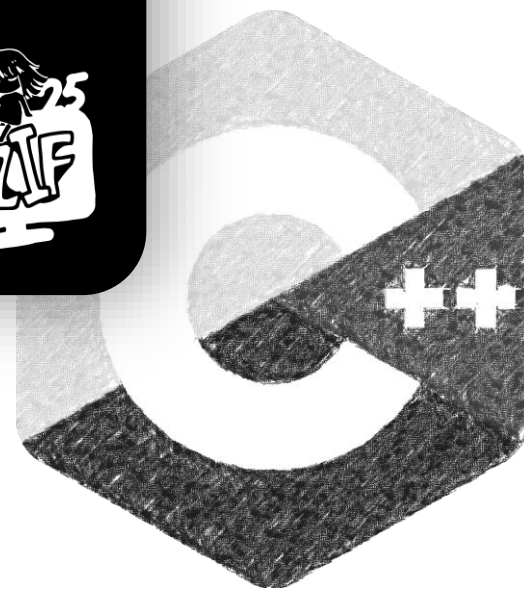





流程控制

1021資訊社






= Assign 賦值

$$A = B + C$$


等號右邊的值 給 等號左邊的值

$$A + B = C$$


等號左邊不能有運算式！



+ - * / 加減乘除

$$A = 2 + 3$$

$$A: 5$$

$$B = 2 - 3$$

$$B: -1$$

$$C = 2 * 3$$

$$C: 6$$

$$D = 7 / 3$$

$$D: 2$$

若 int / int，則結果會是無條件進位到個位數的 int

$$E = 7 / 3.0$$

$$E: 2.333...$$

可以將 int / int 其中一個改成 float 或 double，結果會是 float 或 double

$$E = 7.0 / 3$$



% Mod 取餘

$$A = B \% C$$

$$A = 7 \% 3 \quad A: 1$$

$$7 / 3 = 2 \dots 1$$



$+=$ $-=$ $*=$ $/=$ $\%=$ $++$ $--$

$A = A + B \rightarrow A += B$

$A = A - B \rightarrow A -= B$

$A = A * B \rightarrow A *= B$

$A = A / B \rightarrow A /= B$

$A = A \% B \rightarrow A \% = B$

$A = A + 1 \rightarrow A ++$

$A = A - 1 \rightarrow A --$

比較運算子

$X = 10, y = 3$

$X > y \rightarrow \text{true}$

$X < y \rightarrow \text{false}$

$X \geq y \rightarrow \text{true}$

$X \leq y \rightarrow \text{false}$

$X == y \rightarrow \text{false}$

$X != y \rightarrow \text{true}$

$>$ 大於

$<$ 小於

\geq 大於或等於

\leq 小於或等於

$==$ 等於

$!=$ 不等於

if 敘述

- 抉擇問題：

如果今天下雨，我要帶雨傘出門。

條件

動作/敘述

如果成績低於60分，即為不及格。

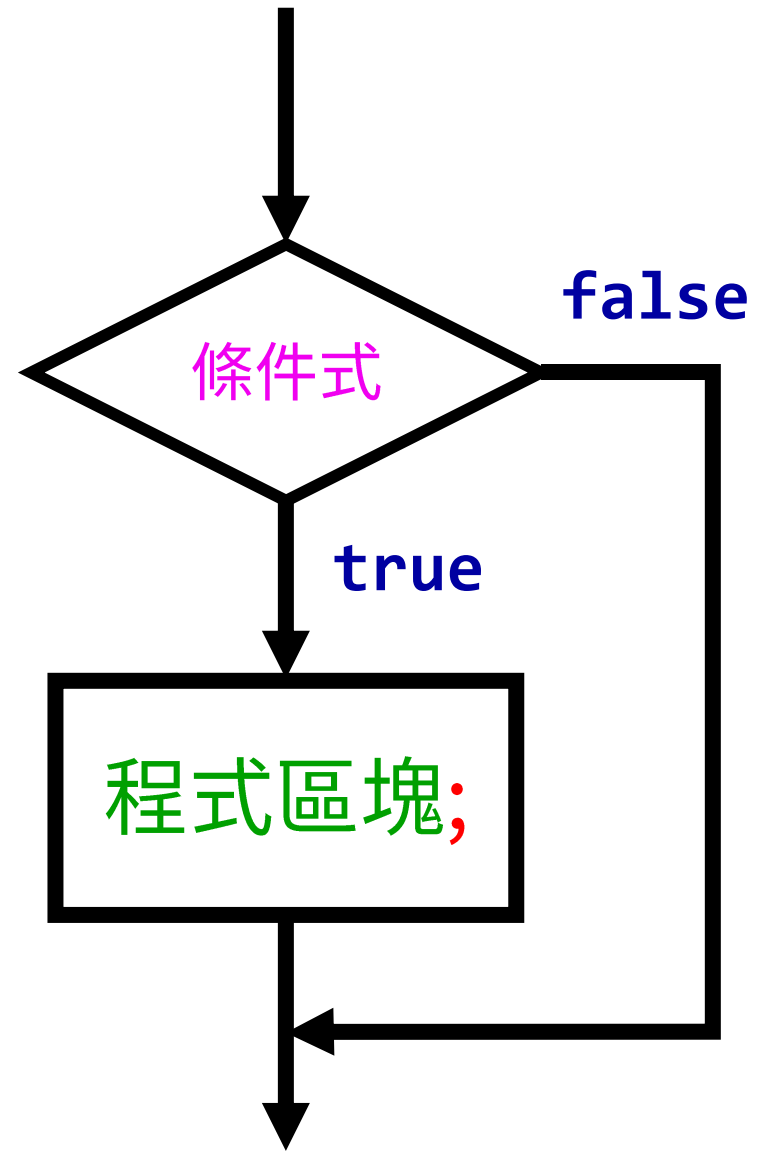
條件

動作/敘述

if 敘述

- 在C++中，可以使用：

if (條件式)
程式區塊;

