

程式設計

Ch03. Selection Structures

Chuan-Chi Lai 賴傳淇

Department of Communications Engineering
National Chung Cheng University

Spring Semester, 2024

Outline

- 1 演算法 (Algorithms)
- 2 結構化程式設計 (Structured Program Design)
- 3 if 選擇敘述式 (if Selection Statement)
- 4 布林運算 (Boolean Algebra)
- 5 if else 選擇敘述式 (if else Selection Statement)
- 6 條件運算子?: (Conditional Operator ?:)
- 7 switch 多重選擇敘述式 (switch Multiple-Selection Statement)
- 8 區塊變數 (Block Variable)

演算法 Algorithms

何謂演算法？

- 演算法 (Algorithm) 是指一個可以被計算機所執行的程序 (procedure)，其中定義了**有限個數**的動作 (actions)，且這些動作具有一定順序 (order)，來解決特定問題。
- 我們常使用虛擬程式碼 (pseudo-code) 或是流程圖 (flowchart) 來表達演算法。



虛擬程式碼 (Pseudocode)

- 虛擬程式碼 (Pseudocode) 並不是真實存在的程式語言，而是一種表達演算法的方式。
- 虛擬程式碼沒有固定的書寫方式，書寫時可能包含多種程式語言的特徵，並含有一些自然語言。
- 另外，虛擬程式碼通常使用 `:=` 或是 `←` 代表指派 (assignment)。
- 舉例：

```
1 ALGORITHM AreaOfTriangle:
2     w := input weight
3     h := input height
4     a := w * h / 2
5     print a
6 END ALGORITHM
```

流程圖 (Flowchart)

- 流程圖 (Flowchart) 是將演算法的圖形表示法，由矩形、菱形、平行四邊形、圓角矩形等組成，每個形狀都具有其含義，並由箭頭連接，表示流向 (flowline)。

圖形	說明
	開始/結束符號
	處理程序
	決策判斷
	輸入輸出

結構化程式設計 Structured Program Design

結構化程式設計 (Structured Program Design)

- 到目前為止，我們介紹過的程式都是一條一條循序的執行。事實上，有一些敘述式 (例如待會會介紹的 if 敘述式) 可以用來控制程式運行的流程。
- 根據伯姆-賈可皮尼 (Böhm-Jacopini) 理論，所有的程式均可由三種結構組成：
 - 循序結構 (Sequence Structure)
 - 選擇結構 (Selection Structure)
 - 迭代結構 (Iteration Structure)，迭代又稱迴圈 (loop)

循序結構 (Sequence Structure)

- 循序結構中，計算機會按照程式中敘述式編寫的順序逐行運行，且無任何分支。

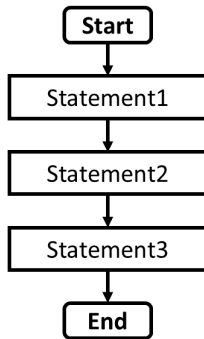
1 ALGORITHM Sequential:

2 Statement1

3 Statement2

4 Statement3

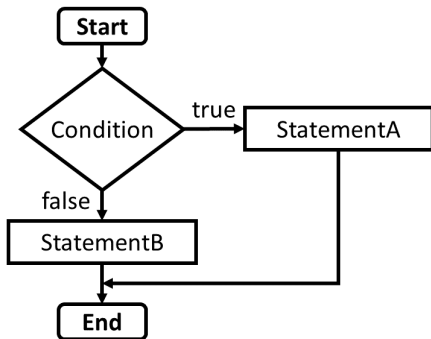
5 END ALGORITHM



選擇結構 (Selection Structure)

- 選擇結構中，計算機會經由判斷式的結果來選擇兩段程式的其中一段。

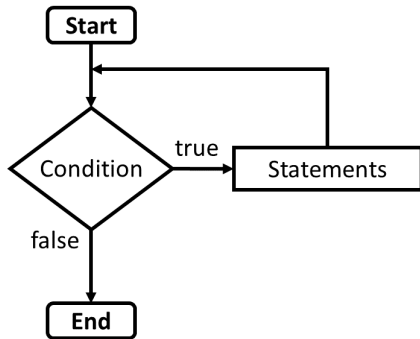
```
1 ALGORITHM Conditional:  
2   IF Conditional is true THEN:  
3     StatementsA  
4   ELSE:  
5     StatementsB  
6   END ALGORITHM
```



