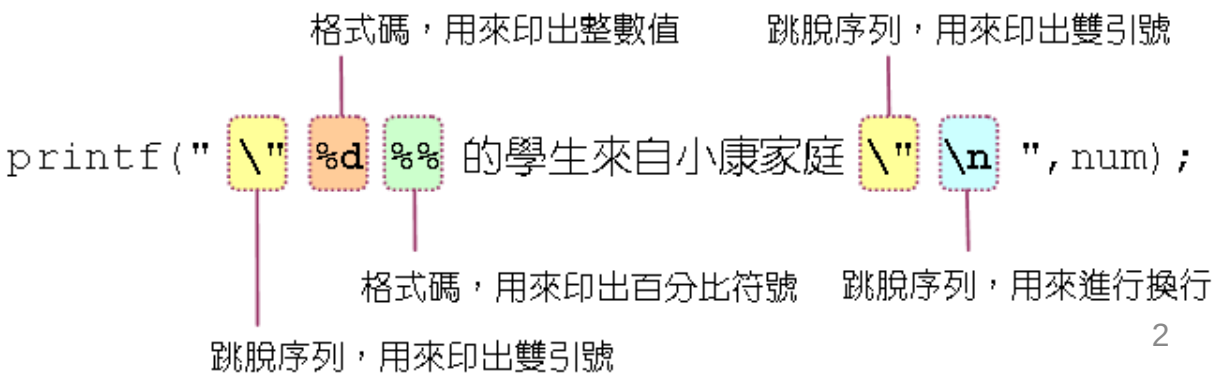
C 程式語言 2

跳脫序列與格式碼的應用

- 下面的程式碼是利用格式碼印出字串：
- 要印出「%」符號， 可用格式碼「 %% 」

```
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
  int num=25;
  printf("\"%d%%的學生來自小康家庭\"\n",num);   /* 印出字串 */

  system("pause");
  return 0;
}
```



認識運算式

- C語言的運算式可以執行運算來產生所需的值，運算式可以簡單到只有一個常數值或變數，也可能複雜到由多個運算子和運算元組成。

- C語言的「運算式」（Expression）是由一序列「運算子」（Operators）和「運算元」（Operands）所組成，

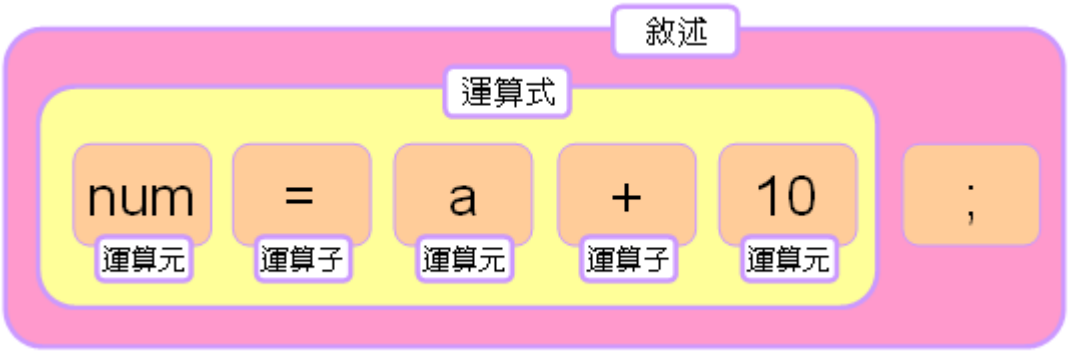
  例如：2個變數相加的加法運算式，如下所示：

  a + b

- 上述運算式告訴電腦計算2個運算元a和b相加的和，換句話說，我們可以在程式碼撰寫運算式來執行程式所需的運算任務。

運算式、運算元與運算子

- 運算式由運算元與運算子組成
  - 運算式：expression
  - 運算元：operand，如變數sum，或常數10等
  - 運算子：operator，如「+」、「-」、「*」與「/」等符號



