計亦發揚此理念，讓程式設計師於程式設計階段使用相同的指令，而編譯器能於執行階段依據不同的需求，執行不同的程式片段，此即為多型。C++ 物件導向的多型表現，常用的分別是多載 (Overloading) 與改寫 (Overriding)，函式的多載已經於 12-1 節介紹，函式改寫說明如下：

函式改寫

函式改寫 (Function overriding) 有些書翻譯成函式覆載，也就是說衍生類別可繼續使用原基礎類別的函式名稱，但可以修改或賦予新的功能，因為名稱相同，這樣才不會影響使用者的程式撰寫習慣。例如，原來 Pass 類別的及格標準是 60 分，現在及格標準是 70 分了，所以衍生類別可以改寫 Check 方法如下：

```

class Pass2:public Pass{
public :
    void Check()    {
        result='F';
    }
}

```



```

        if (score >= 70)
            result = 'P';
    }

};

```

其次，新舊客戶都不影響其程式撰寫習慣，新客戶就樣例衍生類別 `Pass2`，其及格標準是 70 分；舊客戶繼續樣例 `Pass`，其及格標準是 60。請看以下範例。

範例12-3b

示範以上函式改寫。

執行結果

```

66 is F
72 is P
66 is P

```

程式列印

```

#include <iostream>
using namespace std;
class Pass {
    //同上一範例12-3a
};

class Pass2:public Pass{
public :
    void Check() {
        result='F';
        if (score >= 70)
            result='P';
    }
};

int main() {
    Pass2 pa;
    pa.setScore(66);
    pa.Check();
    cout <<pa.getScore() <<" is " <<pa.getResult() <<endl;
    pa.setScore(72);
    pa.Check();
}

```

```

        cout <<pa.getScore() <<" is "<<pa.getResult()<<endl;

        Pass pb;
        pb.setScore(66);
        pb.Check();
        cout <<pb.getScore() <<" is "<<pb.getResult()<<endl;
        return 0;
    }

```

👉 自我練習

1. 同上一範例自我練習 1，但加法改寫為 $c=a+2b$ 。
2. 同上一範例，但是水費計價方式請調為
 - (1) 100 度以下，每度 4 元。
 - (2) 超過 100 度的部分，每度 6 元。

※ 建構子與繼承

基礎類別的所有資料與函式成員皆可被繼承，但是建構子與解構子並沒有被繼承，所以衍生類別的建構子必須自行撰寫。以下範例示範撰寫衍生類別 Pass2 的建構子。

範例12-3c

示範撰寫衍生類別 Pass2 的建構子。

👉 執行結果

```

66 is F
72 is P
66 is P

```

👉 程式列印

```

#include <iostream>
using namespace std;
class Pass {
    protected :
        int score;
        char result ='F';

```

```
public:
    Pass(int a){ //函式名稱與類別名稱相同
        if (a>=0 && a<=100)
            score=a;
        else
            cout <<a <<"is error";
    }
    Pass(){
        score=0;
    }

    void setScore(int a){
        if (a>=0 && a<=100)
            score=a;
        else
            cout <<a <<"is error";
    }
    int getScore(){
        return(score);
    }
    char getResult(){
        return (result);
    }
    void Check() {
        result='F';
        if (score>= 60)
            result='P';
    }
};

class Pass2:public Pass{
public :
    Pass2(int a){ //函式名稱與類別名稱相同
        if (a>=0 && a<=100)
            score=a;
        else
            cout <<a <<"is error";
    }
    Pass2(){
        score=0;
    }
};
```

```

    }
    void Check() {
        result='F';
        if (score>= 70)
            result='P';
    }
};

int main() {
    Pass2 pa(66);
    pa.Check();
    cout <<pa.getScore() <<" is " <<pa.getResult()<<endl;

    Pass2 pb(72);
    pb.Check();
    cout <<pb.getScore() <<" is " <<pb.getResult()<<endl;

    Pass pc(66);
    pc.Check();
    cout <<pc.getScore() <<" is " <<pc.getResult()<<endl;
    return 0;
}