迭代結構 (Iteration Structure)

- 迭代結構中，計算機會經由判斷式的結果來決定是否繼續重複執行某段程式。

```
1 ALGORITHM Loop:  
2   WHILE Condition is true THEN:  
3     Statements  
4 END ALGORITHM
```



if 選擇敘述式 if Selection Statement

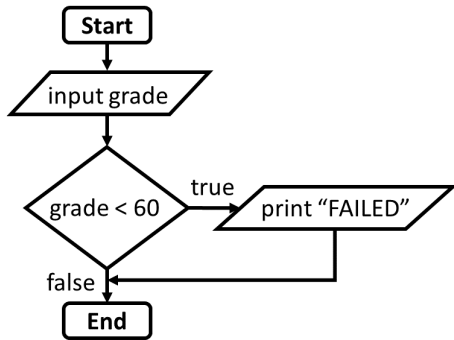
情境

- 當考慮到條件分支時，我們可以使用 if 敘述式控制流程。舉例來說，想寫一個程式：輸入一個成績，判斷這個分數是否及格，如果不及格則印出 FAILED。

```
1 grade := input grade
2 IF grade < 60 THEN:
3     print "FAILED"
```

if 選擇敘述式 (if Selection Statement)

- 輸入一個成績，判斷這個分數是否及格，如果不及格則印出 FAILED。
- 這時，我們就可以使用 if 敘述句，並判斷輸入是否小於 60 以實現這個功能。

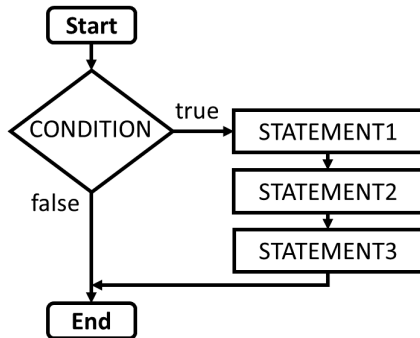


if 選擇敘述式 (if Selection Statement)

if 敘述式

- if 敘述式的 c 語法如下：

```
1 if (CONDITION)
2 {
3     Statement1;
4     Statement2;
5     Statement3;
6 }
```



if 敘述式省略大括號

- 當 if 敘述式的大括號中只有一個敘述式時，可以省略大括號。

```
1 if (CONDITION)
2 {
3     STATEMENT1;
4 }
```

```
1 if (CONDITION)
2     STATEMENT2;
```


if 選擇敘述式 (if Selection Statement)

- 當條件式“不為 0”時，代表 true。
- 如右方程式碼，因為 1 與 -100 不為 0，因此會執行 if 中的敘述，反之，條件式為 0 則不執行。

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  void main()
5  {
6      if (1)
7          printf("1 is true\n");
8      if (0)
9          printf("0 is true\n");
10     if (-100)
11         printf("-100 is true\n");
12 }
```

```
C:\Projects\test.exe  x  +  v
1 is true
-100 is true
```

關係運算子

- 當要比較兩數值之間的大小、等於不等於等關係，需使用關係運算子。關係運算子的運算結果為 0 或 1，分別代表 false 和 true。

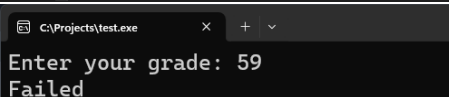
運算子	說明	語法	運算子	說明	語法
>	大於	$a > b$	>=	大於等於	$a >= b$
<	小於	$a < b$	<=	小於等於	$a <= b$
==	等於	$a == b$!=	不等於	$a != b$

- 須注意不要將 == (等於) 寫成 = (指派)

if 選擇敘述式 (if Selection Statement)

- 輸入一個成績，判斷這個分數是否及格，如果不及格則印出 FAILED 的程式如右。

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  void main()
5  {
6      int grade;
7      printf("Enter your grade: ");
8      scanf("%d", &grade);
9      if (grade < 60)
10         printf("Failed\n");
11 }
```



```
C:\Projects\test.exe
Enter your grade: 59
Failed
```

if 選擇敘述式 (if Selection Statement)

