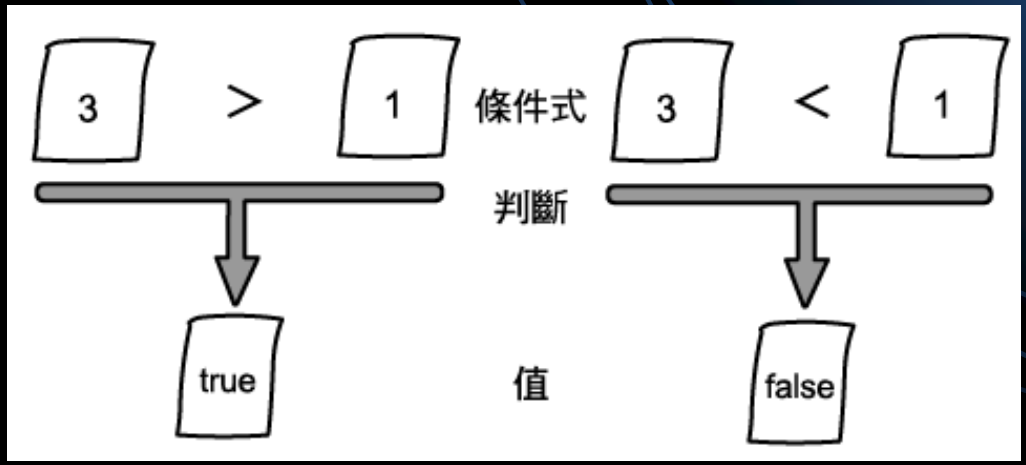
條件處理

5-1 關係運算子與條件式

- 關於條件式
  - 如果學校的考試考得好的話…
    →就和朋友去旅行
  - 如果考得不好的話…
    →就繼續努力用功
  - C語言爲了要配合人類各種不同狀況的需要，使用條件式（condition）來解決這些問題。
  - 關係運算子最後判斷的結果只有2種可能：
    - 真（true）
    - 假（false）

- 條件式的表示法
  - 3 > 1, 3 < 1
  - 「true」／「false」



關係運算子　條件式的結果要等於true，前提是…

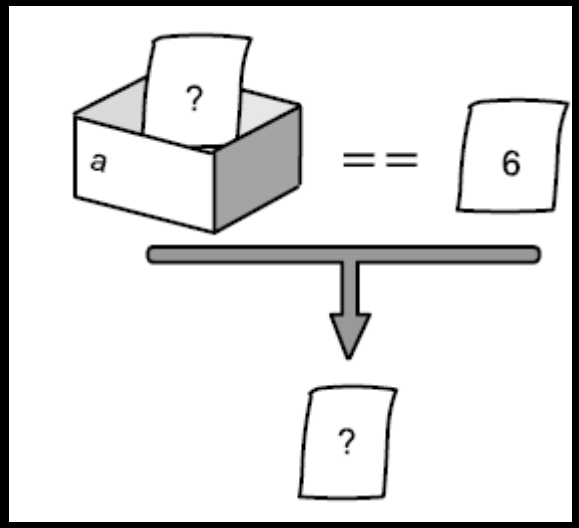
==　　右邊的值等於左邊的值

!=　　右邊的值不等於左邊的值

>　　左邊的值大於右邊的值

- 關係運算子的使用
  - 5 > 3 ← 此條件式的結果爲**true**
  - 5 < 3 ← 此條件式的結果爲**false**
  - a == 6
  - a != 6