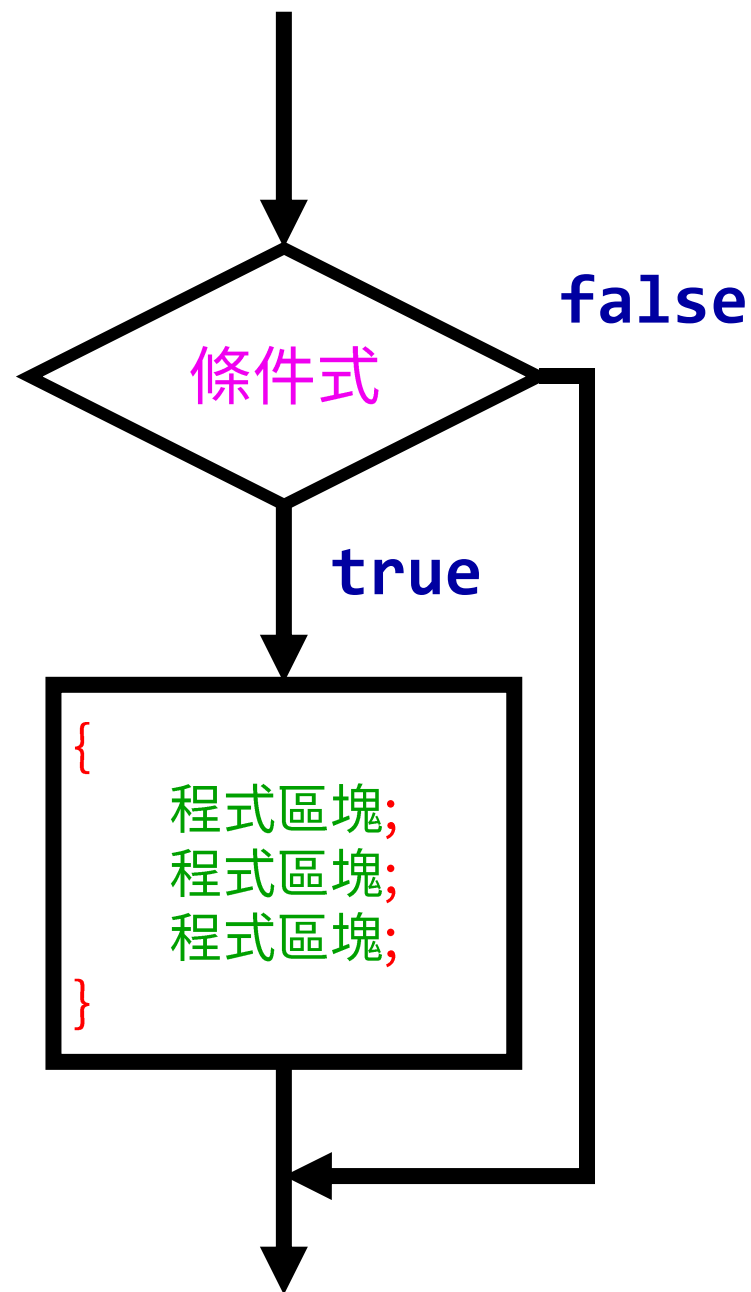


if 敘述

- 如果有多行程式要加 {}

```
if (條件式){  
    程式區塊;  
    程式區塊;  
    程式區塊;  
}
```

```
}
```