運算式與運算子

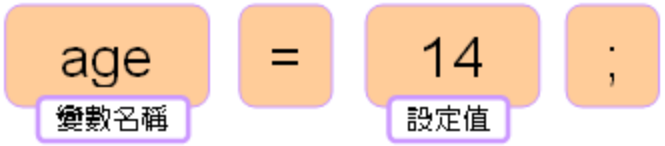
- C 所提供的運算子可分
  - 算數運算子
    - x = a + b; // + 運算子
    - y = c * d; // * 運算子
  - 邏輯運算子
    - a = x && y; //* && 為 AND 運算子的符號
    - b = w || z ; // || 為 OR 運算子的符號
  - 關係運算子
    - x >= y // >= 代表大於等於
    - z == a // == 代表判斷關係上的等於

設定運算子

- 「設定」運算子可將變數設值

設定運算子	意義	範例	說明
=	設定	a=5	設定 a 的值等於 5

- 等號（=）是「設定」的意思，如下面的範例：



一元運算子

- 一元運算子（unary operator）只需要一個運算元

```
– +3;     /* 表示正3，3 為運算元 */
– -a;     /* 表示負a ，a 為運算元 */
– !a;     /* NOT運算，若a為0，則!a為1，若a不為0，則!a為0*/
```

一元運算子	意義	範例	說明
+	正號	a=+5	同 a=5，相當於設定 a 等於正 5
-	負號	a=-3	設定 a 等於-3
!	NOT，否	a=!b	把 b 的值取 NOT，再設給 a 存放

算數運算子

- 算數運算子的成員如下：

算數運算子	意義	範例	說明
+	加法	2+4	計算 2+4
-	減法	3-6	計算 3-6
*	乘法	7*9	計算 7*9
/	除法	6.4/3	計算 6.4/3
%	取餘數	21%9	計算 21 除以 9 的餘數

餘數運算子

- 下面的範例是餘數運算子「%」的練習
- 要印出「%」符號， 用格式碼「 %% 」

```
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
  printf("12%%4=%d\n",12%4);      /* 求出12/4的餘數 */
  printf("12%%5=%d\n",12%5);      /* 求出12/5的餘數 */
  printf("12%%16=%d\n",12%16);   /* 求出12/16的餘數 */

  system("pause");
  return 0;
}
```

關係運算子與 if 敘述

- if 敘述與關係運算子

if 敘述的格式

```
if(判斷條件)
    敘述主體;
```

關係運算子	意義	範例	說明
>	大於	a>b	判別 a 是否大於 b
<	小於	a<b	判別 a 是否小於 b
>=	大於等於	a>=b	判別 a 是否大於等於 b
<=	小於等於	a<=b	判別 a 是否小於等於 b
==	等於	a==b	判別 a 是否等於 b
!=	不等於	a!=b	判別 a 是否不等於 b

遞增與遞減運算子

- 遞增與遞減運算子的成員：

遞增與遞減運算子	意義	範例	說明
++	遞增，變數值加 1	a++	a 加 1 後再設定給 a 存放
--	遞減，變數值減 1	a--	a 減 1 後再設定給 a 存放

- a++ 會先執行整個敘述後，再將 a 的值加 1
- ++a 則是先把 a 的值加 1 後，再執行整個敘述

遞增與遞減運算子

- 下面的程式是使用遞增運算子的範例：

```
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
  int a=3, b=3;
  printf("a=%d",a);
  printf(", a++的傳回值為%d",a++);     /* 計算a++，並印出其傳回值 */
  printf(", a=%d\n",a);
  printf("b=%d",b);
  printf(", ++b的傳回值為%d",++b);     /* 計算++b，並印出其傳回值 */
  printf(", b=%d\n",b);

  system("pause");
  return 0;
}
```