  ← 此條件式的結果會隨變數**a**的值而改變



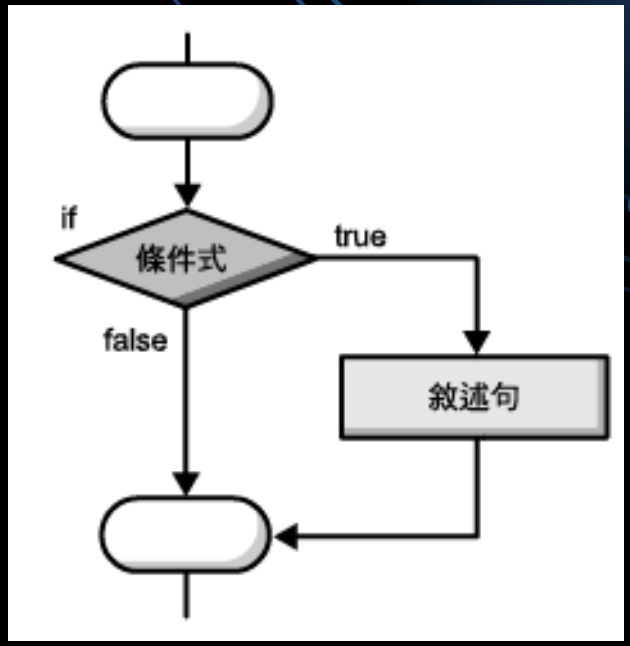
- 請不要弄錯「=」（指定運算子）和「==」（關係運算子）喔。

5-2 IF敘述

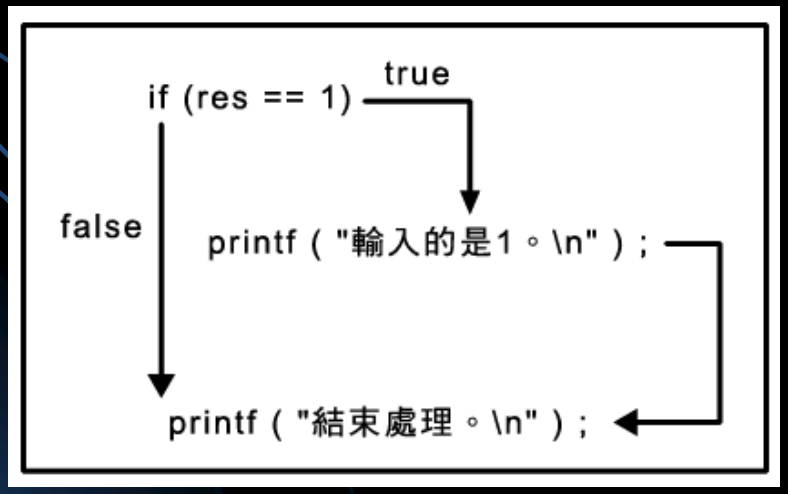
- 語法：

```
if (條件式)
    其他敘述句;
```

  - 唯有當條件式爲true時，才會執行被指定的敘述。



- 範例：



Sample1.c ▶ 使用 if 敘述

```
#include <stdio.h>

int main(void)
{
    int res;

    printf("請輸入整數。\n");

    scanf("%d", &res);

    if (res == 1)
        printf("輸入的是 1 。\n");

    printf("結束處理。\n");

    return 0;
}
```