if 敘述

- 翻譯成C++：

如果今天下雨，
我要帶雨傘出門。

```
if (今天下雨){  
    我要帶雨傘;  
}
```

- 翻譯成C++：

如果成績低於60分，
即為不及格。

```
if (成績 < 60){  
    即為不及格;  
}
```

if...else 敘述

- 抉擇問題：

如果今天下雨，我要帶雨傘出門，否則不用帶雨傘。

條件

成立時的
動作/敘述

不成立時的
動作/敘述

如果成績低於60分，即為不及格，反之為及格。

條件

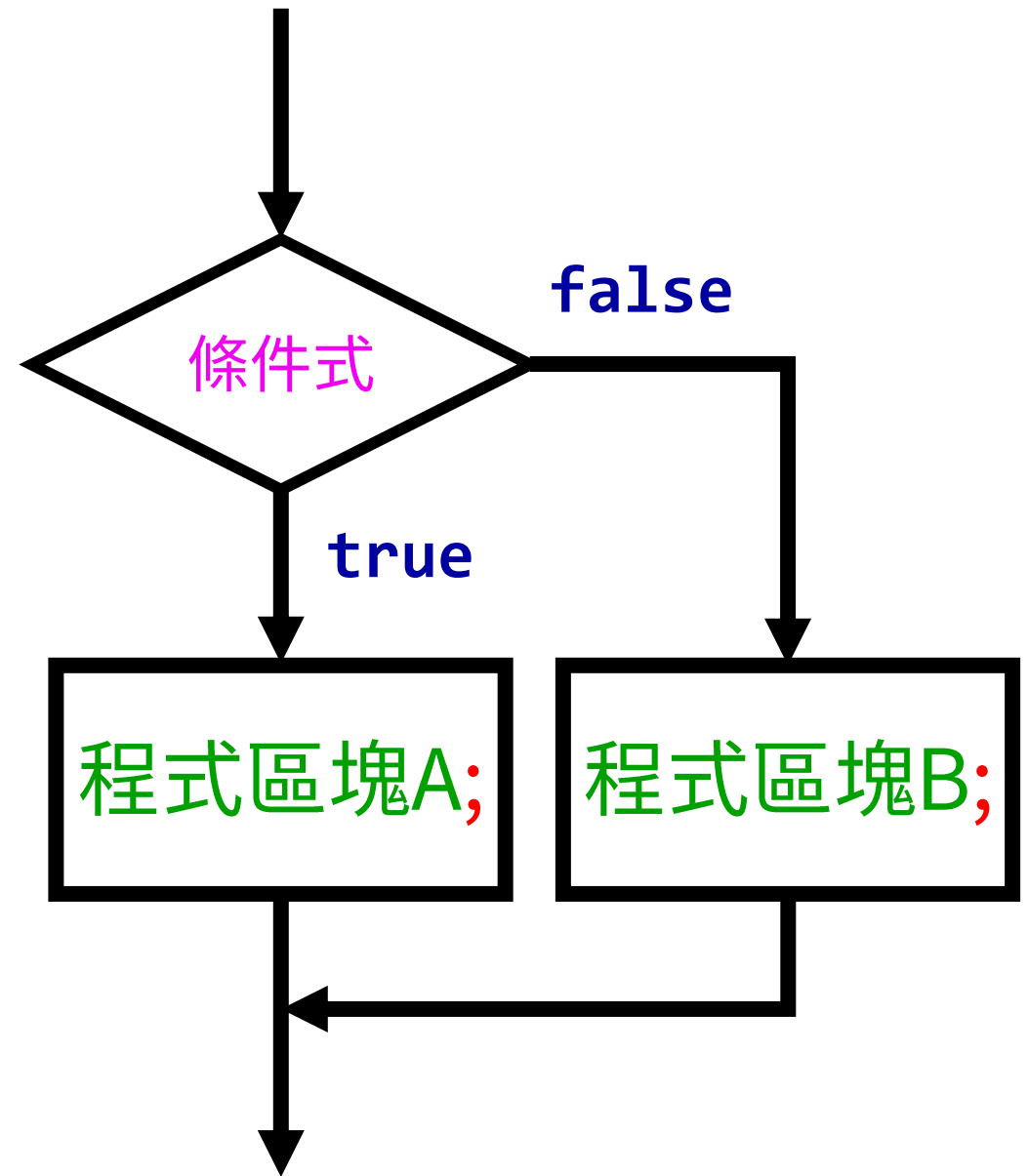
成立時的
動作/敘述

不成立時的
動作/敘述

if...else 敘述

- 在C++中，可以使用：

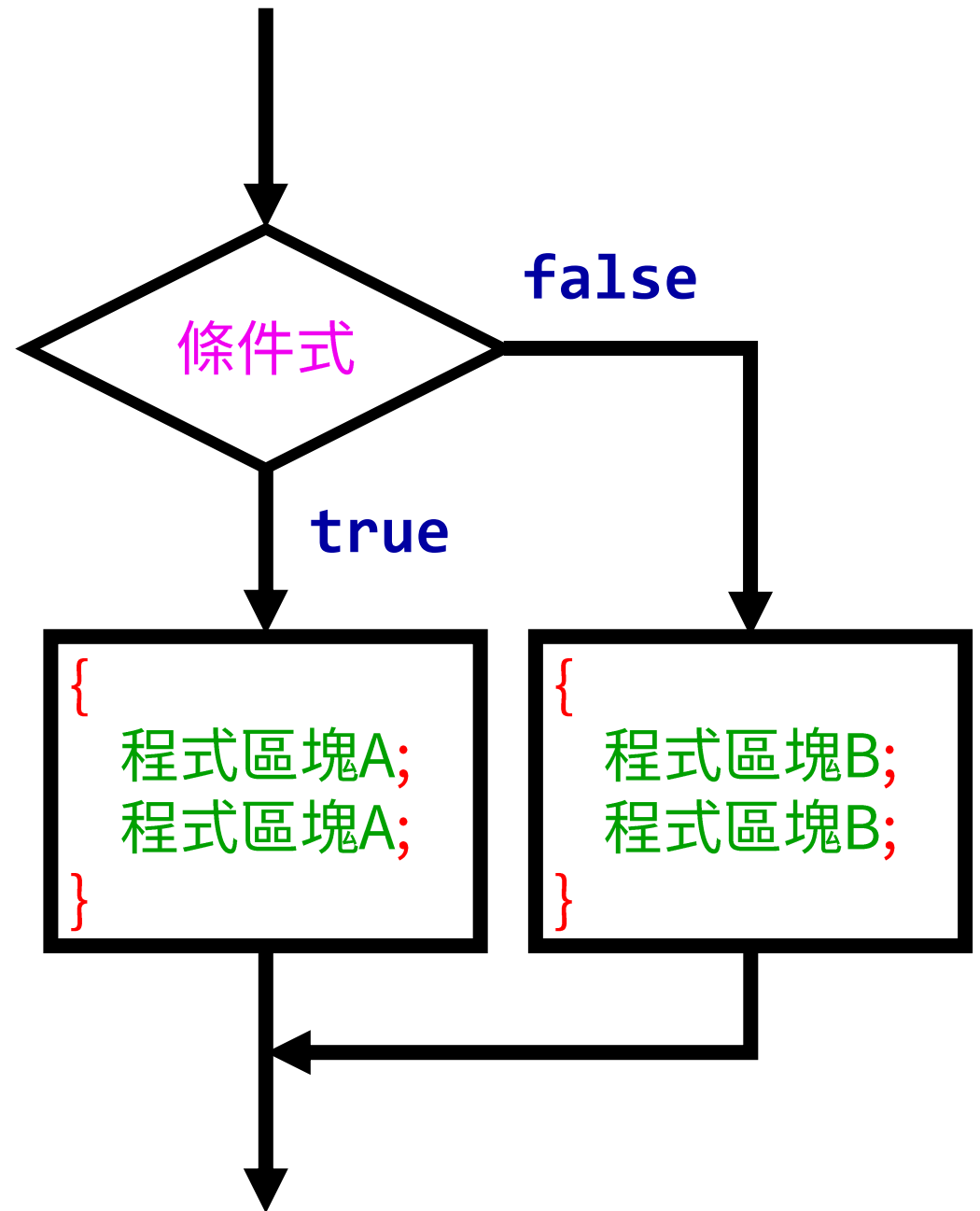
```
if (條件式)  
    程式區塊A;  
else  
    程式區塊B;
```



if...else 敘述

- 如果有多行程式要加 {}

```
if (條件式){  
    程式區塊A;  
    程式區塊A;  
}  
else{  
    程式區塊B;  
    程式區塊B;  
}
```