邏輯運算子

- AND、OR與真值表

邏輯運算子	意義	範例	說明
&&	AND，且	a&&b	計算 a AND b 的結果
||	OR，或	a||b	計算 a OR b 的結果

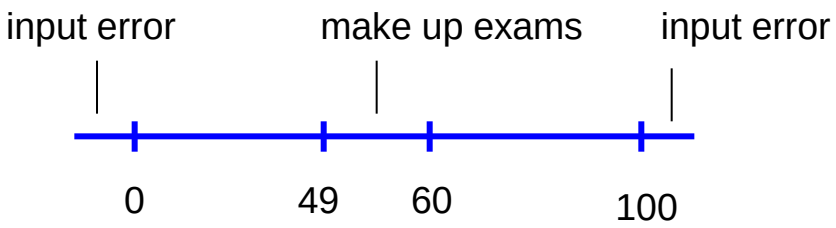
AND 及 OR 真值表

AND	T	F
T	T	F
F	F	F

OR	T	F
T	T	T
F	T	F

邏輯運算子

- 邏輯運算子的應用範例：




```
#include <stdio.h>
#include <stdlib.h>
int main(void) {
  int score;
  printf("Please input a score:");
  scanf("%d",&score);

  if ((score<0) || (score>100))   /* 若成績超出0到100之間 */
    printf("input error!!\n");

  if ((score<60) && (score>49))   /* 若成績介於50到59之間 */
    printf("make up exams!!\n");

  system("pause");
  return 0;
}
```

括號運算子

- 括號運算子「()」用來提高運算式的優先順序

括號運算子	意義
()	提高括號中運算式的優先順序

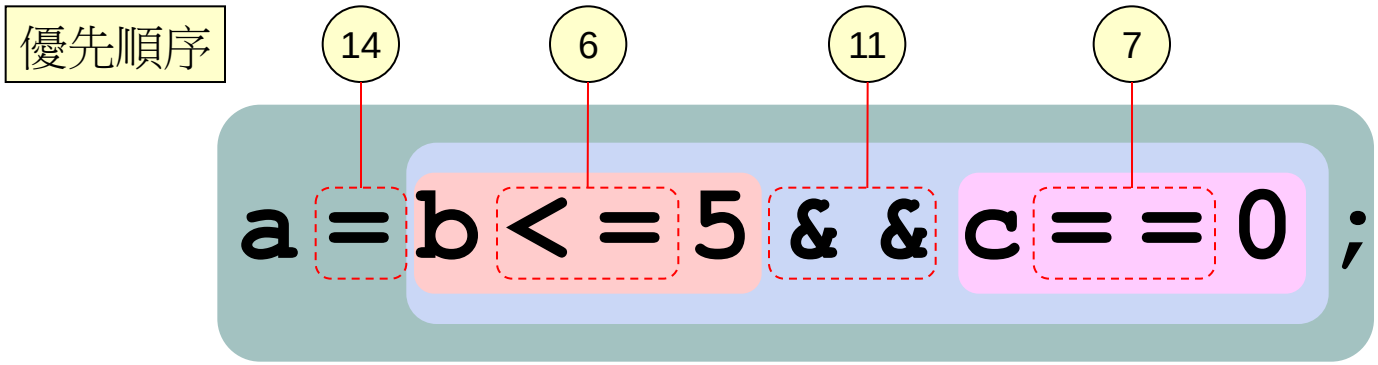
- `3+5*4*6-7;` /* 未加括號的運算式 */
- `(3+5*4)*(6-7);` /* 加上括號的運算式 */

運算子的優先順序

優先順序	運算子	類別	結合性
1	()	括號運算子	由左至右
1	[]	方括號運算子	由左至右
2	!、+ （正號）、- （負號）	一元運算子	由右至左
2	~	位元邏輯運算子	由右至左
2	++、--	遞增與遞減運算子	由右至左
3	*、/、%	算數運算子	由左至右
4	+、-	算數運算子	由左至右
5	<<、>>	位元左移、右移運算子	由左至右
6	>、>=、<、<=	關係運算子	由左至右
7	==、!=	關係運算子	由左至右
8	& （位元運算的 AND）	位元邏輯運算子	由左至右
9	^ （位元運算的 XOR）	位元邏輯運算子	由左至右
10	| （位元運算的 OR）	位元邏輯運算子	由左至右
11	&&	邏輯運算子	由左至右
12	||	邏輯運算子	由左至右
13	?:	條件運算子	由右至左
14	=	設定運算子	由右至左