①將使用者從鍵盤輸入的值指定給變數 res

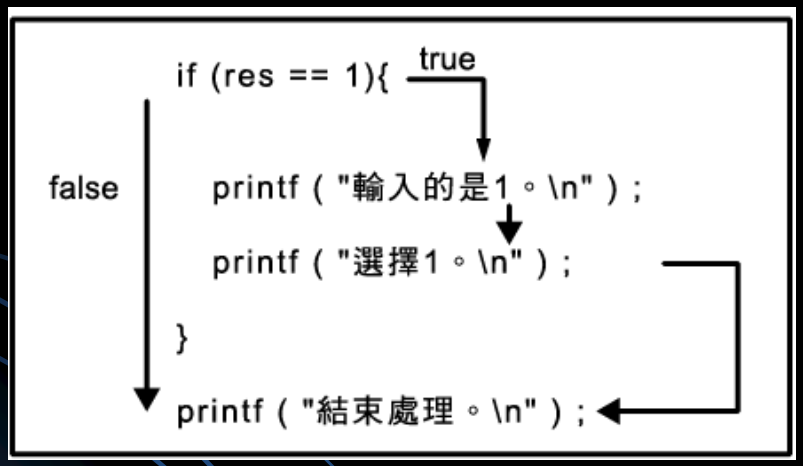
②如果輸入值等於 1 ，則此處的條件式結果為 true

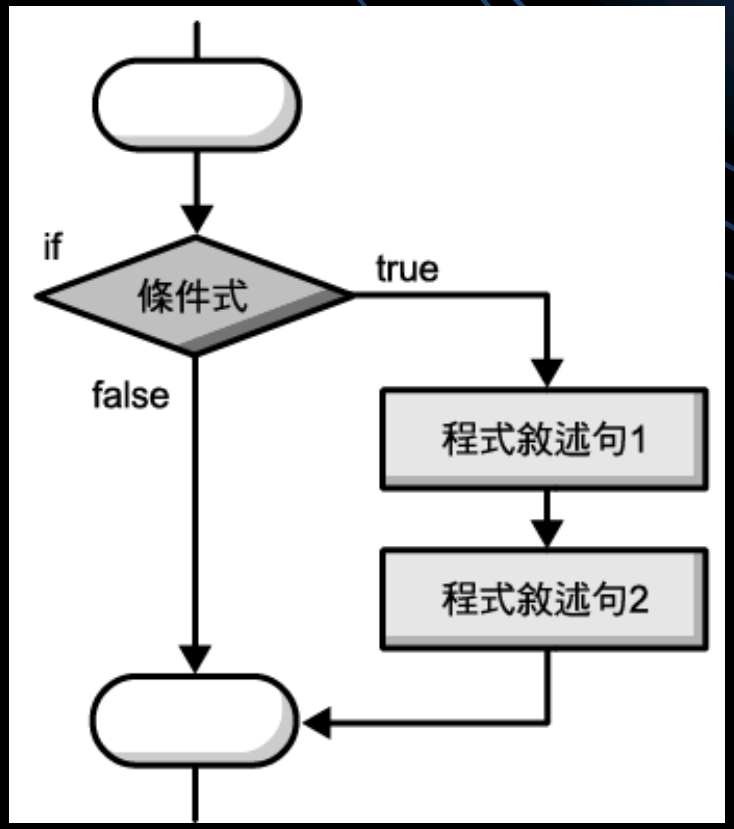
③於是就會執行 if 內部的敘述

- if敘述中有2行以上的敘述句時的語法：

```
if (條件式) {
   程式敘述句1 ;
   程式敘述句2 ;
   …
}
```

- 範例：





- 萬一忘記加上{ }，則只會執行第一行敘述句，之後的敘述句將不屬於if敘述…

Sample2.c ▶ 可以執行多行程式敘述句的 if 敘述

```
#include <stdio.h>

int main(void)
{
    int res;

    printf("請輸入整數。\n");

    scanf("%d", &res);

    if (res == 1){
        printf("輸入的是 1 。\n");
        printf("選擇 1 。\n");
    }

    printf("結束處理。\n");

    return 0;
}
```

如果輸入 1 的話（結果為 true）

依序執行程式區塊內的處理

```c
#include <stdio.h>

int main(void)
{
    int res;

    printf("請輸入整數。\n");

    scanf("%d", &res);

    if (res == 1)
        printf("輸入的是1。\n");
        printf("選擇1。\n");

    printf("結束處理。\n");

    return 0;
}
```