if...else 敘述

- 抉擇問題：

如果今天下雨，

條件

我要帶雨傘出門，

成立時的動作/敘述

否則不用帶雨傘。

不成立時的動作/敘述

- 翻譯成C++：

```
if (今天下雨){  
    我要帶雨傘出門;  
}  
else{  
    不用帶雨傘出門;  
}
```

if...else 敘述

- 抉擇問題：

如果成績低於60分，

條件

即為不及格，反之為及格。

成立時的
動作/敘述

不成立時的
動作/敘述

- 翻譯成C++：

```
if (成績 < 60){  
    不及格;  
}  
else{  
    及格;  
}
```

if...else if 敘述

- 抉擇問題：

如果年紀 < 6，則可看普遍級的影片，

否則如果年紀 < 12，則可看普遍級與保護級的影片，

否則如果年紀 < 18，則可看非限制級的影片，

否則如果年紀 >= 18，則可看各級影片。

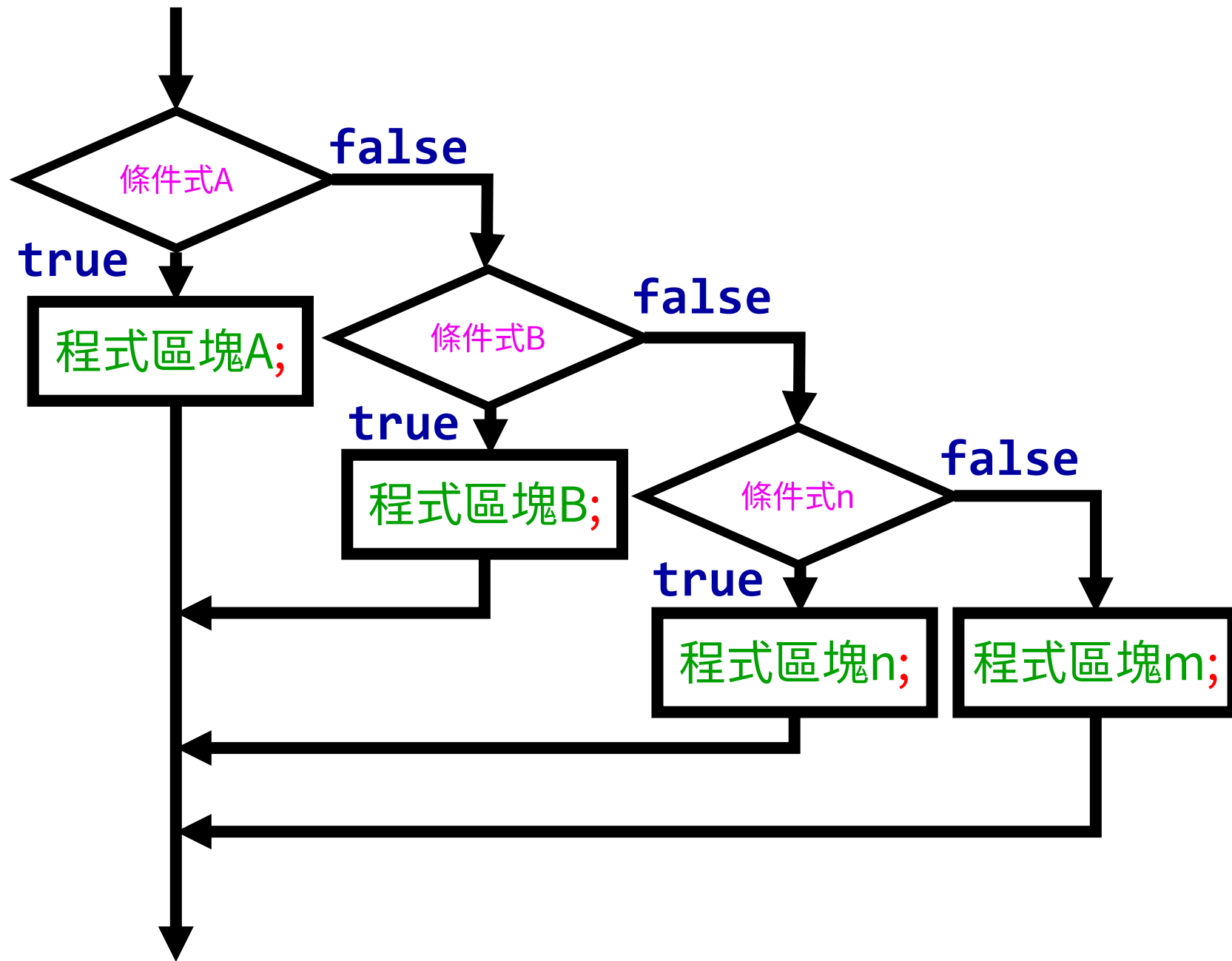
條件

動作/敘述

if...else if 敘述

- 在C++中，可以這樣用：

```
if (條件式A)  
    程式區塊A;  
else if (條件式B)  
    程式區塊B;  
    ⋮  
else if (條件式n)  
    程式區塊n;  
else  
    程式區塊m;
```



if...else if 敘述

```
1  #include <iostream>
2  using namespace std;
3
4  int main(){
5      int age;
6      cout << "請輸入你的年紀：";
7      cin >> age;
8
```

請輸入你的年紀：|

```
9      if (age < 6)
10         cout << "可看普遍級的影片\n";
11     else if (age < 12)
12         cout << "可看普遍級與保護級的影片\n";
13     else if (age < 18)
14         cout << "可看非限制級的影片\n";
15     else
16         cout << "可看各級影片\n";
17     return 0;
18 }
```

練習題 - 兩光法師占卜術

兩光法師時常替人占卜，由於他算得又快有便宜，因此生意源源不絕，時常大排長龍，他想算得更快一點，因此找了你這位電腦高手幫他用電腦來加快算命的速度。

他的占卜規則很簡單，規則是這樣的，輸入一個日期，然後依照下面的公式：

M=月

D=日

$S = (M * 2 + D) \% 3$

得到 S 的值，再依照 S 的值從 0 到 2 分別給與普通、吉、大吉 等三種不同的運勢

輸入說明

輸入資料共一行，包含兩個整數，分別為月份及日期

輸出說明

運勢

範例輸入1

1 1

範例輸出1

普通

範例輸入2

1 2

範例輸出2

吉

練習題 - 兩光法師占卜術

```
1  #include <iostream>
2  using namespace std;
3
4  int main(){
5      int M, D, S;
6      cin >> M >> D;
7      S = (M * 2 + D) % 3;
8  }
```

宣告月份M、日期D、S

輸入月份M 和 日期D

計算S

判斷運勢

```
if (S == 0)
    cout << "普通";
else if (S == 1)
    cout << "吉";
else if (S == 2)
    cout << "大吉";

return 0;
```

邏輯運算 – AND「而且」邏輯

AND運算二側的運算元須同時為真，其結果才為真

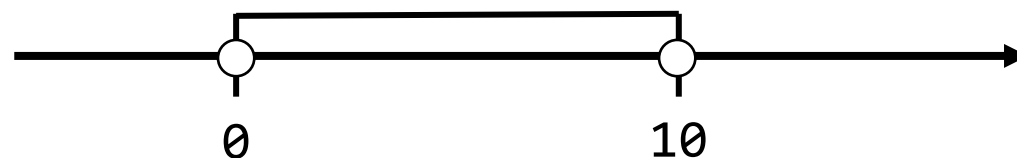
AND運算	A: true	A: false
B: true	true	false
B: false	false	false

在C++中為「&&」或「and」

邏輯運算 – AND 「而且」 邏輯

```
1  #include <iostream>
2  using namespace std;
3
4  int main(){
5      int a;
6      cin >> a;
7      if (0 < a && a < 10)
8          cout << "0 < " << a << " < 10\n";
9  }
```