運算子的優先順序

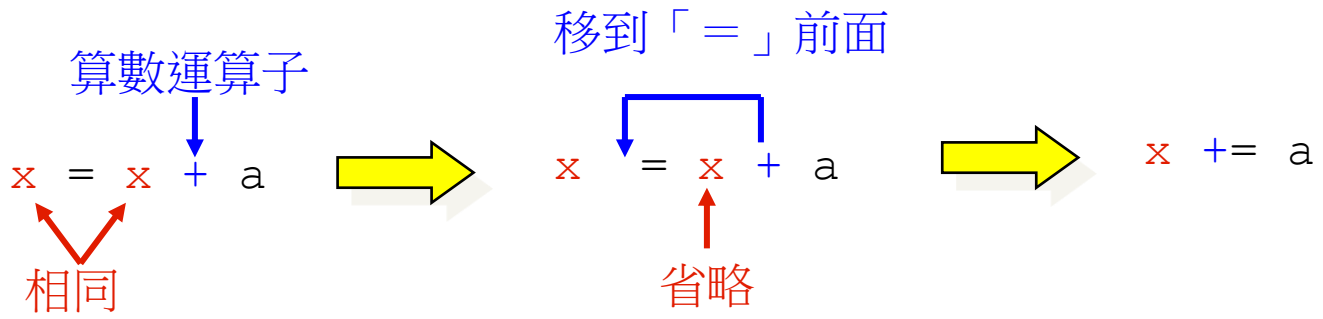
- 運算子優先順序的範例：



1. 先計算 b<=5 (<=的優先順序為6)
2. 再計算 c==0 (==的優先順序為7)
3. 然後進行&&運算 (&&的優先順序為11)
4. 最後再把運算結果設給變數 a 存放 (= 的優先順序為14)

運算式與簡潔運算子

- 算數運算子與指定運算子「=」的簡便寫法
  - 例如：**x = x + a 可以寫成 x += a**



運算式與簡潔運算子

運算式 為運算子與運算元組成

- 簡潔運算子可簡化運算式

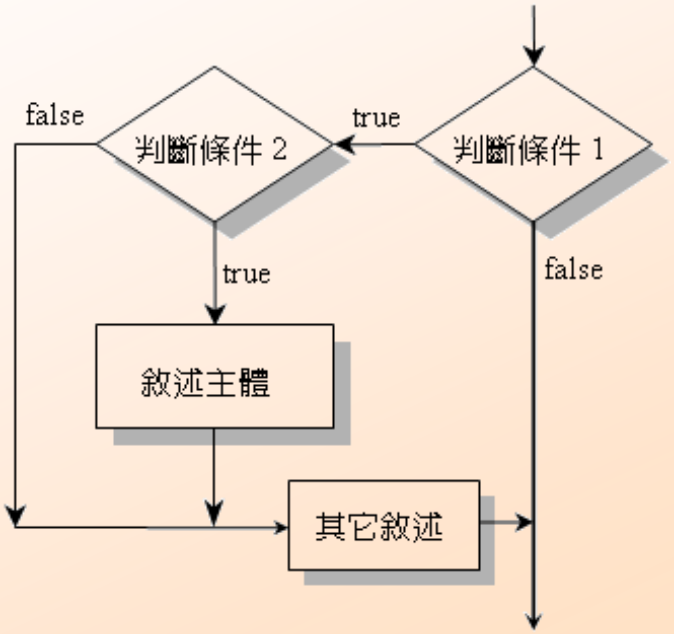
運算子	範例用法	說明	意義
+=	a+=b	a+b 的值存放到 a 中	a=a+b
-=	a-=b	a-b 的值存放到 a 中	a=a-b
*=	a*=b	a*b 的值存放到 a 中	a=a*b
/=	a/=b	a/b 的值存放到 a 中	a=a/b
%=	a%=b	a%b 的值存放到 a 中	a=a%b