- 注意，C 語言的關係運算子不能像數學一樣連續比較。例如 $0 < \text{val} < 5$ ， $\text{val} = 10$ 時的運算：

$$0 < 10 < 5$$

$$\Rightarrow \underline{0 < 10} < 5$$

$$\Rightarrow 1(\text{true}) < 5$$

$$\Rightarrow \underline{1 < 5}$$

$$\Rightarrow 1(\text{true})$$

- 計算結果為 `true`，並非預期的 `false`，要解決這個問題，需要用到下一節介紹的邏輯運算子。

布林運算 Boolean Algebra

布林運算

- 布林運算是離散數學中的課題，常被用來解決集合 (交集、聯集、補集) 或是邏輯 (且、或、非) 的運算。

布林值 (Boolean)

- 布林值只能儲存 1(true) 與 0(false)，是許多高階語言的資料型態，但並不是 C 語言的基本資料型態。
- 使用版本 C99 以後的 C 語言，只要引入 `<stdbool.h>` 標頭檔，就可以使用布林值資料型態 `bool` 以及符號 `true` 與 `false`，分別代表整數 1 與 0。

 `stdbool.h`

```
1  #ifndef STDBOOL_H
2  #define STDBOOL_h
3
4  #define bool _Bool
5  #ifndef true
6      #define true 1
7  #endif
8
9  #ifndef false
10     #define false 0
11 #endif
12
13 #endif
```

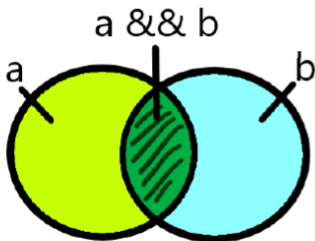
邏輯運算子

- C 語言提供的邏輯運算子可以進行且 (AND)、或 (OR)、非 (NOT) 的布林運算。邏輯運算子的運算結果為 0 或 1，分別代表 false 和 true。

運算子	說明	語法
&&	且 (logic AND)	$p \ \&\& \ q$
	或 (logic OR)	$p \ \ q$
!	非 (logic NOT)	$!p$

布林運算 (Boolean Algebra)

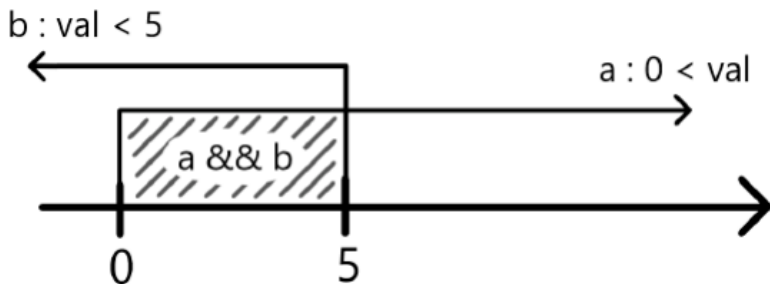
&& 且 (logic AND)



$\begin{smallmatrix} a \\ b \end{smallmatrix}$	0	1
0	0	0
1	0	1

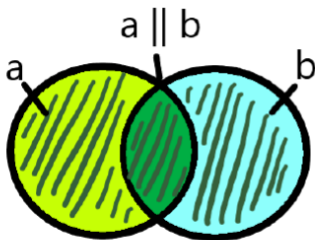
布林運算 (Boolean Algebra)

- 舉例來說， $0 < \text{val} \ \&\& \ \text{val} < 5$ ，結果為 true 時，val 的範圍：



布林運算 (Boolean Algebra)