範例：判斷一數是否在某一區間內



需要 $0 < a$ 和 $a < 10$
同時為true，運算結果才
為true

邏輯運算 – OR「或者」邏輯

OR運算二側的運算元只要有一者為真，其結果恆為真

OR運算	A: true	A: false
B: true	true	true
B: false	true	false

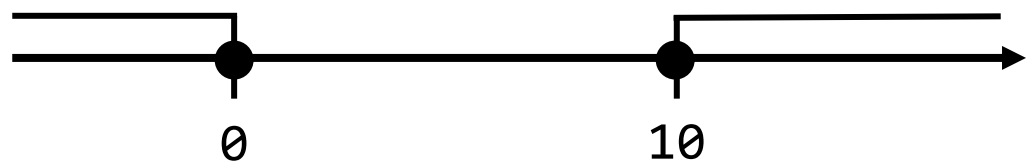
在C++中為「||」或「or」



邏輯運算 – OR 「或者」 邏輯

```
1  #include <iostream>
2  using namespace std;
3
4  int main(){
5      int a;
6      cin >> a;
7      if (a <= 0 || a >= 10)
8          cout << "a <= 0 || a >= 10\n";
9  }
```

範例：判斷一數是否在某一區間內



需要 $a \leq 0$ 或 $a \geq 10$
任一為true，運算結果為
true

邏輯運算 – NOT「反向」邏輯

當運算元為真時，NOT運算結果為非。當運算元為非時，NOT運算結果為真。

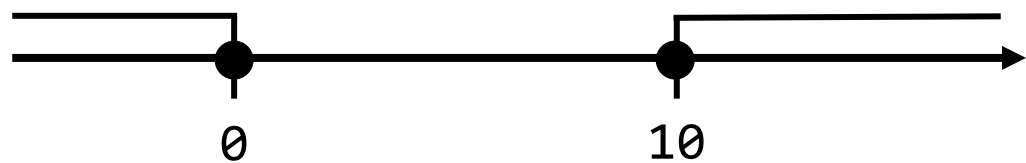
NOT運算	A: true	A: false
	false	true

在C++中為「!」或「not」

邏輯運算 – NOT「反向」邏輯

```
1  #include <iostream>
2  using namespace std;
3
4  int main(){
5      int a;
6      cin >> a;
7      if (!(0 < a && a < 10))
8          cout << "!(0 < a && a < 10)";
9  }
```

範例：判斷一數是否在某一區間內



需要 $0 < a \ \&\& \ a < 10$
的結果為false時，運算結果才為true

練習題 – 常客優惠方案

臺北大眾捷運股份有限公司針對持儲值卡的乘客有常客優惠方案，以下為其基本規則：

1. 持儲值卡搭乘捷運以全票票價扣款，並於票卡累積搭乘次數及搭乘金額。
2. 依每卡每月累計搭乘次數，決定現金回饋比例，並依累計搭乘金額，計算回饋金。

當月回饋金＝前月累計搭乘金額×現金回饋比例 (尾數不滿1元者，按四捨五入計算)	
前月累計搭乘次數	現金回饋比例
11～20次	10%
21～30次	15%
31～40次	20%
41～50次	25%
51次以上	30%

3. 當月搭乘捷運首次通過捷運閘門時，回饋金即以自動加值方式，存入同一張票卡之電子錢包。回饋金等同現金，除可用於搭乘捷運，亦可小額消費。
4. 每月累積期間：自每月1日0時至該月最後1日24時止。
5. 回饋金自動加值有效期間：自當月1日零時起，1年內有效。於有效期間內，首次通過捷運閘門即自動加值，超過有效期間均未搭乘捷運，則尚未存入電子錢包之回饋金，將於到期日之翌日零時起失效歸零。



輸入說明

小明是每天持悠遊卡普通卡（全票票價）使用臺北捷運通勤的乘客，他很好奇他每個月的常客優惠回饋金有多少。

請寫出一支程式幫他計算每個月的常客優惠回饋金。

輸入資料共一行，包含兩個整數並使用空格分開，分別為該月搭乘次數及前月累計搭乘金額。

輸出說明

請依照上文的資訊來計算該月的回饋金，並將其輸出（四捨五入至整數位）。

範例輸入1

22 1080

範例輸出1

162

練習題 – 常客優惠方案

```
1  #include <iostream>
2  #include <cmath>
3
4  using namespace std;
5
6  int main() {
7      int cnt, money;
8      float cashback;
9
10     cin >> cnt >> money;
11     if (11 <= cnt && cnt <= 20){
12         cashback = money * 0.1;
13     }
14 }
```

宣告該月搭乘次數cnt、
前月累計搭乘金額money

宣告回饋金cashback

如果 11 <= cnt <= 20

```
15
16     else if (21 <= cnt && cnt <= 30){
17         cashback = money * 0.15;
18     }
19     else if (31 <= cnt && cnt <= 40){
20         cashback = money * 0.2;
21     }
22     else if (41 <= cnt && cnt <= 50){
23         cashback = money * 0.25;
24     }
25     else if (cnt >= 51){
26         cashback = money * 0.3;
27     }
28     cout << round(cashback);
29
30     return 0;
31 }
```

如果 21 <= cnt <= 30

如果 31 <= cnt <= 40

如果 41 <= cnt <= 50

如果 cnt >= 51

輸出四捨五入後的cashback

ZeroJudge介紹

- ZeroJudge 是一個 Online Judge 系統的線上解題系統，於2006年由國立高雄師大附中的江其勳教師所製作，可以讓學習程式語言的使用者（學生、老師、任何人），有題目可以練習，並且可以透過程式的判斷機制，了解自己所寫的程式是否正確。



適合所有中學生及初學者的 Online Judge 系統

AC (Accept): 即表示通過

NA (Not Accept): 在多測資點的題目中若未通過所有測資點則出現 NA

WA (Wrong Answer): 表示答案錯誤，並在訊息中指出錯誤行數及正確答案

TLE (Time Limit Exceed): 表示執行超過時間限制

MLE (Memory Limit Exceed): 表示程序執行超過記憶體限制

OLE (Output Limit Exceed): 表示程序輸出權超過限制

RE (Runtime Error): 表示執行時錯誤，通常為記憶體配置錯誤 如：使用了超過陣列大小的位置

RF (Restricted Function): 表示使用了被禁止使用的函式，並在錯誤訊息中指出使用了什麼不合法的函式。

CE (Compile Error): 表示編譯錯誤，並在訊息中列出完整錯誤訊息，以利判斷。 [關於編譯器](#)