巢狀 if 敘述

- if 裡面還有其它的if 敘述，則稱為巢狀 if 敘述

巢狀 if 敘述的語法

```
if(判斷條件1)
{
    if(判斷條件2)
    {
        敘述主體;
    }
    ...
    其它敘述;
}
```



巢狀 if 敘述的練習

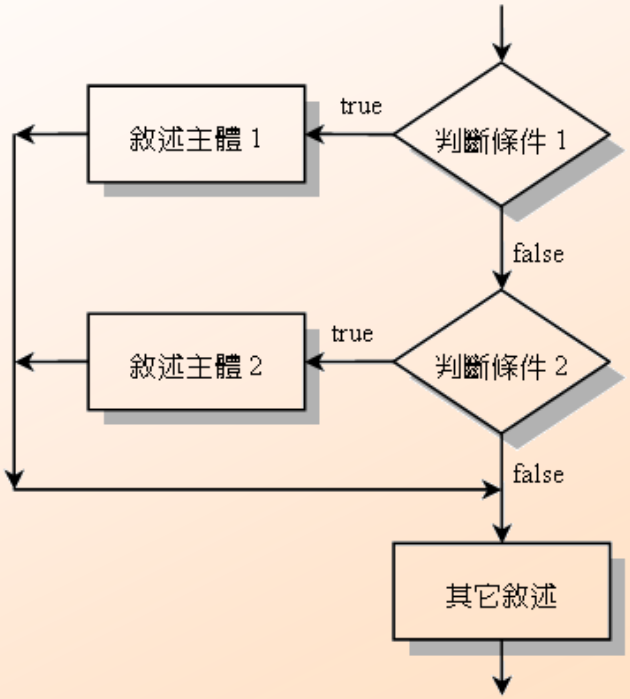
```
#include <stdio.h>
#include <stdlib.h>
int main(void) {
  int score;
  printf(" Please input a score: ");
  scanf("%d",&score);

  if (score<60)                                  /* 如果score<60 */
  {
    if(score>=50)                                /* 如果score>=50 */
      printf(" make up exams \n");
    else
      printf(“Fail\n");
  }
  else
    printf(“Pass\n");

  system("pause");
  return 0;
}
```

使用 if-else-if 敘述

- if-else-if：當 if 判斷不成立，必須進行其它判斷時

if-else-if 敘述的語法

```
if(判斷條件1)
{
    敘述主體1;
}
else if(判斷條件2)
{
    敘述主體2;
}
```



if-else-if 敘述的應用

```
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
  int score;
  printf("Your score:");
  scanf("%d",&score);
  if (score>=80)
    printf("%d is A\n",score);              /* 印出A */
  else if (score>=70)
        printf("%d is B\n",score);          /* 印出B */
      else if (score>=60)
            printf("%d is C\n",score);      /* 印出C */
          else
            printf("Failed!!\n");           /* 印出字串"Failed!!" */

  system("pause");
  return 0;
}
```

if 與 else 的配對問題

- else會與離它最近的 if 配對：

```
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
  int num;
  printf("Please input an integer: ");
  scanf("%d",&num);

  if (num>=0)
    if(num<=10)
      printf("number between 0-10 \n");
    else
      printf("number greater than 10 \n");

  system("pause");
  return 0;
}
```

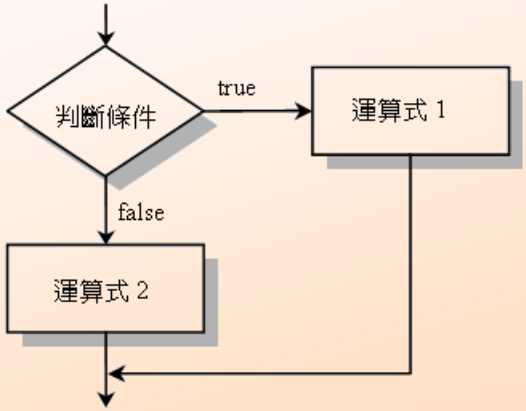
條件運算子

- 條件運算子可代替簡單的 if-else 敘述

條件運算子	意義
?:	根據條件的成立與否，來決定結果為?或:後的運算式

條件運算子的語法

條件判斷 ? 運算式1 : 運算式2



條件運算子的範例

- 利用條件運算子判斷兩個數中較大的數：

```
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
  int num1,num2,larger;
  printf("Please input two integers: :");
  scanf("%d %d",&num1,&num2);

  num1>num2 ? (larger=num1) : (larger=num2);  /* 條件運算子 */
  printf("%d greater value \n",larger);

  system("pause");
  return 0;
}
```

