

零基礎上手! C語言實戰班



黃熠程 Miller

- 2020 中央大學大氣物理學博士候選人
- 2018-2021 安吉氣象資訊 – 專案經理
- 2021-2022 和碩集團 – 軟體工程師
- 2022-now 陽明海運 – 網站全端工程師

程式語言

- C/C++, C#, Fortran, Python, MATLAB, JavaScript, HTML5/CSS

專長

- 數值模擬, 深度學習, 軟體開發, 網站開發

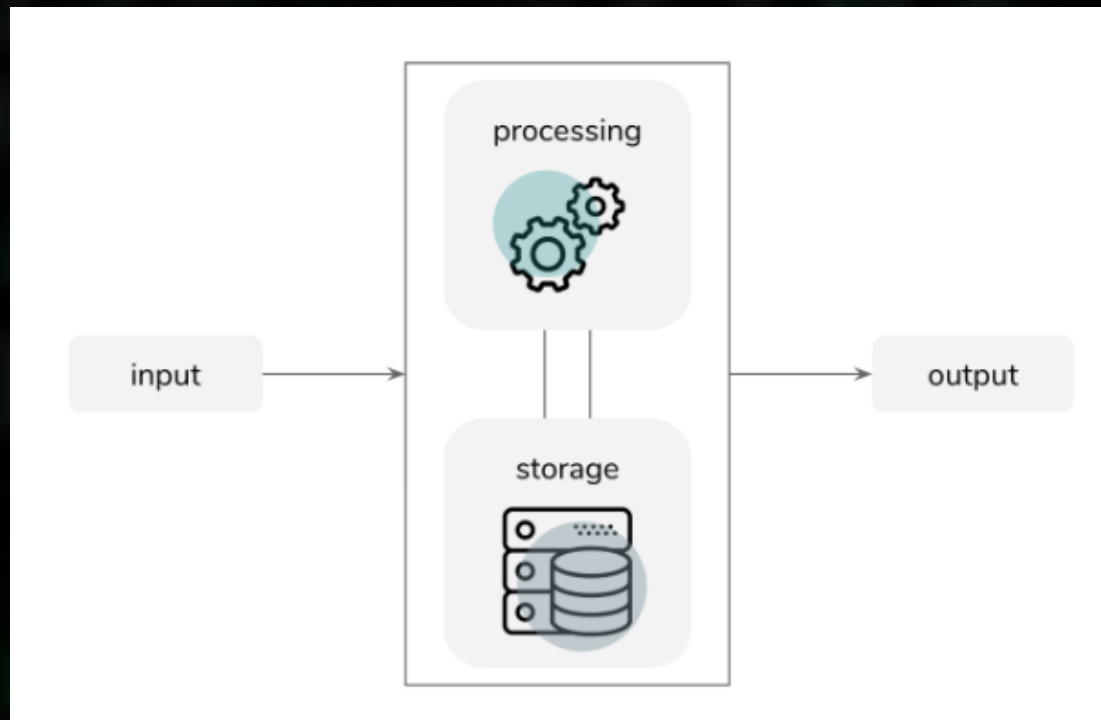
課程簡介



程式設計

程式是什麼

Ans: 用來控制電腦的指令



從我們生活的玩樂到商業只要有用到電腦運算的叫做 **程式**

為什麼需要程式

Ans: 計算量(重複性)超出人腦負荷時, 需要依靠電腦

1. 計算全台灣指考考生的物理平均
2. 計算本校這學期的平均遲到人數
3. 設計定時紅綠燈標誌
4. ...