只有這一行程式（①）屬於 if 敘述的一部分

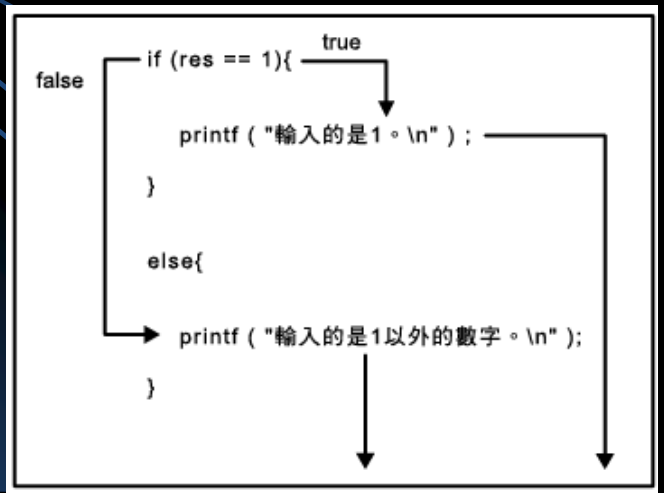
這一行程式（②）已經不屬於 if 敘述的一部分

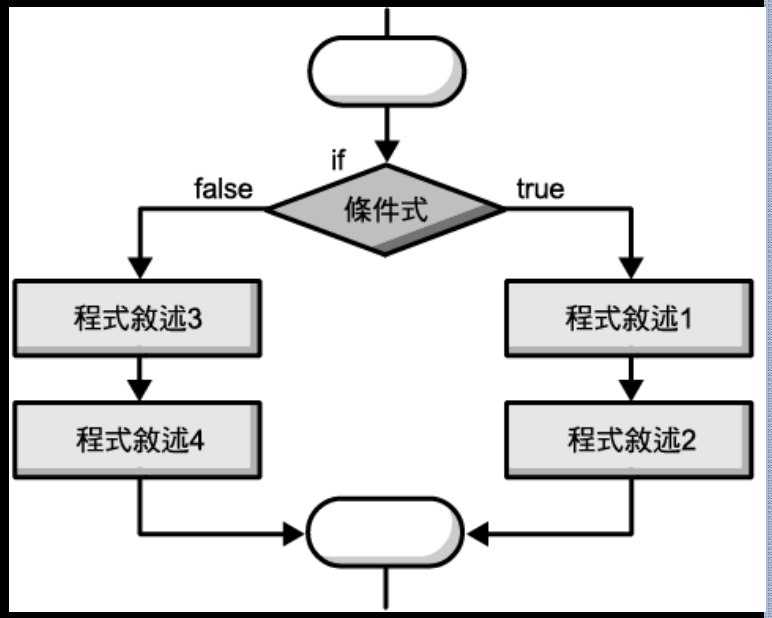
5-3 IF～ELSE敘述

- 語法：

```
if (條件式)
        程式敘述1;
else
        程式敘述2;
```

- 條件式不成立時便會執行else之後的敘述句。
- 範例：





Sample3.c ▶ 使用 if ～ else 敘述

```
#include <stdio.h>

int main(void)
{
    int res;

    printf("請輸入整數。\n");

    scanf("%d", &res);

    if (res == 1){
        printf("輸入的是 1 。\n");
    }
    else{
        printf("輸入的是 1 以外的數字。\n");
    }

    return 0;
}
```

只要使用者輸入 1（條件式為 true），就會執行這個部分

如果使用者輸入 1 以外的其他值（條件式為 false），則執行這個部分

5-4 IF～ELSE IF～ELSE

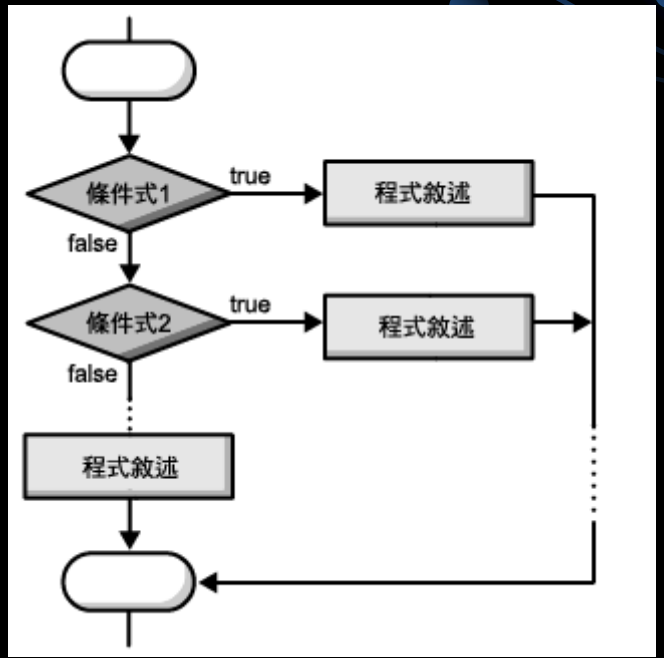
- 語法：

```
if～else if～else敘述
if (條件式1) {
   程式敘述1;
   程式敘述2;
   …
}
else if (條件式2) {
   程式敘述3;
   程式敘述4;
   …
}
else {
    …
}
```

