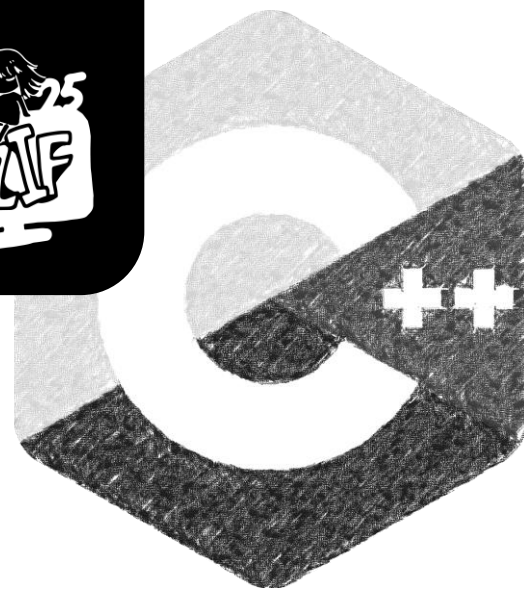




C++探索

0930資訊社





cout 輸出

```
1  #include <iostream>
2  using namespace std;
3
4  int main () {
5      cout << "iPhone 14 Pro\n";
6      cout << "Pro. Beyond." << endl;
7      return 0;
8  }
```

<< 後面放輸出的東西

利用 \n 或 endl 換行



cout 輸出

```
1  #include <iostream>
2  using namespace std;
3
4  int main() {
5      cout << "iPhone 14 Pro\n";
6      cout << "Pro. Beyond." << endl;
7      return 0;
8  }
```

<< 後面放輸出的東西

利用 \n 或 endl 換行

```
0930.exe
iPhone 14 Pro
Pro. Beyond.

Process returned 0 (0x0)   execution time : 0.468 s
Press any key to continue.
```



cout 輸出

```
1    #include <iostream>
2    using namespace std;
3
4    int main () {
5        cout << "日期\t活動\n";
6        cout << "10/13\t第一次段考\n";
7        cout << "10/14\t第一次段考\n";
8        cout << "11/05\t校慶\n";
9        return 0;
10 }
```

利用 \t 移動到下一個tab位置



cout 輸出

```
1  #include <iostream>
2  using namespace std;
3
4  int main () {
5      cout << "日期\t活動\n";
6      cout << "10/13\t第一次段考\n";
7      cout << "10/14\t第一次段考\n";
8      cout << "11/05\t校慶\n";
9      return 0;
10 }
```

利用 \t 移動到下一個tab位置



日期	活動
10/13	第一次段考
10/14	第一次段考
11/05	校慶



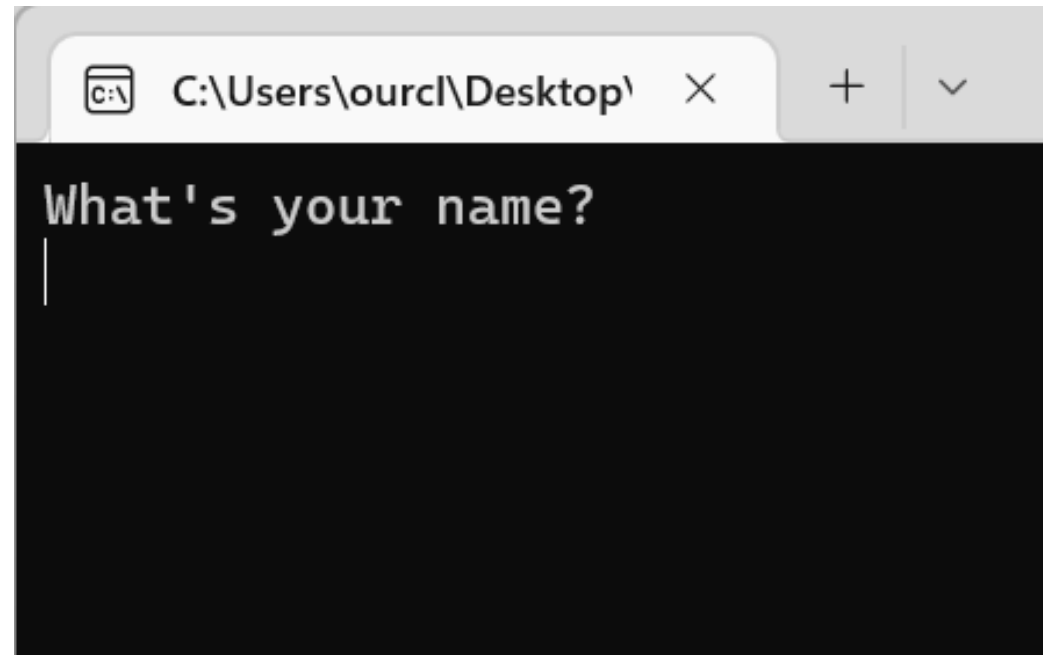
cin 輸入

```
1    #include <iostream>
2    using namespace std;
3
4    int main() {
5        string name; 宣告一個名為name的string(字串)
6        cout << "What's your name?\n";
7        cin >> name;  >> 後面放輸入到的地方
8        cout << "Hello, " << name << "!\n";
9        return 0;
10 }
```