“程式語言” 是什麼？











Ans: 人類用來跟機器溝通的工具



“程式語言” 怎麼選？

Ans: 看你的需求

目標	應用	語言
系統程式設計	作業系統	C
網路後端應用程式	資料庫、網站架構	PHP、Ruby、Python
網路前端應用程式	所有網頁的視覺編排	JavaScript、HTML、CSS
Android 應用程式	Android App	Java、Kotlin
Apple iOS 應用程式	iOS App	Swift、Objective-C
資料分析	大數據、文本分析	R、Python

Jun 2022	Jun 2021	Change	Programming Language		Ratings	Change
1	2	▲		Python	12.20%	+0.35%
2	1	▼		C	11.91%	-0.64%
3	3			Java	10.47%	-1.07%
4	4			C++	9.63%	+2.26%
5	5			C#	6.12%	+1.79%
6	6			Visual Basic	5.42%	+1.40%
7	7			JavaScript	2.09%	-0.24%
8	10	▲		SQL	1.94%	+0.06%
9	9			Assembly language	1.85%	-0.21%
10	16	▲		Swift	1.55%	+0.44%

<https://tw.alphacamp.co/blog/programming-for-beginners>

<https://www.tiobe.com/tiobe-index/>

Why C

- C 語言是一個高階語言
- C 語言具有可攜性(編譯過程)
- C語言可以寫出較有效率的程式

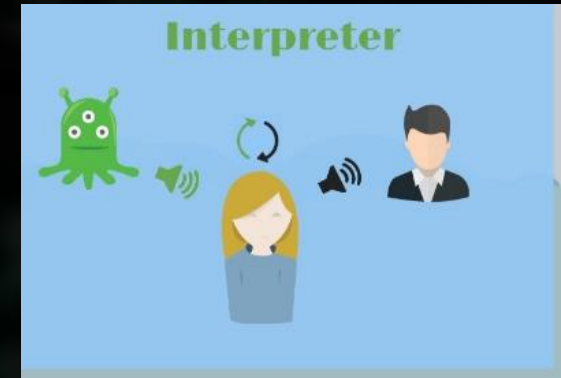
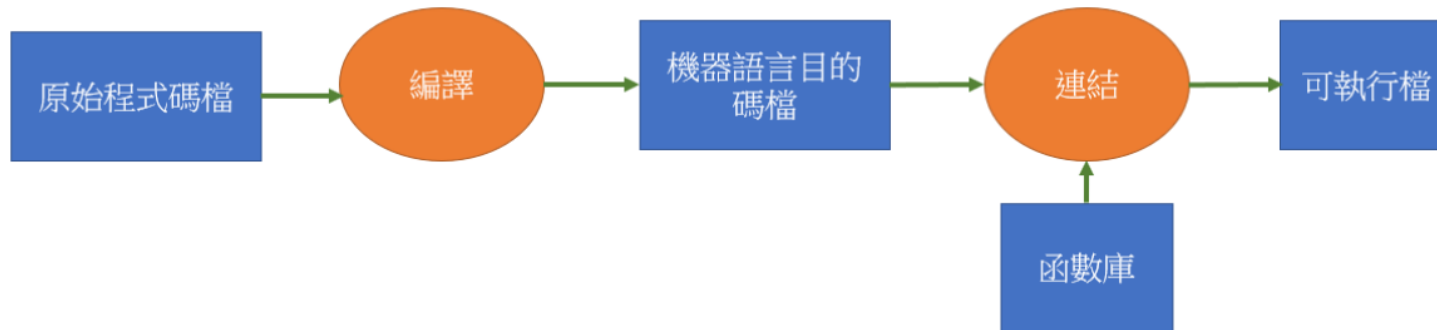
直譯 vs 編譯