條件式1爲true
便執行這部份

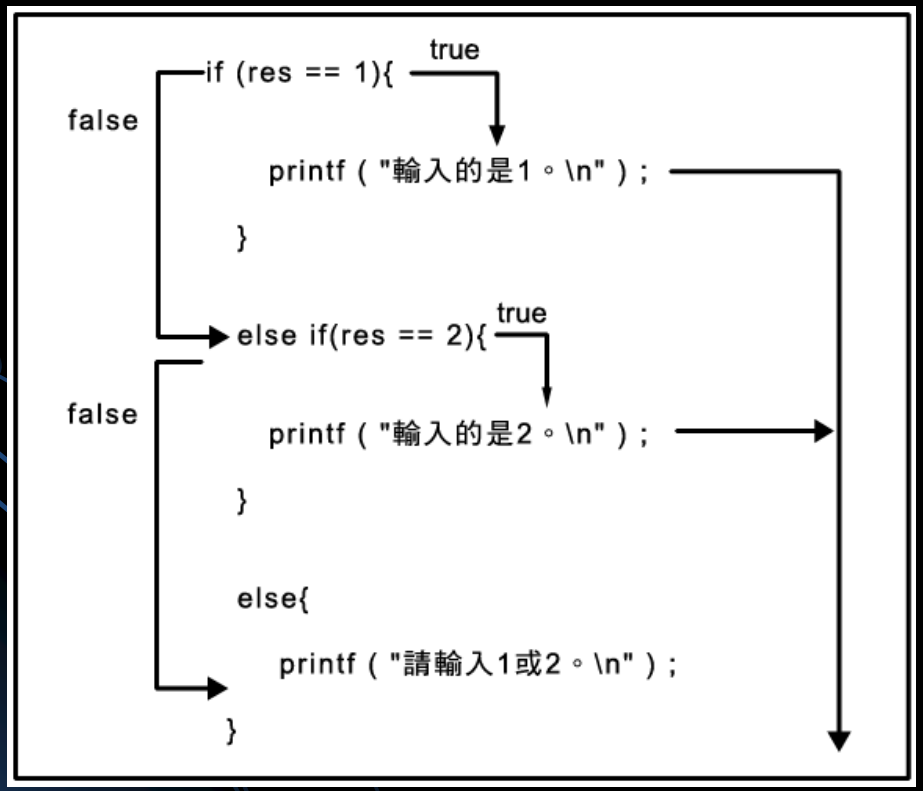
條件式1爲false且
條件式2爲true
便執行這部份

條件式1、2均爲false
便執行這部份



- if~else if~else敘述可以一次處理多個條件式。

- 若是確定一定會有條件式符合時，也可以省略最後的else。
- 範例：



Sample4.c ▶ 使用 if ～ else if ～ else 敘述

```
#include <stdio.h>

int main(void)
{
    int res;

    printf("請輸入整數。\n");

    scanf("%d", &res);

    if (res == 1){
        printf("輸入的是1。\n");
    }
    else if(res == 2){
        printf("輸入的是2。\n");
    }
    else{
        printf("請輸入1或2。\n");
    }

    return 0;
}
```