資料來源：

<https://steam.oxxostudio.tw/category/python/zerojudge/about.html>

ZeroJudge介紹



適合所有中學生及初學者的 Online Judge 系統

AC (Accept): 即表示通過

NA (Not Accept): 在多測資點的題目中若未通過所有測資點則出現 NA

WA (Wrong Answer): 表示答案錯誤，並在訊息中指出錯誤行數及正確答案

TLE (Time Limit Exceed): 表示執行超過時間限制

MLE (Memory Limit Exceed): 表示程序執行超過記憶體限制

OLE (Output Limit Exceed): 表示程序輸出檔超過限制

RE (Runtime Error): 表示執行時錯誤，通常為記憶體配置錯誤 如：使用了超過陣列大小的位置

RF (Restricted Function): 表示使用了被禁止使用的函式，並在錯誤訊息中指明使用了什麼不合法的函式。

CE (Compile Error): 表示編譯錯誤，並在訊息中列出完整錯誤訊息，以利判斷。 [關於編譯器](#)

點選登入

ZeroJudge介紹

點選 用Google登入

帳號： 請輸入帳號

密碼： 請輸入密碼

☐

我不是機器人



reCAPTCHA
隱私權 - 條款

登入


忘記密碼？

用 Google 登入

以現有的 Google 帳號創建身份

ZeroJudge介紹

*請使用學校提供的校務信箱登入

 使用 Google 帳戶登入

登入

繼續使用「[zerojudge.tw](#)」

電子郵件地址或電話號碼

1113****@dcsh.tp.edu.tw

[忘記電子郵件地址？](#)

如要繼續進行，Google 會將您的姓名、電子郵件地址、語言偏好設定和個人資料相片提供給「[zerojudge.tw](#)」。

[建立帳戶](#)

繼續

ZeroJudge介紹

A Alpha 2.0 0

Home

A

解除綁定 Google 帳號

登出

即時訊息收件夾

解題統計

修改個人資訊

+ 參加課程

a(兔瓦斯)	2736/ 94%
(王肩圣)	2334/ 80%
nown)	2016/ 70%
(「我們所認識 像天使...)	1709/ 59%
Caido)	1695/ 58%
en@g...	1668/ 58%
(Kevin_Wang	1607/ 55%
s^&*()_...)	1492/ 51%

more

點選 + 參加課程

ZeroJudge介紹

請輸入要加入的「課程代碼」

請向開設課程的使用者索取「課程代碼」

Close參加課程

課程代碼：

aNu9IK

switch case

- **switch case** 可用來比較 數字 或 字元

```
switch(變數或運算式){  
    case 數字或字元:  
        陳述句;  
        break;  
    case 數字或字元:  
        陳述句;  
        break;  
    default:  
        陳述句;  
        break;  
}
```

先與case設的數字或字元做比較
符合條件就會執行對應case後的陳述句

若沒有符合的數字或字元，則執行
default後的陳述句

default可省略

switch case

```
int a = 2;
switch(a){
    case 1: 當 a = 1 時
        cout << "一\n";
        break;
    case 2: 當 a = 2 時
        cout << "二\n";
        break;
}
```

輸出：二

```
char chr = 'h';
switch(chr){
    case 'c': 當 chr = 'c' 時
        cout << "c\n";
        break;
    case 'd': 當 chr = 'd' 時
        cout << "d\n";
        break;
    default: 當 chr 不是以上條件時
        cout << "?\n";
        break;
}
```

輸出：?

switch case

```
int a = 2;
switch(a){
    case 1: 當 a = 1 時
        cout << "一\n";
        break;
    case 2: 當 a = 2 時
        cout << "二\n";
        break;
}
```

輸出：二

```
char chr = 'h';
switch(chr){
    case 'c': 當 chr = 'c' 時
        cout << "c\n";
        break;
    case 'd': 當 chr = 'd' 時
        cout << "d\n";
        break;
    default: 當 chr 不是以上條件時
        cout << "?\n";
        break;
}
```

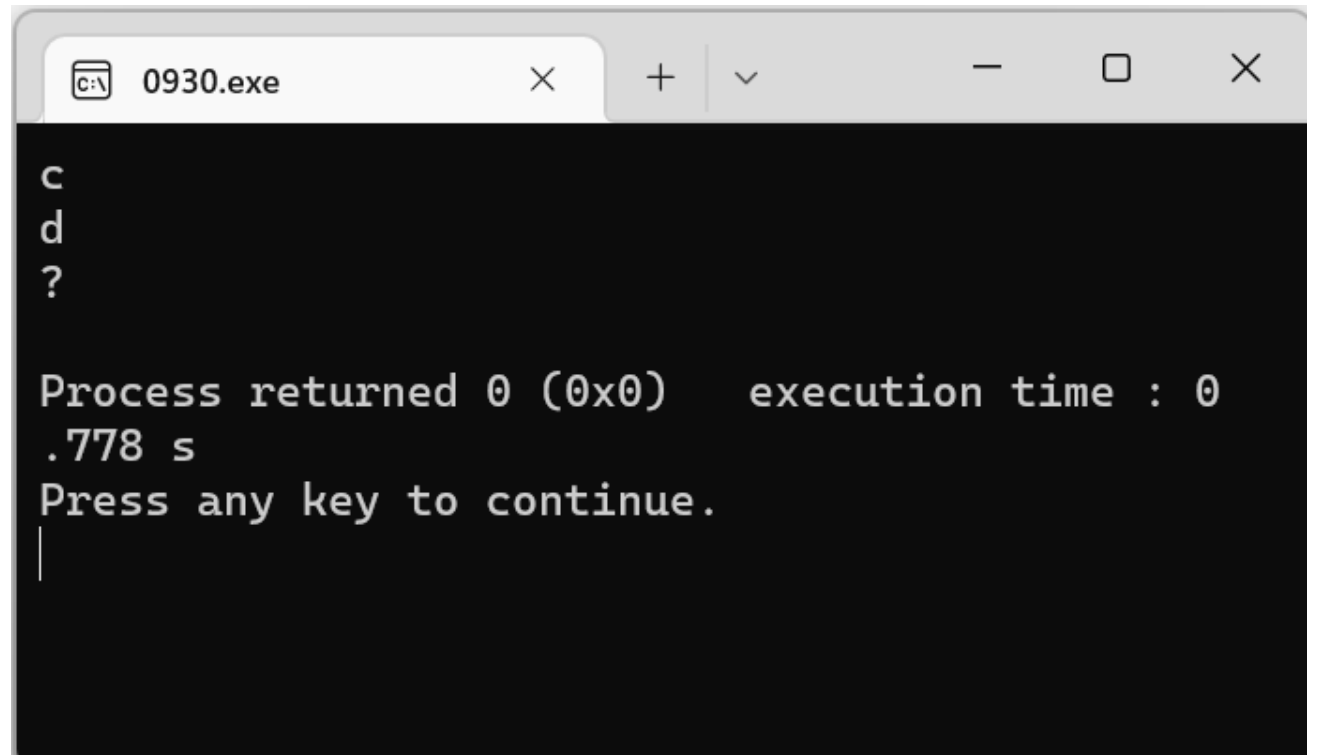
輸出：?