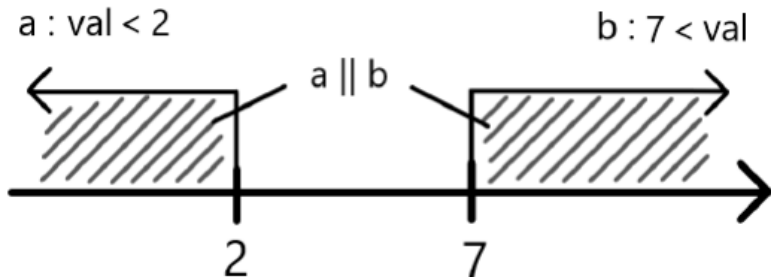
\parallel 或 (logic OR)



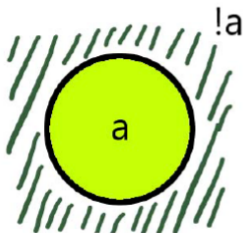
$\begin{smallmatrix} a \\ \backslash \\ b \end{smallmatrix}$	0	1
0	0	1
1	1	1

布林運算 (Boolean Algebra)

- 舉例來說， $val < 2 \parallel 7 < val$ ，結果為 true 時， val 的範圍：



! 非 (logic NOT)



a	0	1
!a	1	0

思考練習

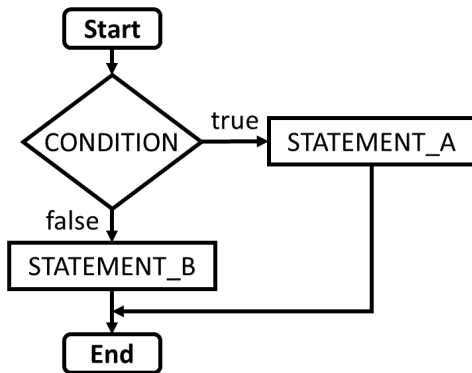
- 判斷式 “ $!(10 < a \parallel 0 > a) \&\& a > 5$ ”，結果為 true 時，a 的範圍為何？

if else 選擇敘述式 if else Selection Statement

if else 選擇敘述式 (if else Selection Statement)

- if 敘述式會在判斷式為 true 時執行大括號中的敘述，而在 false 的時候不執行任何敘述。若要在 false 時選擇執行另一段敘述時，需使用 if else 敘述句。

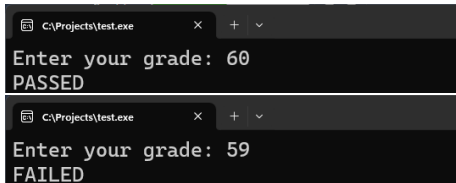
```
1 if (CONDITION)
2 {
3     STATEMENT_A;
4 }
5 else
6 {
7     STATEMENT_B;
8 }
```



if else 選擇敘述式 (if else Selection Statement)

- 輸入一個成績，判斷這個分數是否及格，如果及格則印出 PASSED，否則印出 FAILED 的程式：

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 void main()
5 {
6     int grade;
7     printf("Enter your grade: ");
8     scanf("%d", &grade);
9     if (grade >= 60) {
10         printf("PASSED\n");
11     } // end if
12     else {
13         printf("FAILED\n");
14     } // end else
15 }
```



```
C:\Projects\test.exe x + v
Enter your grade: 60
PASSED

C:\Projects\test.exe x + v
Enter your grade: 59
FAILED
```

巢狀 if else

- 有些情況，我們會需要判斷不只一個條件進行選擇，例如成績的等第 (A、B、C 等)。
- 假設百分比制換算等第標準如下：

100 ~ 80 分	A 等
79 ~ 70 分	B 等
69 ~ 60 分	C 等
59 ~ 50 分	D 等
49 ~ 80 分	E 等

if else 選擇敘述式 (if else Selection Statement)

- 照著上頁的等第換算表，我們可以將程式碼寫成 (a)，並且因為 else 的大括號內只有一個敘述式，可以將大括號省略，如 (b)。
- 註：if 與 else 同為一個敘述式

```
if (grade >= 80)
{
    printf("A\n");
}
else
{
    if (grade >= 70)
    {
        printf("B\n");
    }
    else
    {
        if (grade >= 60)
        {
            printf("C\n");
        }
        else
        {
            if (grade >= 50)
            {
                printf("D\n");
            }
            else
            {
                printf("E\n");
            }
        }
    }
}
```

(A)