①使用者輸入 1 時會執行這個部分

②輸入 2 時會執行這個部分

③輸入 1 或 2 以外的其他數字時會執行這個部分

5-5 SWITCH敘述

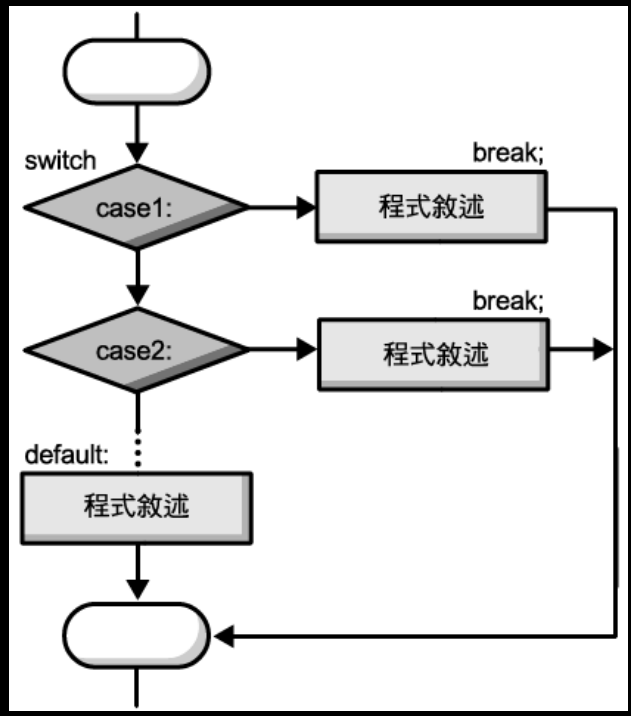
- 語法：

```
…
switch (運算式) {
  case 常數1:
  程式敘述1;
  …
  break;
  case 常數2:
  程式敘述2;
  …
  break;
  default:
  程式敘述D;
  …
  break;
}
```

如果運算式判斷後等於常數1時，會執行這一部份

如果運算式判斷後等於常數2時，會執行這一部分

如果運算式判斷後不等於常數1和常數2時，則執行這一部份



- 在switch敘述中，如果括號內之運算式的值與任何一個case後面的值一致的話，就會執行該case下面的程式碼，直到碰上break指令爲止。若是沒有和運算式的值相符合的case的話，就會直接跳到「default:」的部份執行。
- switch敘述比if~else if~else敘述更易於使用。
- break指令可以強制中斷程式的執行流程，必須注意break指令在switch敘述中的位置，否則將會影響程式的執行。

- 範例：

```
…
switch(res){
   case 1:
      printf("輸入的是1。\n");
      break;
   case 2:
      printf("輸入的是2。\n");
      break;
   default:
      printf("請輸入1或2。\n");
      break;
}
```

輸入**1**的時候執行這個部份

輸入**2**的時候執行這個部份

若輸入**1**和**2**以外的其他數值時，則執行這個部份

Sample5.c ▶ 使用 switch 敘述

```
#include <stdio.h>

int main(void)
{
    int res;

    printf("請輸入整數。\n");

    scanf("%d", &res);

    switch(res){
        case 1:
            printf("輸入的是 1 。\n");
            break;
        case 2:
            printf("輸入的是 2 。\n");
            break;
        default:
            printf("請輸入 1 或 2 。\n");
            break;
    }
    return 0;
}
```

輸入 1 的時候，執行這個部分

輸入 2 的時候，執行這個部分

若輸入 1 和 2 以外的其他數值時，則執行這個部分

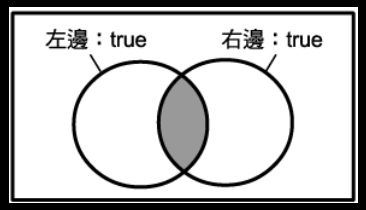
```c
#include <stdio.h>

int main(void)
{
    int res;

    printf("請輸入整數。\n");

    scanf("%d", &res);

    switch(res){
        case 1:
            printf("輸入的是1。\n");
        case 2:
            printf("輸入的是2。\n");
        default:
            printf("請輸入1或2。\n");
    }
    return 0;
}
```