cin 輸入

```
1    #include <iostream>
2    using namespace std;
3
4    int main () {
5        string name;
6        cout << "What's your name?\n";
7        cin >> name;
8        cout << "Hello, " << name << "!\n";
9        return 0;
10   }
```



```
C:\Users\ourcl\Desktop\ x + v
What's your name?
|
```

宣告一個名為name的string(字串)

>> 後面放輸入到的地方



int 整數

大小：4 bytes

範圍： $-2^{31} \sim 2^{31} - 1$ ($-2,147,483,648 \sim 2,147,483,647$)

例如：8、4858938、-24537、379072、0



long 長整數

大小：4 bytes (32位元電腦)

範圍：同 **int**

例如：8、4858938、-24537、379072、0

大小：8 bytes (64位元電腦)

範圍： $-2^{63} \sim 2^{63} - 1$

(-9,223,372,036,854,775,808 ~ 9,223,372,036,854,775,807)

例如：8、4858938、-24537、379072、0



long long 長整數

大小：8 bytes

範圍： $-2^{63} \sim 2^{63} - 1$

($-9,223,372,036,854,775,808 \sim 9,223,372,036,854,775,807$)

例如：8、4858938、-24537、379072、0



float 單精度浮點數

大小：4 bytes

範圍： $\pm 1.1754943 \cdot 10^{-38} \sim \pm 3.4028234 \cdot 10^{38}$

保證精準位數：6 位

例如：23.15、3.1415、-341.343、0.00003



double 倍精度浮點數

大小：8 bytes

範圍： $\pm 2.2250738585072014 \cdot 10^{-308} \sim$
 $\pm 1.7976931348623157 \cdot 10^{308}$

保證精準位數：15 位

例如：23.1500900338564、-1.0000000000000001、2.45683



bool 布林

大小：1 bytes

範圍： $-2^7 \sim 2^7 - 1$ ($-128 \sim 127$)

例如：**true** (1)、**false** (0)

0 = **false**

非0 = **true**



char 字元

大小：1 bytes

範圍： $-2^7 \sim 2^7 - 1$ ($-128 \sim 127$)

例如：'a'、'B'、'c'、'\n'、'\t'




string 字串


例如： "Hello, World!" 、 "Never gonna give you up" 、
"今天天氣晴"



= 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

$$A = B \% C$$

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

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



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

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

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

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

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

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

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

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



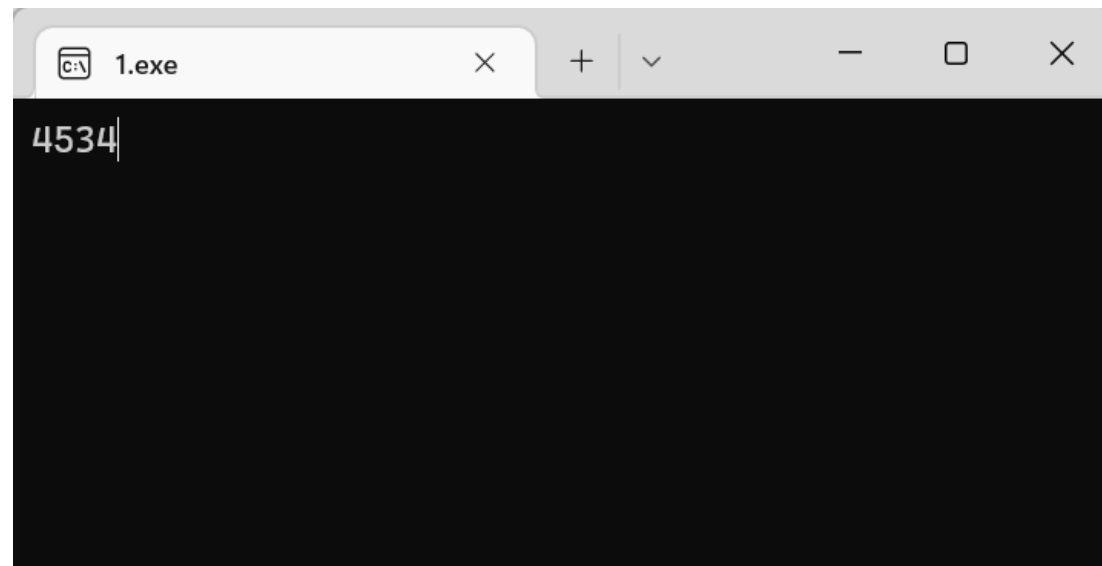
幫我算數學

```
1  #include <iostream>
2  using namespace std;
3
4  int main () {
5      int a, b;
6      宣告二個int來存輸入的二個int
7      cin >> a >> b;  利用cin >>來輸入兩個 int 到 a 和 b 中
8      cout << a + b << endl;
9  }  輸出 a + b
```