```
if (grade >= 80)
    printf("A\n");
else
    if (grade >= 70)
        printf("B\n");
    else
        if (grade >= 60)
            printf("C\n");
        else
            if (grade >= 50)
                printf("D\n");
            else
                printf("E\n");
```

(B)

if else 選擇敘述式 (if else Selection Statement)

- 接著，因為 (b) 的縮排層數太多不易閱讀的關係，我們通常會寫成 (c) 的形式，並將 else if 看作是一個語法。
- 部分教科書將此稱為 “if ... else if ... else 敘述式”。

```
if (grade >= 80)
    printf("A\n");
else
    if (grade >= 70)
        printf("B\n");
    else
        if (grade >= 60)
            printf("C\n");
        else
            if (grade >= 50)
                printf("D\n");
            else
                printf("E\n");
```

(B)

```
if (grade >= 80)
    printf("A\n");
else if (grade >= 70)
    printf("B\n");
else if (grade >= 60)
    printf("C\n");
else if (grade >= 50)
    printf("D\n");
else
    printf("E\n");
```

(C)

思考練習

- 下方 2 段程式碼，最後變數 a 的值分別為何？

```
1 int a = 3;  
2 if (a >= 2)  
3     a *= 2;  
4 else if (a >= 3)  
5     a *= 3;  
6 else if (a >= 4)  
7     a *= 4;  
8 else if (a >= 5)  
9     a *= 5;
```

```
1 int a = 3;  
2 if (a >= 2)  
3     a *= 2;  
4 if (a >= 3)  
5     a *= 3;  
6 if (a >= 4)  
7     a *= 4;  
8 if (a >= 5)  
9     a *= 5;
```

條件運算子?: (Conditional Operator ?:)

條件運算子?:
Conditional Operator ?:

條件運算子?: (Conditional Operator ?:)

條件運算子?:

- 條件運算子?: 是 C 的唯一一個三元運算子，甚至可以用三元運算子來代稱條件運算子。其語法如下：

```
1 CONDITION ? EXPRESSION1 : EXPRESSION2;
```

- EXPRESSION1 與 EXPRESSION2 的資料型態不一致時，會進行算術轉換成相同的資料型態。
- 當 CONDITION 的值不為 0(=true) 時，條件運算子的運算結果為 EXPRESSION1 的值。
- 當 CONDITION 的值為 0(=false) 時，條件運算子的運算結果為 EXPRESSION2 的值。

條件運算子?: (Conditional Operator ?:)

- 在有些情況，if else 敘述句中，兩個大括號內的敘述僅僅只有一小部分的差異，例如：

```
1 if (type == 'A')
2     val = 100;
3 else
4     val = 200;
```

- 可以使用條件運算子將上述改寫為：

```
1 val = type == 'A' ? 100 : 200;
```

- 但是使用條件運算子會使程式碼閱讀性較差，須斟酌使用。

條件運算子?: (Conditional Operator ?:)

- 運算優先度由高而低排序，以分隔線表示不同優先度：

運算子	運算	關聯性
sizeof()	取變數或型態的大小	由右至左
!	邏輯 NOT	由右至左
&	取位址	由右至左
(type)	型態轉換	由右至左
*, /, %	乘法、除法、模 (餘) 數	由左至右
+, -	加法、減法	由左至右
<, <=, >, >=	小於、小於等於、大於、大於等於	由左至右
==, !=	等於、不等於	由左至右
&&	邏輯 AND	由左至右
	邏輯 OR	由左至右
?:	條件	由右至左
=, +=, -=, *=, /=, %=	簡單指派、複合指派	由右至左

switch 多重選擇敘述式 (switch Multiple-Selection Statement)

switch 多重選擇敘述式 switch Multiple-Selection Statement

switch 多重選擇敘述式 (switch Multiple-Selection Statement)

- 有時，我們可能會遇到判斷一個數值是否為數個整數中的其中一個，並採取對應的動作，以 if else 可寫成右方程式碼。
- 但這個寫法重複寫了好多次相同的變數，而且有些不工整。