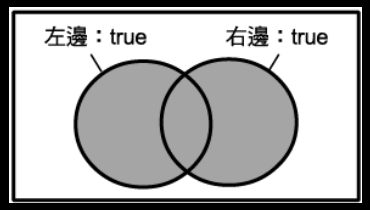
這是省略了 break 的 switch 敘述

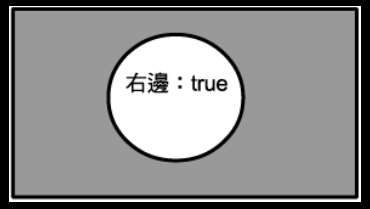
5-6 邏輯運算子

- 關於邏輯運算子
  - 邏輯運算子的功用是：進一步判斷條件式，然後得到true或false

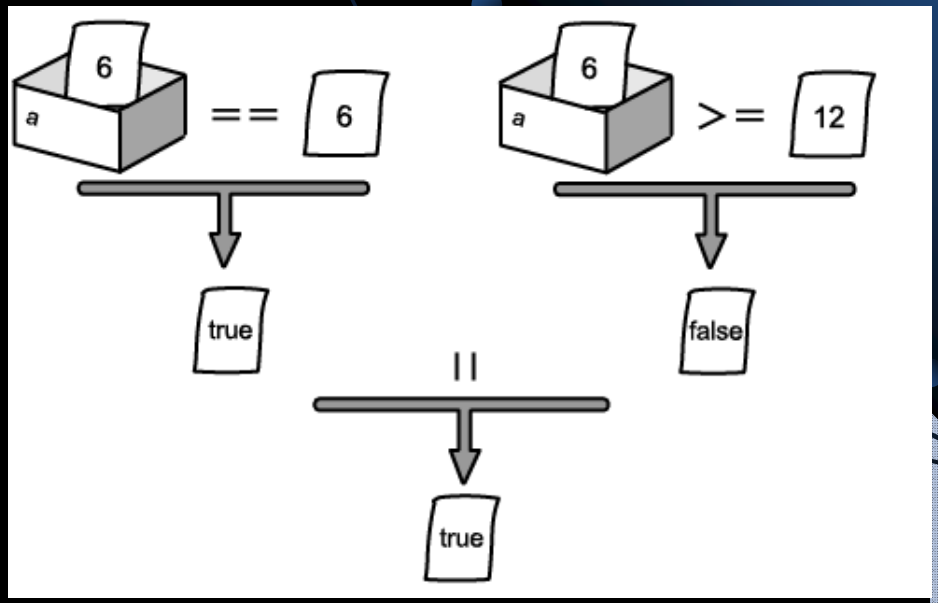
邏輯運算子	得到true值的條件	條件判斷		
&&	必須左右兩邊的條件式同時爲true	左	右	全體
		false	false	false
		false	true	false
		true	false	false
		true	true	true
||	左右兩邊的條件式只要任一方爲true即可	左	右	全體
		false	false	false
		false	true	true
		true	false	true
		true	true	true
!	右邊的條件式爲false時		右	全體
			false	true
			true	false







- 邏輯運算子使用範例：
  - 5>3 && 3==4
    (結果爲false)
  - a==6 || a>=12
    (如果變數a的值等於6或是大於等於12，結果就會是true)
  - ! (a==6)
    (當變數a等於6以外的其他值時，結果就會是true)



- 位元單位的邏輯運算子可以進行2進制數值的位元運算，並以0或1傳回結果。
- 位元單位的邏輯運算子：
  - & 左右兩邊的位元必須同時爲1
  - | 左右兩邊任一方爲1時
  - ^ 左右兩邊的位元不同時