Python作為一種直譯語言，並不需要編譯器，因此不會輸出可執行的檔案，直譯式與編譯語言運行流程如下：

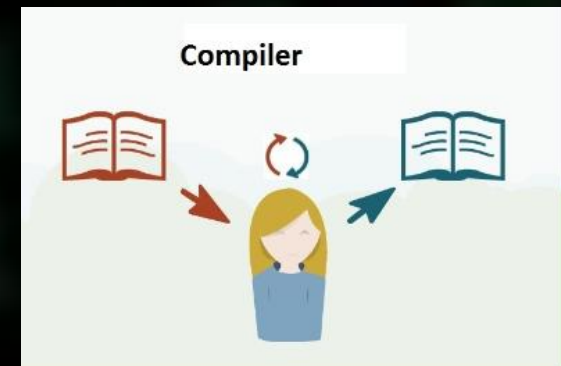
- 直譯式語言



- 編譯式語言



直譯器就像一個口譯者
所見即所得



編譯器就像一個翻譯者
換成另一個語言

Why C

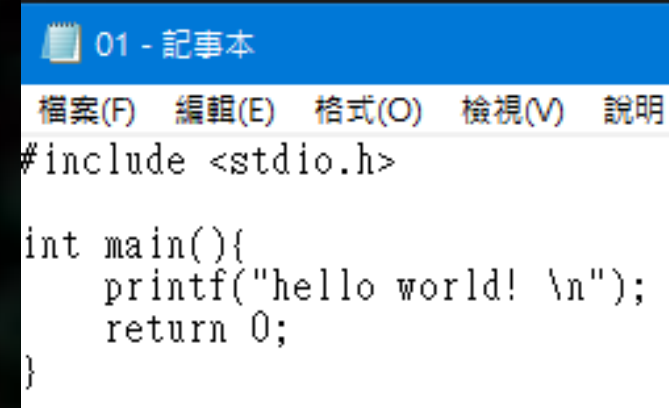
- C 語言是一個高階語言
- C 語言具有可攜性(編譯過程)
- C語言可以寫出較有效率的程式

開發環境

怎麼開始寫C語言?

```
#include <stdio.h>

int main(){
    printf("hello world! \n");
    return 0;
}
```



01 - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明

```
#include <stdio.h>

int main(){
    printf("hello world! \n");
    return 0;
}
```

文字編輯器

原始碼檔

編譯器

可執行檔

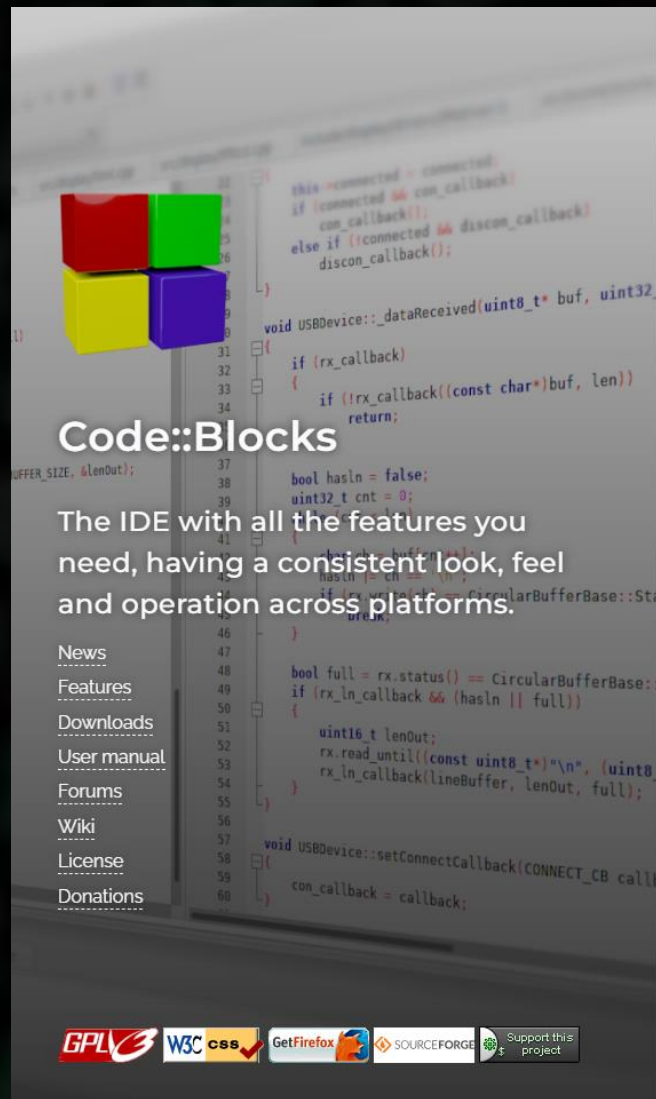
需要至少兩個工具:

1. 文字編輯器
2. C語言編譯器

整合開發環境

- 具有圖像化界面
- 內建編譯器與文字編輯器
- 自動建置工具

Code::Blocks



Code::Blocks

The IDE with all the features you need, having a consistent look, feel and operation across platforms.