switch 敘述的範例

- 依據選擇值進行四則運算：

```
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
  float a,b;
  char oper;
  printf("Please input the expression:(ex:3+2):");        /* 輸入運算式 */
  scanf("%d %c %d",&a,&oper,&b);

  switch(oper)
  {
    case '+':
      printf("%d+%d=%d\n",a,b,a+b);     /* 印出a+b */
      break;
```

switch 敘述的範例

```
 case '-':
        printf("%d-%d=%d\n",a,b,a-b);                    /* 印出a-b */
        break;
     case '*':
        printf("%d*%d=%d\n",a,b,a*b);                    /* 印出a*b */
        break;
     case '/':
        printf("%d/%d=%.3f\n",a,b,a/b); /* 印出a%b */
        break;
     default:
        printf(“input error!!\n");                       /* 印出字串 */
   }
   system("pause");
   return 0;
}
```

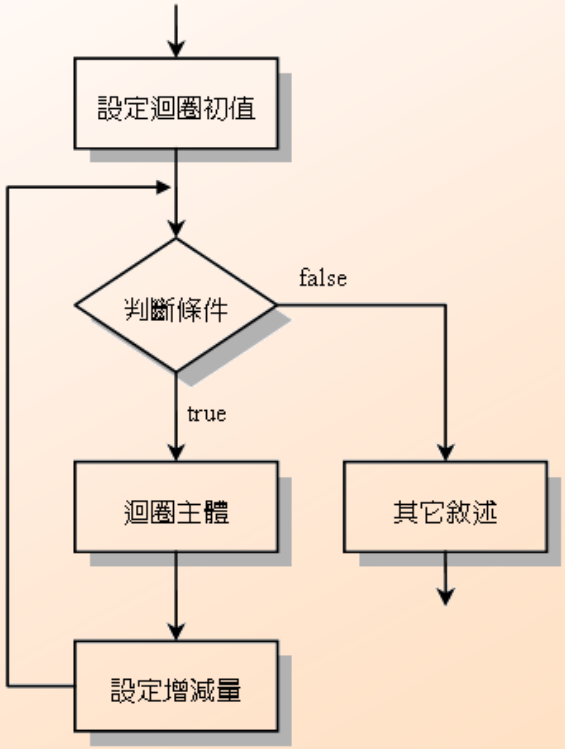
使用while 迴圈

- while 最適合用在迴圈執行次數為未知時

while 迴圈的語法

```
設定迴圈初值;
while(判斷條件)
{
    迴圈主體;
    設定增減量;
}
```

這兒不可以加分號



while迴圈的範例

- 將while用在迴圈執行次數為未知：

```
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
  int i=1,sum=0;      /* 設定迴圈初值 */
  while(i<=100)    /* while迴圈，當sum小於100則繼續累加 */
  {
    sum+=i;
    printf(" From 1 to accumulate %2d=%2d\n",i,sum);
    i++;
  }
  printf(" It must be added to the %d\n",i-1);
  system("pause");
  return 0;
}
```

以巢狀while迴圈改寫九九乘法表

```
#include <stdio.h>
#include <stdlib.h>
int main(void)  {
  int i=1, j=1;                    /* 設定迴圈控制變數的初值 */
  while (i<=9)                     /* 外層迴圈 */
  {
    while (j<=9)                   /* 內層迴圈 */
    {
      printf("%d*%d=%2d  ",i,j,i*j);
      j++;
    }
    printf("\n");
    i++;
    j=1;
  }
  system("pause");
  return 0;
}
```

-The End-