switch case

如果不加break會發生什麼？

它會一直執行下去

```
1  #include <iostream>
2  using namespace std;
3
4  int main(){
5      char chr = 'c';
6      switch(chr){
7          case 'c':
8              cout << "c\n";
9          case 'd':
10             cout << "d\n";
11         default:
12             cout << "?\n";
13     }
14 }
```



```
c
d
?

Process returned 0 (0x0)   execution time : 0.778 s
Press any key to continue.
|
```

switch case

我們可以利用這個特點

```
1  #include <iostream>
2  using namespace std;
3
4  int main(){
5      int a = 6;
6      switch(a){
7          case 10:
8              cout << "S級\n";
9              break;
10         case 9: case 8:
11             cout << "A級\n";
12             break;
```

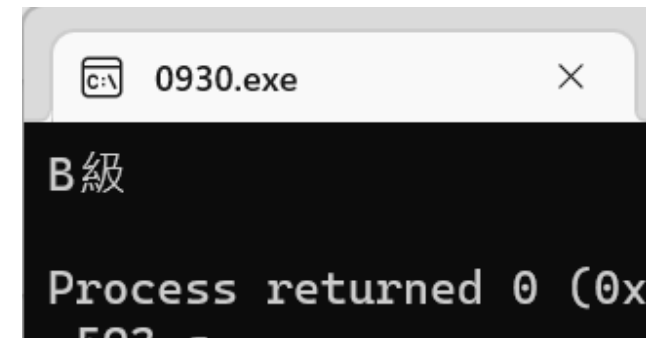
```
13         case 7: case 6: case 5:
14             cout << "B級\n";
15             break;
16         case 4: case 3: case 2:
17             cout << "C級\n";
18             break;
19         default:
20             cout << "D級\n";
21             break;
22     }
23     return 0;
24 }
```


switch case

我們可以利用這個特點

```
1  #include <iostream>
2  using namespace std;
3
4  int main(){
5      int a = 6;
6      switch(a){
7          case 10:
8              cout << "S級\n";
9              break;
10         case 9: case 8:
11             cout << "A級\n";
12             break;
```

```
13         case 7: case 6: case 5:
14             cout << "B級\n";
15             break;
16         case 4: case 3: case 2:
17             cout << "C級\n";
18             break;
19         default:
20             cout << "D級\n";
21             break;
22     }
23     return 0;
24 }
```



a001-哈囉

學習所有程式語言的第一個練習題

請寫一個程式，可以讀入指定的字串，並且輸出指定的字串。

比如：輸入字串 "world", 則請輸出 "hello, world"

輸入說明

輸入總共一行，內含一組文字

輸出說明

輸出題目指定的文字。

範例輸入1

world

範例輸入2

C++

範例輸入3

Taiwan

範例輸出1

hello, world

範例輸出2

hello, C++

範例輸出3

hello, Taiwan

a001-哈囉

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7      string input;
8
9      cin >> input;
10     cout << "hello, " << input;
11
12     return 0;
13 }
```

宣告一個字串input
用於儲存輸入的字串

讀入字串到input

先輸出 "hello, "
在輸出 字串input

a002-簡易加法

請寫一個程式，讀入兩個數字，並求出它們的和。

輸入說明

每組輸入共一行，內含有兩個整數 a, b ，以空白隔開， a, b 絕對值皆小於 10^6 。

輸出說明

對於每組輸入，輸出該兩整數的和。

範例輸入1

5 10

範例輸入2

1 2

範例輸出1

15

範例輸出2

3

a002-簡易加法

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7      int a, b;
8
9      cin >> a >> b;
10     cout << a + b;
11
12     return 0;
13 }
```

宣告整數a, b

讀入整數a, b

輸出 a + b

c379-成為出題者

在Zerojudge中，要成為出題者的條件之一是：至少答對30%的題目。

現在Zerojudge上面有 x 題(x 保證是10的倍數)，請問至少要答對幾題，才能成為出題者？

輸入說明

輸入只有一行，一個正整數 x ($0 < x < 2000$)

輸出說明

輸出答案，已經在題目敘述中敘述

範例輸入1

100

範例輸出1

30

c379-成為出題者

```
1  #include <iostream>
2  #include <cmath>
3
4  using namespace std;
5
6  int main(){
7      int x; 宣告整數x
8
9      cin >> x; 讀入整數x
10     cout << celi(x * 0.3);
11         輸出無條件進位過後的(x * 0.3)
12     return 0;
13 }
```

d064- < 一' 數？

英文的 Odd 是「奇怪」的意思，Odd Number 當然就是「奇怪的數字」簡稱「奇數」。問題是「奇數」到底是唸成「ㄌ一數」或是「< 一' 數」呢？

輸入說明

輸入只有一行，其中含有一個整數 i。

輸出說明

如果 i 是奇數，輸出 Odd；如果 i 是偶數，則輸出 Even。

範例輸入1

1

範例輸出1

Odd

範例輸入2

4

範例輸出2

Even

d064- < 一 ' 數 ?