[Code::Blocks / Downloads](#)

Downloads

There are different ways to download and install Code::Blocks on your computer:

- [Download the binary release](#)

This is the easy way for installing Code::Blocks. Download the setup file, run it on your computer and Code::Blocks will be installed, ready for you to work with it. Can't get any easier than that!

- [Download a nightly build](#)

There are also more recent so-called nightly builds available in the [forums](#). Please note that we consider nightly builds to be stable, usually, unless stated otherwise.

- Other distributions usually follow provided by the community (big "Thank you!" for that!). If you want to provide some, make sure to announce in the forums such that we can put it on the official C::B homepage.

- [Download the source code](#)

If you feel comfortable building applications from source, then this is the recommend way to download Code::Blocks. Downloading the source code and building it yourself puts you in great control and also makes it easier for you to update to newer versions or, even better, create patches for bugs you may find and contributing them back to the community so everyone benefits.

- [Retrieve source code from SVN](#)

This option is the most flexible of all but requires a little bit more work to setup. It gives you that much more flexibility though because you get access to any bug-fixing we do at the time we do it. No need to wait for the next stable release to benefit from bug-fixes!

Besides Code::Blocks itself, you can compile extra plugins from contributors to extend its functionality.

Thank you for your interest in downloading Code::Blocks!

See also

<https://www.codeblocks.org/downloads>

第一支C程式

The First C Code

程式進入點

; 代表程式語句結束

從這裡開始執行

```
#include <stdio.h>
```

```
int main(){
```

```
printf("hello world! \n");
```

```
return 0;
```

