程式語言一　講義

條件判斷

Hanly《C 語言詳論》6e

§ 1　名詞速查表

中文	English	一句話定義	科目	章節
條件判斷	if statement	條件為真才執行某段程式	程式語言一	條件判斷
否則	else	if 條件為假時執行的分支	程式語言一	條件判斷
多重分支	else if chain	一連串互斥條件，第一個成立就停	程式語言一	條件判斷
多路分支	switch	依一個整數/字元值跳到對應 case	程式語言一	條件判斷
貫穿	fall-through	switch 漏寫 break，會繼續執行下一個 case	程式語言一	條件判斷
巢狀	nested if	if 裡面再放 if	程式語言一	條件判斷
懸置 else	dangling else	`else` 歸屬不清時，配對最近的 if	程式語言一	條件判斷
三元運算子	ternary `?:`	`條件 ? 真值 : 假值`，簡短的條件取值	程式語言一	條件判斷

§ 2　核心概念

核心概念

`if (條件) { ... }`：條件為**真（非零）**就執行大括號內容，**假（0）**就跳過。`if`／`else` 永遠**只執行其中一個**。

C 的真假：**0 是假**（含 `'\0'`、`NULL`），**任何非零都是真**（`1`、`-1`、`42`…）。所以 `if (x = 0)` 是把 0 指派給 x、回傳 0＝假；要比較得用 `==`。

§ 3 主要內容

3.1 if / else / else if 鏈

```
if (score >= 90)      printf("A\n");
else if (score >= 80) printf("B\n");
else if (score >= 70) printf("C\n");
else                  printf("F\n");
```

從上到下，**找到第一個成立的就進去、後面全跳過**。所以 **順序很重要：嚴格的條件放上面、寬鬆的放下面**。若把 `score >= 60` 放最前面，95 分也會先掉進去，後面的 `>= 90` 永遠到不了。

3.2 永遠加大括號

```
if (x > 0)
    printf("正數\n");
printf("x=%d\n", x);   // 看起來在 if 裡，其實永遠執行！
```

沒大括號時 `if` 只管**下一個敘述**。第二行不屬於 `if`。Apple 2014 的「goto fail」安全漏洞就是少了大括號。**永遠加 `{}`**。

3.3 C 的真假與 `=` 陷阱

```
int x = 5;
if (x = 0) printf("A"); else printf("B");   // 印 B，且 x 被改成 0！
```

`x = 0` 是**指派**（回傳 0＝假）→ 走 `else`。比較要用 `==`；`-Wall` 會幫你抓這個錯。

3.4 巢狀 if 與 dangling else

```
int a = 1, b = 0;
if (a)
    if (b) printf("A\n");
    else   printf("B\n");   // 這個 else 配對誰？
```

規則：`else` **配對最近的** `if`（這裡配 `if (b)`）。`a=1`→進外層→`b=0`→`else`→印 `B`。**加大括號就沒有歧義**。

3.5 獨立 if vs else if

```
// 獨立 if：每個都檢查          // else if：互斥，只走一條
if (x > 0)  printf("正");      if (x > 100) printf("超大");
if (x > 10) printf("大");      else if (x > 10) printf("大");
if (x > 100)printf("超大");    else if (x > 0)  printf("正");
// x=50 → 「正大」              // x=50 → 只印「大」
```

求最大值就靠**獨立 if**（每個都檢查）：`max=a; if(b>max)max=b; if(c>max)max=c;`。

3.6 switch：多路分支

```
switch (grade) {
    case 'A': printf("優\n"); break;
    case 'B': printf("良\n"); break;
    default:  printf("其他\n");
}
```

依一個整數／字元跳到對應 `case`。**每個 case 結尾要 `break`**，否則會**貫穿（fall-through）**繼續執行下一個 case。`default` 處理沒對到的情況。

§ 4 語法與函式速查

```
if (cond) { ... } else if (cond2) { ... } else { ... }

switch (x) {              // x 必須是整數/字元
    case 1: ...; break;
    case 2: ...; break;
    default: ...;
}

cond ? a : b              // 三元：cond 真取 a、假取 b
```

§ 5 常見錯誤

常見錯誤

- `if (x = 0)` 把 `==` 寫成 `=`（指派、回傳值當條件）。
- 省略大括號，誤以為下一行也在 if 裡（goto fail 教訓）。
- `else if` 鏈把寬鬆條件放上面，嚴格條件永遠到不了。
- `switch` 的 case 忘了 `break`，發生貫穿。
- 把該互斥的分支寫成多個獨立 if（每個都檢查，邏輯錯）。
- 忘記 `else` 配最近的 if（dangling else）；加 `{}` 解決。

§ 6 練習題

練習 1（一般題）：else if 鏈追蹤

`int score = 78;`，寫出輸出。

```c
if (score >= 90) printf("A\n");
else if (score >= 80) printf("B\n");
else if (score >= 70) printf("C\n");
else printf("F\n");
```

> **引導步驟**
>
> 1. 從上到下找第一個成立的，後面全跳過。

解答

`C`。`78>=90` 假、`78>=80` 假、`78>=70` 真→印 C，後面跳過。

練習 2（一般題）：switch 貫穿

寫出輸出（注意有沒有 break）。

```
int n = 1;
switch (n) {
    case 1: printf("one\n");
    case 2: printf("two\n"); break;
    case 3: printf("three\n");
}
```

引導步驟

1. `case 1` 沒有 break，會貫穿到 `case 2`。

解答

```
one
two
```

`n=1` 進 case 1 印 one，**沒 break → 貫穿** case 2 印 two，遇到 break 停。

練習 3（重要題）：抓 bug

下列為何印 `B` 且改了 `x`？怎麼修？

```c
int x = 5;
if (x = 0) printf("A\n"); else printf("B\n");
```

> **引導步驟**
>
> 1. `x = 0` 是比較還是指派？回傳什麼？
> 2. 0 在 if 裡算真還假？

解答

`x = 0` 是指派（把 0 存進 x），回傳 0＝假 → 走 `else` 印 B；而且 x 被改成 0。要比較得寫 `if (x == 0)`。開 `-Wall` 會警告。

§ 7　自我檢核

- ☐ 知道條件「真＝非零、假＝0」，會用 if / else / else if。
- ☐ 知道 else if 鏈只走一條、嚴格條件要放上面。
- ☐ 永遠加大括號，知道 goto fail 的教訓。

- [ ] 分得清 `=`（指派）與 `==`（比較）在條件裡的陷阱。
- [ ] 知道 else 配最近的 if，會用 `{}` 消歧義。
- [ ] 分得清獨立 if（各自檢查）與 else if（互斥），會寫三數最大值。
- [ ] 會用 switch，知道每個 case 要 break、否則貫穿。