幫我算數學

```
1  #include <iostream>
2  using namespace std;
3
4  int main () {
5      int a, b;
6      宣告二個int來存輸入的二個int
7      cin >> a >> b;  利用cin >>來輸入兩個 int 到 a 和 b 中
8      cout << a + b << endl;
9  }  輸出 a + b
```





幫我算數學 – 除法

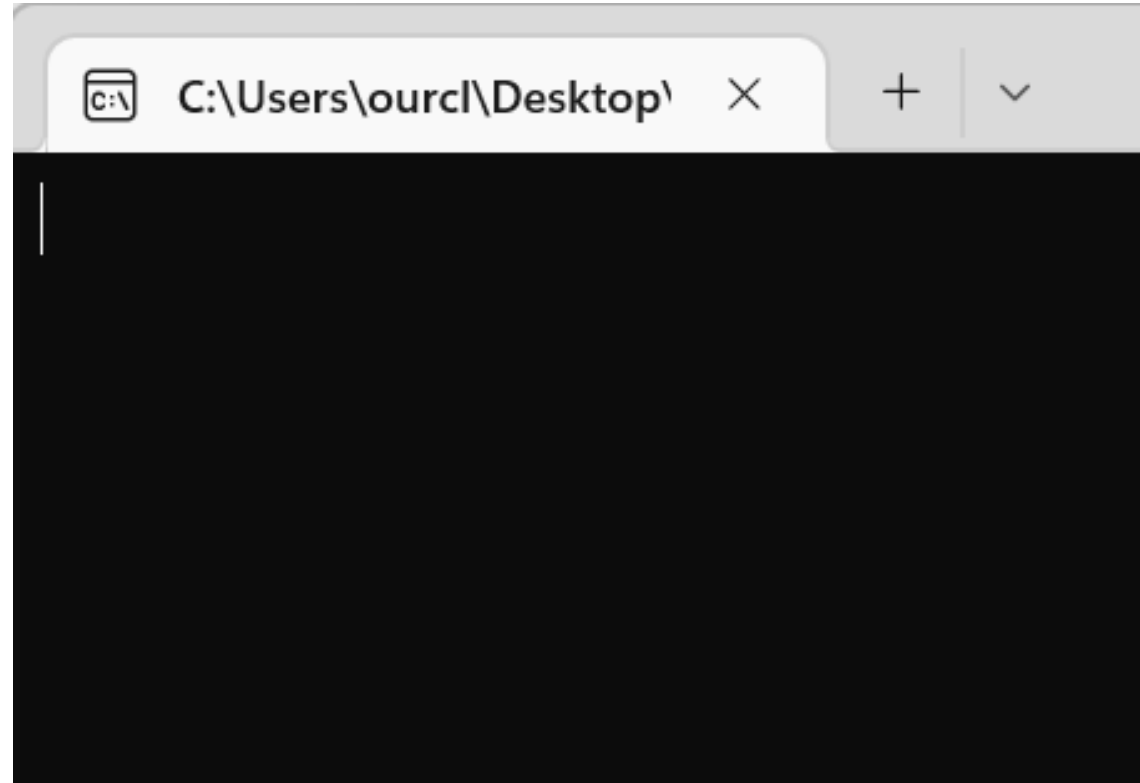
```
1  #include <iostream>
2  using namespace std;
3
4  int main () {
5      int a, b;
6      宣告二個int來存輸入的二個int
7      cin >> a >> b;  利用cin >>來輸入兩個 int 到 a 和 b 中
8      cout << a / b << endl;
9  }  輸出 a / b
```