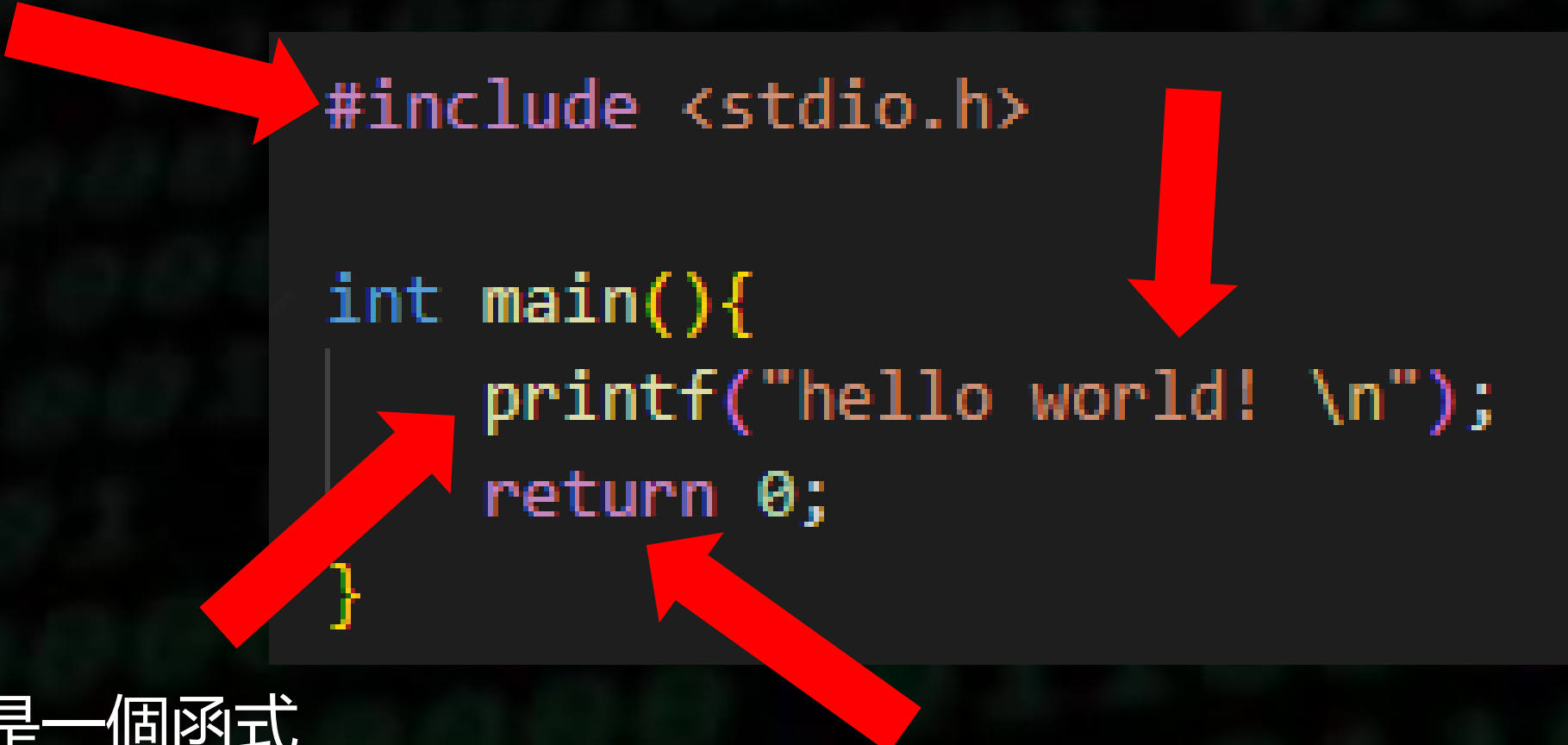
```
}
```

} 代表一個程式區塊

The First C Code

讓編譯器找到 printf()

"" 代表字串



```
#include <stdio.h>

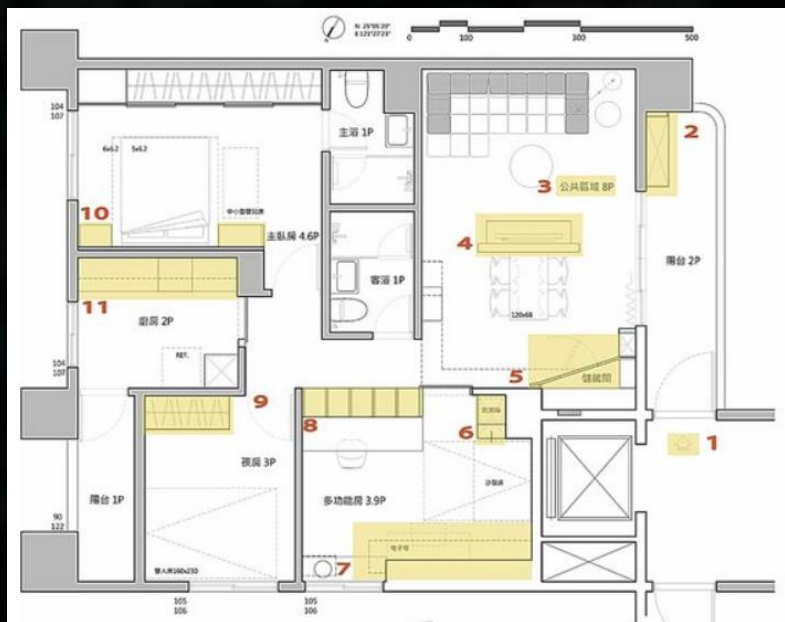
int main(){
    printf("hello world! \n");
    return 0;
}
```