```

自我練習

1. 同上一範例自我練習，但請增加建構子。

※ 多重繼承 (Multiple inheritance)

如果爸爸有錢，媽媽很漂亮，通常生出既漂亮又有錢的兒女，那程式設計是否可以如此多重繼承呢？答案是亦是肯定的。多重繼承的語法如下：

```
class 衍生類別:繼承方式,基礎類別1,[ 繼承方式,基礎類別2,][...]
```

繼承方式同單一繼承的 private、protected、public，繼承效力也同單一繼承。例如，類別 Fish 如下，有游泳能力。

```

class Fish{
    public:
        void swimming(){

```

```

        cout<<"I can swimming"<<endl;
    }
};

```

類別 Bird 如下，有飛翔能力。

```

class Bird{
public:
    void flying(){
        cout<<"I can flying"<<endl;
    }
};

```

使用多重繼承如下：

```

class Super:public Fish,public Bird{};

```

衍生類別 Super 即同時擁有游泳與飛翔能力，例如，主程式不用吹灰之力，不用撰寫任何程式，即可同時使用 Fish 類別的 swimming() 函式與 Bird 類別的 Flying() 函式，請看以下範例。

```

Super su;
su.swimming();
su.flying() ;

```

範例12-3d

示範多重繼承。

執行結果

```

I can swimming
I can flying

```

程式列印

```

#include <iostream>
using namespace std;
class Fish{
public:
    void swimming(){

```

```

        cout<<"I can swimming"<<endl;
    }
};
class Bird{
public:
    void flying(){
        cout<<"I can flying"<<endl;
    }
};
class Super:public Fish,public Bird{
};
int main() {
    Super su;
    su.swimming();
    su.flying() ;
    return 0;
}

```

自我練習

題號	題目	結果
1	<pre> class Operadd { public : int a; int b; int c; void add() { c=a+b; } }; class Opersub { public : int d; int e; int f; void sub() { f=d-e; } }; </pre>	

```

class Oper3:public Operadd,public Opersub{
public:
    void mul() {
        c=a*b;
    }
};

int main() {
    Oper3 op1;
    op1.a=4;op1.b=3;
    op1.add();
    cout<<op1.c<<endl;

    op1.d=4;op1.e=3;
    op1.sub();
    cout<<op1.f<<endl;

    op1.a=4;op1.b=3;
    op1.mul();
    cout<<op1.c<<endl;
}

```

2. 關於物件導向程式語言的敘述，下列何者錯誤？（111 年統測）
- (A) 物件導向程式語言具有「多型」、「繼承」與「封裝」的特性
 - (B) 物件導向程式語言的「多型」可以達到資訊隱藏的目的（資料的抽象化）
 - (C) 「繼承」的特性讓物件導向程式語言具有避免重複撰寫相同程式碼的優點
 - (D) 物件導向程式語言可以簡化大型程式的開發流程

12-4 應用實例

☞ 複數運算

複數的標準式為 $a+bi$ ，若有兩以標準式表示的複數 $P=a+bi$ 與 $Q=c+di$ ，則兩複數的加減乘除運算如下：

- (1) $P+Q=(a+c)+(b+d)i$
- (2) $P-Q=(a-c)+(b-d)i$
- (3) $P*Q=(ac-bd)+(ad+bc)i$
- (4) $P/Q=(ac+bd)+(bc-ad)i/(a^2+d^2)$

範例12-4a

請開發一個類別，可以執行以上運算，並輸出結果。

 執行結果

```
1+2i
2+3i
3+5i
```

 程式列印

```
#include <iostream>
using namespace std;
class Complex{
    private:
        int real,img;
    public:
        Complex(){}
        Complex(int m,int n){
            real=m;
            img=n;
        }
        void Add( Complex a, Complex b){
            real=a.real+b.real;
            img=a.img+b.img;
        }
        void Data(){
            cout <<real<<"+"<<img<<"i"<<endl;
        }
};

int main() {
    Complex a(1,2);
    a.Data();
    Complex b(2,3);
    b.Data();
    Complex c;
    c.Add(a,b);
    c.Data();
    return 0;
}
```