幫我算數學 – 除法

```
1    #include <iostream>
2    using namespace std;
3
4    int main () {
5        int a, b;
6
7        cin >> a >> b;
8        cout << a / b << endl;
9        }
```

int int



整數 除 整數 是無條件捨去到整數位

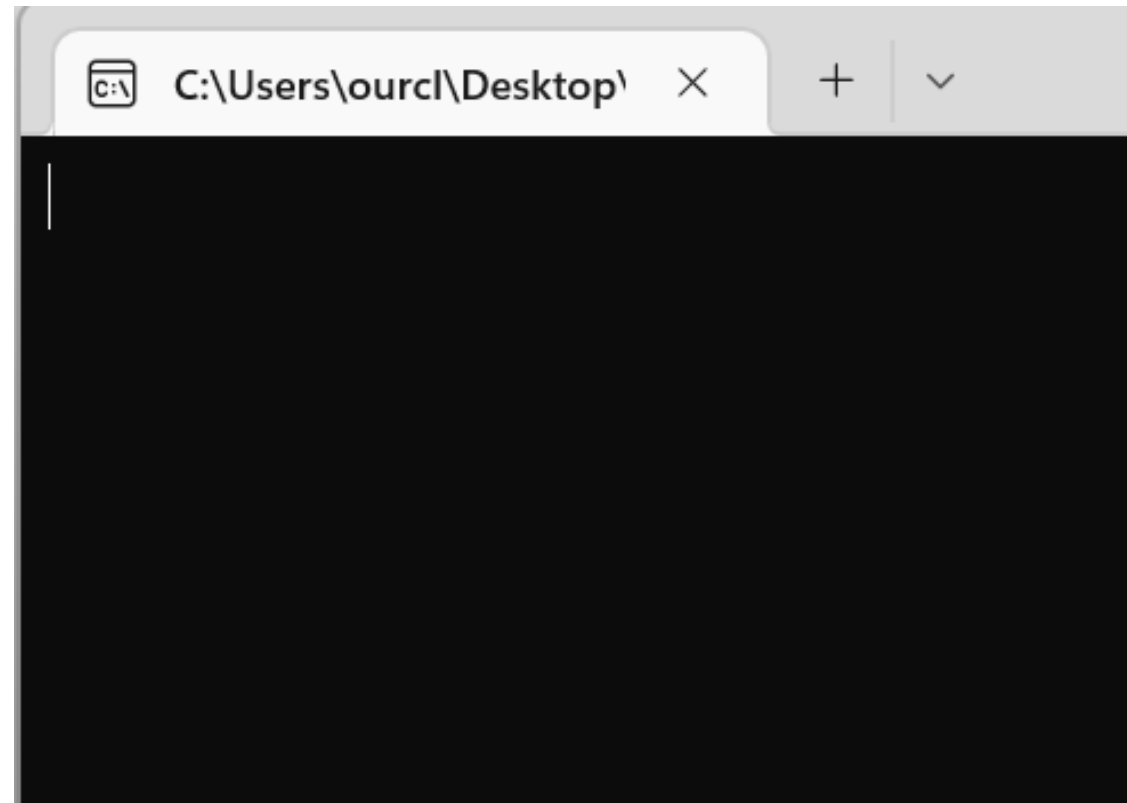


幫我算數學 – 除法

```
1  #include <iostream>
2  using namespace std;
3
4  int main () {
5      int a;
6      float b;
7      cin >> a >> b;
8      cout << a / b <<
9      }
```

宣告 int 和 float
來存輸入的數字

int float



可以利用 浮點數 來做除法運算

int / float -> float

float / int -> float

float / float -> float

E.g.



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

- 抉擇的問題：

如果今天下雨，我要帶雨傘。

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;
```

如果條件式成立
(等於true)



執行程式區塊



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;  
}
```

如果條件式成立
(等於true)



執行程式區塊



if 敘述

- 抉擇的問題：

如果今天下雨，我要帶雨傘。

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 (條件式)
程式區塊;

條件

if (今天下雨)
我帶雨傘;

動作/
敘述

條件

if (成績 < 60)
不及格;

動作/
敘述



if...else 敘述

- 抉擇的問題：

如果今天下雨，我要帶雨傘，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;
```

如果條件式

成立(true)

執行程式區塊A

不成立
(false)

執行程式區塊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;  
}
```

如果條件式

成立(true)

執行程式區塊A

不成立
(false)

執行程式區塊B



if...else 敘述

- 抉擇的問題：

如果今天下雨，我要帶雨傘，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

if (今天下雨)