Sample6.c ▶ 使用邏輯運算子來寫條件式

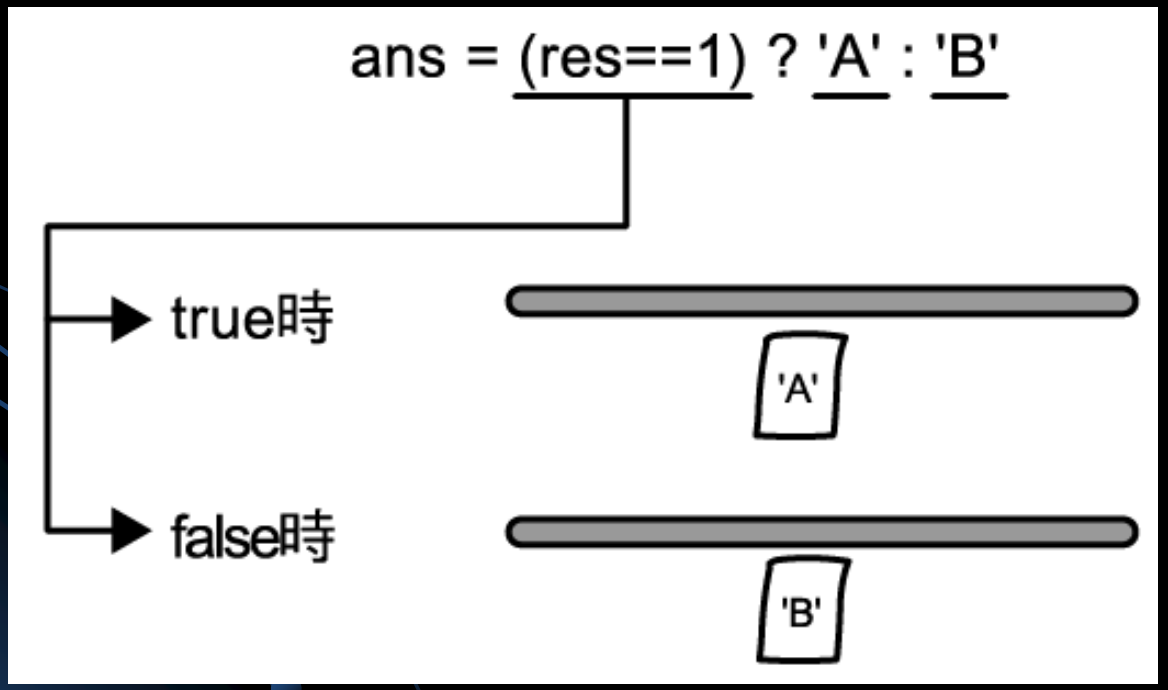
```
#include <stdio.h>

int main(void)
{
    char res;

    printf("你是男生嗎？\n");
    printf("請輸入Y或N。\n");

    res = getchar();

    if (res == 'Y' || res == 'y'){
        printf("你是男生喔。\n");
    }
    else if(res == 'N' || res == 'n'){
        printf("你是女生喔。\n");
    }
    else{
        printf("請輸入Y或N。\n");
    }

    return 0;
}
```

使用者輸入大寫Y或小寫y的話會處理這一行

使用者輸入大寫N或小寫n的話會處理這一行

輸入Y、y、N、n以外字母的話會處理這一行

- 條件運算子的語法：
  - 條件式　？　❶true的時候　：　❷false的時候
- 範例：



```c
#include <stdio.h>

int main(void)
{
    int res;
    char ans;

    printf("請問要選哪條路線？\n");
    printf("請輸入整數。\n");

    scanf("%d", &res);

    if(res == 1)
        ans = 'A';
    else
        ans = 'B';

    printf("選擇的是%c路線。\n", ans);

    return 0;
}
```

使用 if 敘述作為條件判斷

Sample7.c ▶ 使用條件運算子

```
#include <stdio.h>

int main(void)
{
    int res;
    char ans;

    printf("請問要選哪條路線？n");
    printf("請輸入整數。\n");

    scanf("%d", &res);

    ans = (res==1) ? 'A' : 'B';

    printf("選擇的是%c路線。\n", ans);

    return 0;
}
```

用條件運算子取代原來的 if 敘述