自行練習

1. 請自行完成以上範例的減法、乘除法。

※ 運算子改寫

以上的加法運算是採用函式完成，例如

```
c.Add(a,b);
```

那是否可如同我們平常的撰寫習慣，寫成

```
c=a+b;
```

這樣的話，就是讓使用者不用進行任何調整，即可使用舊有書寫習慣。現在要讓程式語言有智慧，去判斷此加法(+)是實數還是複數的加法運算，實數就採用實數的加法規則，複數就採用複數的加法規則。反正使用者只要下達加法指令就好了，電腦就會依照當時的情況執行相加，此即為物件導向的多型表現，就如同本章開頭說的，您只要下達開門就好，電腦就會依照門的構造，執行外推，內推或外拉等開門動作。運算子改寫簡要語法如下：

```
friend 傳回型態 operator運算子(參數列){
    return 傳回資料;
}
```

以上的傳回型態、參數列、傳回資料均與函式相同。但是類別的私有資料成員僅有類別的函式成員可存取，所以要將此運算子成員宣告為 friend(意思為伙伴或朋友)。請看以下範例說明。

範例12-4b

示範運算子改寫。

執行結果

```
1+2i
2+3i
3+5i
11
```

 程式列印

1. 改寫加法運算子如下：

```
friend Complex operator+( Complex a, Complex b){
    Complex c;
    c.real=a.real+b.real;
    c.img=a.img+b.img;
    return c;
}
```

則主程式即可使用：

```
Complex a(1,2);
Complex b(2,3);
Complex c;
c=a+b;
c.Data(); // 3+5i
```

其次，實數的加法亦使用相同運算子如下，電腦會自動依照其型態執行正確的運算。

```
int d=3,e=8;
int f=d+e;
cout<<f;    //11
```

2. 全部程式列印如下：

```
#include <iostream>
using namespace std;
//class Complex;
class Complex{
private:
    int real,img;
public:
    Complex(){}
    Complex(int m,int n){
        real=m;
        img=n;
    }
    void Add( Complex a, Complex b){
        real=a.real+b.real;
```

```
        img=a.img+b.img;
    }
    friend Complex operator+( Complex a, Complex b){
        Complex c;
        c.real=a.real+b.real;
        c.img=a.img+b.img;
        return c;
    }
    void Data(){
        cout <<real<<"+"<<img<<"i"<<endl;
    }
};

int main() {
    Complex a(1,2);
    a.Data();
    Complex b(2,3);
    b.Data();
    Complex c;
    c=a+b;//3+5i
    c.Data();
    int d=3,e=8;
    int f=d+e;//11
    cout<<f<<endl;
    return 0;
}
```

課後習題

題號	題目	結果
1	<pre>using namespace std; class Polar{ private: int c,r; public: Polar(){} Polar(int m,int n){ c=m; r=n; } void mul(Polar a, Polar b){ c=a.c*b.c; r=a.r+b.r; } void print(){ cout <<c<<"∠"<<r<<endl; } }; int main() { Polar a(3,30); a.print(); Polar b(4,45); b.print(); Polar c; c.mul(a,b); c.print(); return 0; }</pre>	

```
2 using namespace std;
class Polar{
    private:
        int c,r;
    public:
        Polar(){}
        Polar(int m,int n){
            c=m;
            r=n;
        }
        void mul( Polar a, Polar b){
            c=a.c*b.c;
            r=a.r+b.r;
        }
        friend Polar operator*(Polar a,Polar b){
            Polar d;
            d.c=a.c*b.c;
            d.r=a.r+b.r;
        }
    return d;
    }
    void print(){
        cout <<c<<"\n"<<r<<endl;
    }
};

int main() {
    Polar a(3,30);
    a.print();
    Polar b(4,45);
    b.print();
    Polar c;
    c.mul(a,b); //使用函數
    c.print();
    c=a*b; //使用運算子
    c.print();
    int e=3,f=5;
    int g=e*f;
    cout<<g<<endl;
    return 0;
}
```