printf() 是一個函式

return 回傳結果

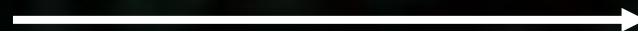
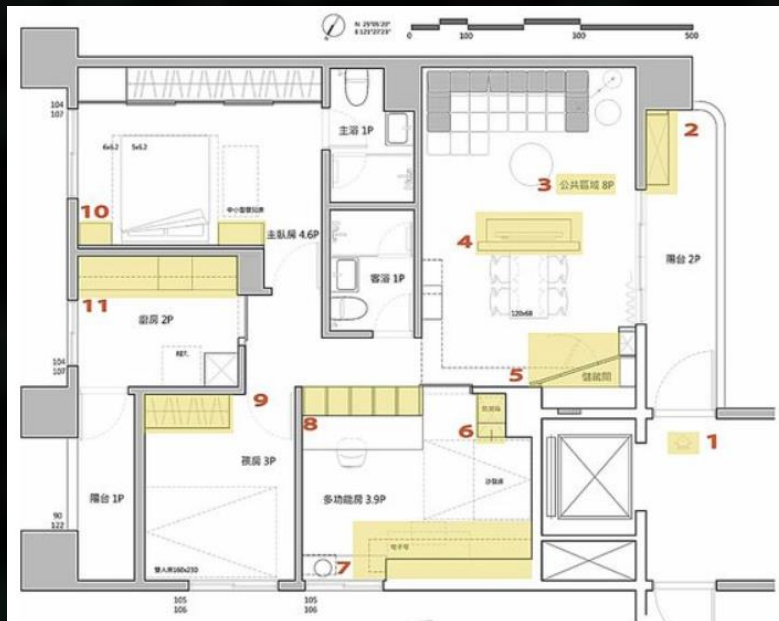
是一段別人寫好的程式碼

變數宣告



想像一下，你手上有一個設計藍圖，你可以根據這個藍圖，蓋出任意外觀的房子





型別
type

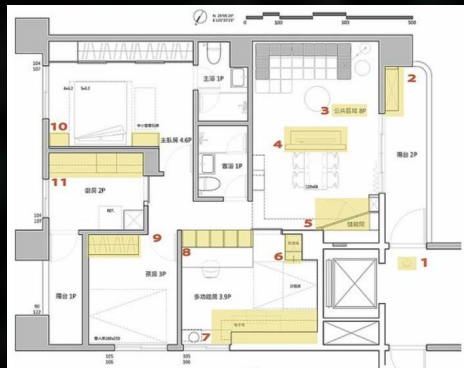


→ 變數名稱

型別(type)

變數名稱

描述：value



500 坪



100 坪

型別(type)

物件(object)/變數(variable)

描述 : value

int

integerObj

10

宣告

指派

C++
C#
Java

```
Int integer01 = 10
```

```
float ff01 = 10
```

```
char c01 = 'A'
```

靜態宣告

型別(type)

integer

變數(variable)

integerObj

描述: value

10

宣告

指派

靜態宣告

```
integer a = 10  
integer a = 20  
Integer b = 10
```

10



Position 1

20



Position 1

10



Position 2

記憶體

每宣告完一個物件(變數), 即為該物件開一個記憶體位置。
修改物件(變數)的value, 即是對該記憶體位置的value做修改。

變數命名

- 變數名稱大小寫有別
- 變數名稱不得是系統內的保留字: void, int, for ...
- 變數名稱必須由英文、數字與底線（_）組成
第一個字元必須以英文或是底線開頭、中間不能有空白
- 變數名稱提供程式記憶體位置，用來儲存資料在記憶體(memory)內
- 所有變數使用前都必須事先宣告
- 變數名稱必須賦予有意義的命名

Try it !!!