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main(){
6      int i; 宣告整數i
7
8      cin >> i; 讀入整數i
9      if (i % 2 == 0) i 除 2 於 0
10         cout << "Even"; 輸出"Even"
11     else
12         cout << "Odd"; 輸出"Odd"
13
14     return 0;
15 }
```

a003-兩光法師占卜術

兩光法師時常替人占卜，由於他算得又快有便宜，因此生意源源不絕，時常大排長龍，他想算得更快一點，因此找了您這位電腦高手幫他用電腦來加快算命的速度。

他的占卜規則很簡單，規則是用這樣的，輸入一個日期，然後依照下面的公式：

M=月

D=日

$$S=(M*2+D)\%3$$

得到 S 的值，再依照 S 的值從 0 到 2 分別給與普通、吉、大吉 等三種不同的運勢

輸入說明

輸入資料共一行，包含兩個整數，分別為月份及日期

輸出說明

運勢

範例輸入1

1 1

範例輸出1

普通

範例輸入2

1 2

範例輸出2

吉

a003-兩光法師占卜術

```
1  #include <iostream>
2  using namespace std;
3
4  int main(){
5      int M, D, S;
6      cin >> M >> D;
7      S = (M * 2 + D) % 3;
8  }
```

宣告月份M、日期D、S

輸入月份M 和 日期D

計算S

判斷運勢

```
if (S == 0)
    cout << "普通";
else if (S == 1)
    cout << "吉";
else if (S == 2)
    cout << "大吉";

return 0;
```

j105-常客優惠方案

臺北大眾捷運股份有限公司針對持儲值卡的乘客有常客優惠方案，以下為其基本規則：

1. 持儲值卡搭乘捷運以全票票價扣款，並於票卡累積搭乘次數及搭乘金額。
2. 依每卡每月累計搭乘次數，決定現金回饋比例，並依累計搭乘金額，計算回饋金。

當月回饋金＝前月累計搭乘金額×現金回饋比例 (尾數不滿1元者，按四捨五入計算)	
前月累計搭乘次數	現金回饋比例
11～20次	10%
21～30次	15%
31～40次	20%
41～50次	25%
51次以上	30%

3. 當月搭乘捷運首次通過捷運閘門時，回饋金即以自動加值方式，存入同一張票卡之電子錢包。回饋金等同現金，除可用於搭乘捷運，亦可小額消費。
4. 每月累積期間：自每月1日0時至該月最後1日24時止。
5. 回饋金自動加值有效期間：自當月1日零時起，1年內有效。於有效期間內，首次通過捷運閘門即自動加值，超過有效期間均未搭乘捷運，則尚未存入電子錢包之回饋金，將於到期日之翌日零時起失效歸零。



輸入說明

小明是每天持悠遊卡普通卡（全票票價）使用臺北捷運通勤的乘客，他很好奇他每個月的常客優惠回饋金有多少。

請寫出一支程式幫他計算每個月的常客優惠回饋金。

範例輸入1

22 1080

範例輸出1

162

輸入資料共一行，包含兩個整數並使用空格分開，分別為該月搭乘次數及前月累計搭乘金額。

輸出說明

請依照上文的資訊來計算該月的回饋金，並將其輸出（四捨五入至整數位）。

j105-常客優惠方案

```
1  #include <iostream>
2  #include <cmath>
3
4  using namespace std;
5
6  int main() 宣告該月搭乘次數cnt、
7  {          前月累計搭乘金額money
8      int cnt, money;
9      float cashback;
10         宣告回饋金cashback
11     cin >> cnt >> money;
12     if (11 <= cnt && cnt <= 20){
13         cashback = money * 0.1;
14     } 如果 11 <= cnt <= 20
```

```
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31 }
```

```
else if (21 <= cnt && cnt <= 30){
    cashback = money * 0.15;
} 如果 21 <= cnt <= 30
else if (31 <= cnt && cnt <= 40){
    cashback = money * 0.2;
} 如果 31 <= cnt <= 40
else if (41 <= cnt && cnt <= 50){
    cashback = money * 0.25;
} 如果 41 <= cnt <= 50
else if (cnt >= 51){
    cashback = money * 0.3;
} 如果 cnt >= 51

cout << round(cashback);
return 0; 輸出四捨五入後的cashback
```

d066-上學去吧！

板橋高中規定同學必須在 7:30 以前到校早自習，最後一堂課則在 17:00 下課。給你現在的時間，請判斷現在是不是必須在學校的時間。

輸入說明

輸入只有一行，其中含有兩個由空隔開的整數 hh 及 mm，hh:mm 則代表現在的時間 (24小時制)。

輸出說明

如果現在是上學時間，請輸出「At School」，否則請輸出「Off School」

範例輸入1

17 00

範例輸出1

Off School

d066-上學去吧！

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main(){
6      int hh, mm;
7
8      cin >> hh >> mm;
9
10     if ((hh == 7 && mm >= 30) || (8 <= hh && hh <= 16)){
11         cout << "At School";
12     }
13     else{
14         cout << "Off School";
15     }
16
17     return 0;
18 }
```

宣告整數
hh, mm,
用於紀錄
時和分

讀入整數hh, mm

時間是 07:大於30 或 時間是 08:00 ~ 16:59

輸出"Off School"

i 除 2 於 0

輸出"At School"

a053-Sagit's 計分程式

sagit 是一位高中電腦老師，這學期正在教學生寫C++程式。他的評分標準是依照每一位學生在 ZeroJudge 系統上解出的題數，去計算出對應的得分。為了不讓分數落差太大，因此他並不是採取每一題固定得分的方式，而是隨著題數增加而調整每題的得分。規則如下：

答對題數在 0~10 者，每題給6分。

題數在 11~20 者，從第11題開始，每題給2分。(前10題還是每題給6分)

題數在 21~40 者，從第21題開始，每題給1分。

題數在 40 以上者，一律100分。

如此一來，只要寫10題，就可以得到60分，寫20題，就可以得到80分，不過要得到滿分100分，則是要寫到40題，所以同學們分數的差距就大大地減少了。

不過問題來了，雖然學生們因為這樣的計分公式而大大地提升了及格率，但因為 sagit 有600多位學生，一個一個去計算真的是一件很吃重的工作，所以現在想請你幫他寫個程式解決這個問題。

輸入說明

每組測資只有一個整數 N ($0 \leq N \leq 100$)，代表學生在 ZeroJudge 系統上解出的題數。

輸出說明

印出該位同學的得分。

範例輸入1

10

範例輸出1

60

範例輸入2

40

範例輸出2

100

a053-Sagit's 計分程式

```
1  #include <iostream>
2  using namespace std;
3
4  int main(){
5      int N, point = 0;
6
7      cin >> N;
8
9      if (0 <= N && N <= 10)
10         point = N * 6;
11
12
13
14
15
16
17
18
19
20
21 }
```

宣告答對題數N、
總分point = 0

輸入答對題數N

如果 $0 \leq N \leq 10$

如果 $11 \leq N \leq 20$

```
else if (11 <= N && N <= 20)
    point = 10 * 6 + (N - 10) * 2;
else if (21 <= N && N <= 40)
    point = 10 * 6 + 10 * 2 + (N - 20) * 1;
else
    point = 100;

cout << point;

return 0;
```

如果 $21 \leq N \leq 40$