```
1 if (day == 1)
2 {
3     printf("MON\n");
4 }
5 else if (day == 2)
6 {
7     printf("TUE\n");
8 }
9 else if (day == 3)
10 {
11     printf("WED\n");
12 }
13 else if (day == 4)
14 {
15     printf("THU\n");
16 }
17 else if (day == 5)
18 {
19     printf("FRI\n");
20 }
21 else
22 {
23     printf("WEEKEND\n");
24 }
```

switch 多重選擇敘述式 (switch Multiple-Selection Statement)

- 當一個判斷式都在判斷同一個整數時，可以選擇格式比較工整的 switch 敘述式。

```
1 switch (day)
2 {
3     case 1:
4         printf("MON\n");
5         break;
6     case 2:
7         printf("TUE\n");
8         break;
9     case 3:
10        printf("WED\n");
11        break;
12    case 4:
13        printf("THU\n");
14        break;
15    case 5:
16        printf("FRI\n");
17        break;
18    default:
19        printf("WEEKEND\n");
20        break;
21 }
```

switch 多重選擇敘述式 (switch Multiple-Selection Statement)

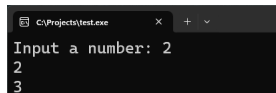
- switch 敘述式語法如右。
- 其中，EXPRESSION 必須是整數型態，CONST-EXPRESSION 必須是整數字面常量 (或由字面常量組成的運算式)。
- 執行序會從與 EXPRESSION 的值相等的 case 標籤進入，直到碰到 break 時離開 switch 敘述式。若沒有相符合的 case 標籤，則會從 default 標籤進入。

```
1 switch (EXPRESSION)
2 {
3     case CONST-EXPRESSION1:
4         STATEMENTS;
5         break;
6     case CONST-EXPRESSION2:
7         STATEMENTS;
8         break;
9     default:
10        STATEMENTS;
11        break;
12 }
```

switch 多重選擇敘述式 (switch Multiple-Selection Statement)

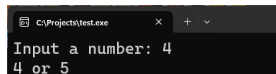
- 執行序會從符合的 case 標籤進入，直到碰到 break 時離開 switch 敘述式。下圖為一實際程式碼範例，以及輸入與輸出結果。

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  void main()
5  {
6      int val;
7      printf("Input a number: ");
8      scanf("%d", &val);
9
10     switch (val)
11     {
12         case 1:
13             printf("1\n");
14         case 2:
15             printf("2\n");
16         case 3:
17             printf("3\n");
18             break;
19         case 4:
20         case 5:
21             printf("4 or 5\n");
22             break;
23     }
24 }
```



C:\Projects\test.exe

Input a number: 2
2
3



C:\Projects\test.exe

Input a number: 4
4 or 5

區塊變數 Block Variable

區塊變數 (Block Variable)

- 在章節的最後，我們來釐清變數於區塊 (大括號) 中的作用域 (scope) 與生命週期 (lifetime)。

區塊變數 (Block Variable)

- 情況一：當變數於區塊外宣告，其作用域包括區塊內的範圍。

```
1 #include <stdio.h>
2 int main()
3 {
4     int foo = 10;
5     if (1)
6     {
7         printf("%d", foo); // OK
8     }
9     printf("%d", foo); // OK
10 }
```

區塊變數 (Block Variable)

- 情況二：當變數於區塊內宣告，其作用域不包括區塊外的範圍，且生命週期僅到離開區塊之前。

```
1 #include <stdio.h>
2 int main()
3 {
4     if (1)
5     {
6         int foo = 5;
7         printf("%d", foo); // OK
8     }
9     printf("%d", foo); // ERROR!
10 }
```

區塊變數 (Block Variable)

- 情況三：當變數於區塊內宣告，且與區塊外的變數同名。
 - 區塊內宣告的變數作用域不包括區塊外的範圍，且生命週期僅到離開區塊之前。
 - 區塊外的變數的作用域不包括區塊內同名變數宣告後的範圍。

```
1 #include <stdio.h>
2 int main()
3 {
4     int foo = 10;
5     if (1)
6     {
7         printf("%d", foo); // print 10
8         int foo = 5;
9         printf("%d", foo); // print 5
10    }
11    printf("%d", foo); // print 10
12 }
```

Q & A