條件

我帶雨傘；

成立時的動作/敘述

else

不用帶雨傘；

不成立時的動作/敘述



if...else 敘述

- 抉擇的問題：

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 (成績 < 60)

條件

不及格;

成立時的動作/敘述

else

及格;

不成立時的動作/敘述



if...else if 敘述

- 抉擇的問題：

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。



if …else if 敘述

• 抉擇的問題：

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，

條件

動作/敘述

否則如果年紀 < 18，則可看非限制級的影片，

否則則可看各級影片。



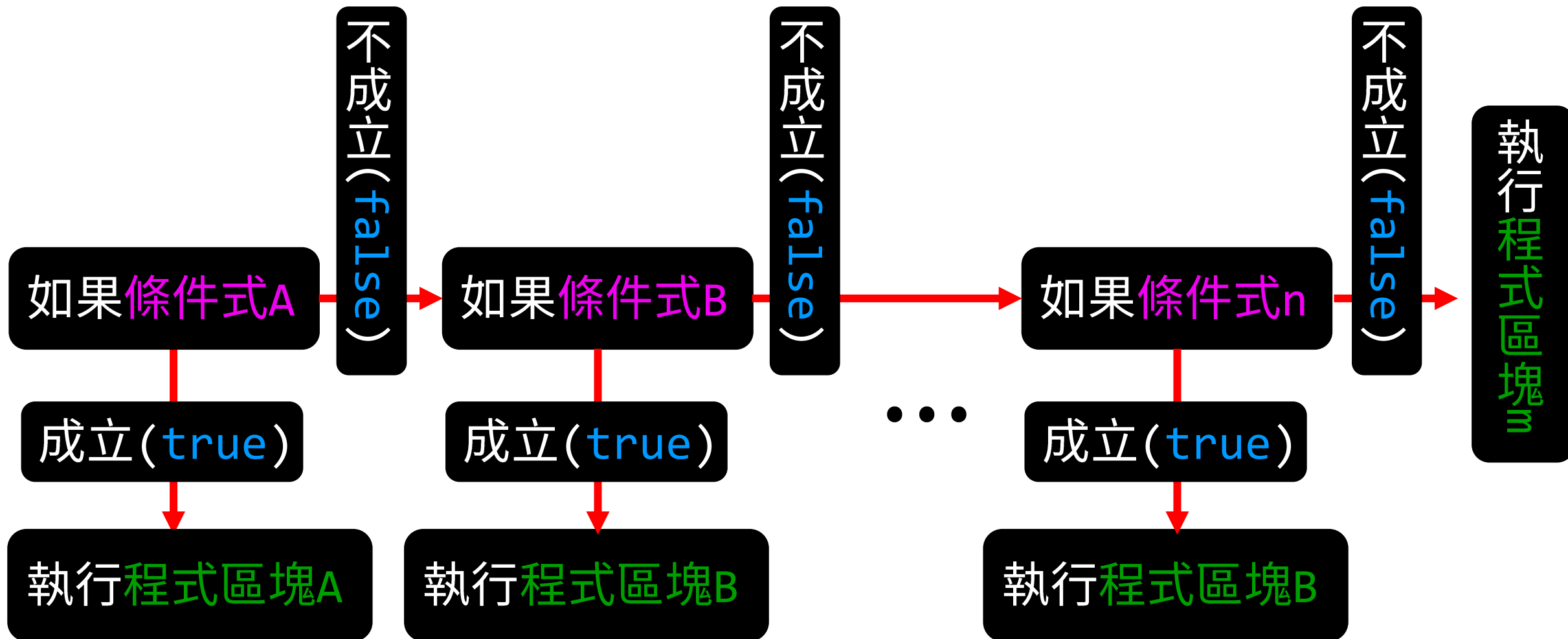
if …else if 敘述

在C++中，可以使用：

```
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





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;
```



```
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 - 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

範例輸入2

40

範例輸出1

60

範例輸出2

100



練習題 - 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)
```

如果 $21 \leq N \leq 40$

```
    point = 10 * 6 + 10 * 2 + (N - 20) * 1;
```

```
else
```

```
    point = 100;
```

```
cout << point;
```

```
return 0;
```

```
}
```



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 }
```

```
0930.exe
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
14         case 7: case 6: case 5:
15             cout << "B級\n";
16             break;
17         case 4: case 3: case 2:
18             cout << "C級\n";
19             break;
20         default:
21             cout << "D級\n";
22             break;
23     }
24     return 0;
}
```