範例: Ch2

- 整數
- 浮點數
- 字元
- C++ 觀點: 整數, 浮點數, 字元
- C vs C++ 輸入與輸出

運算子

數學計算

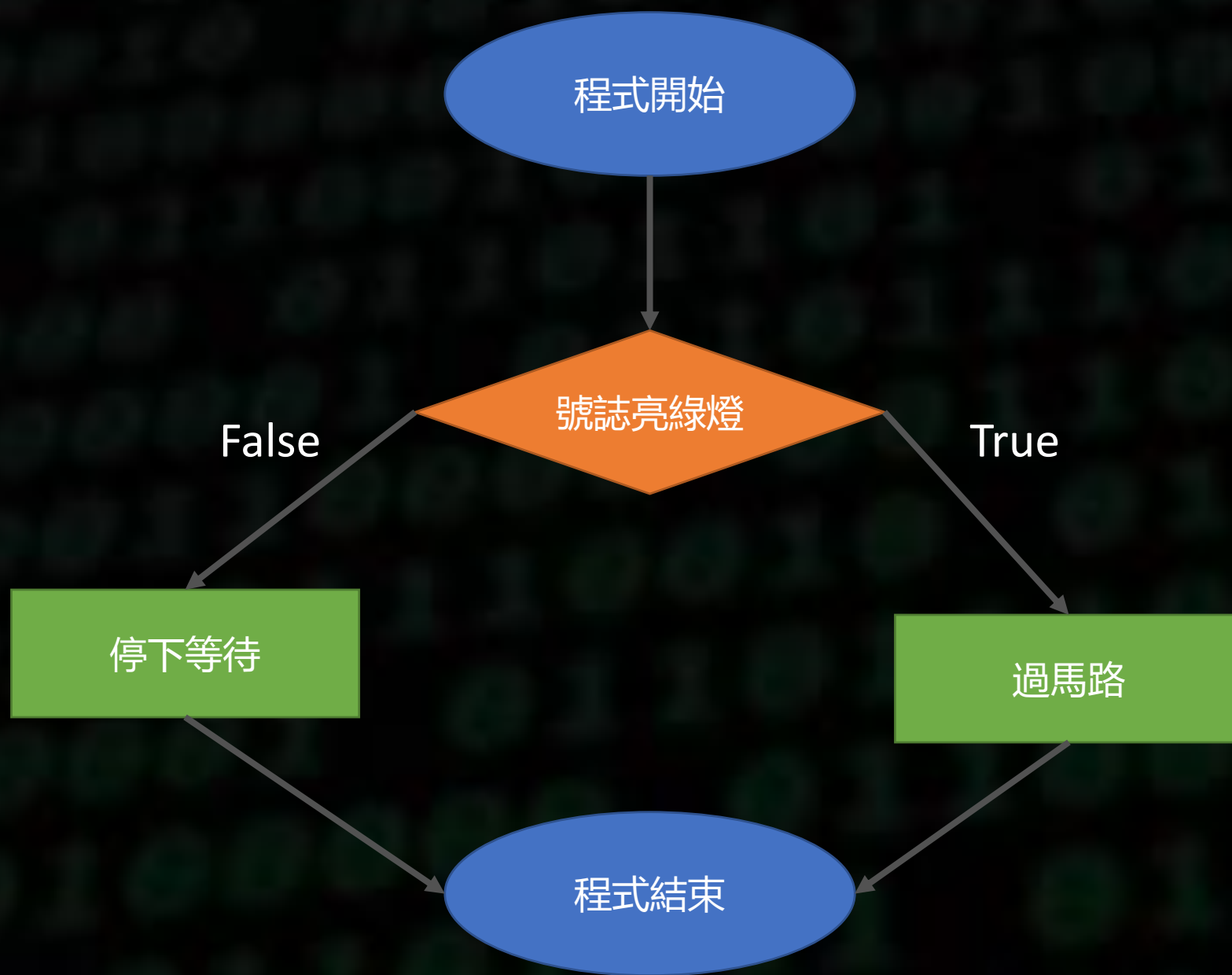
數學符號	功能
+	加法
-	減法
*	乘法
/	除法
%	取餘數

- 所有數學運算符都可以混合使用
- 遵守四則運算規則

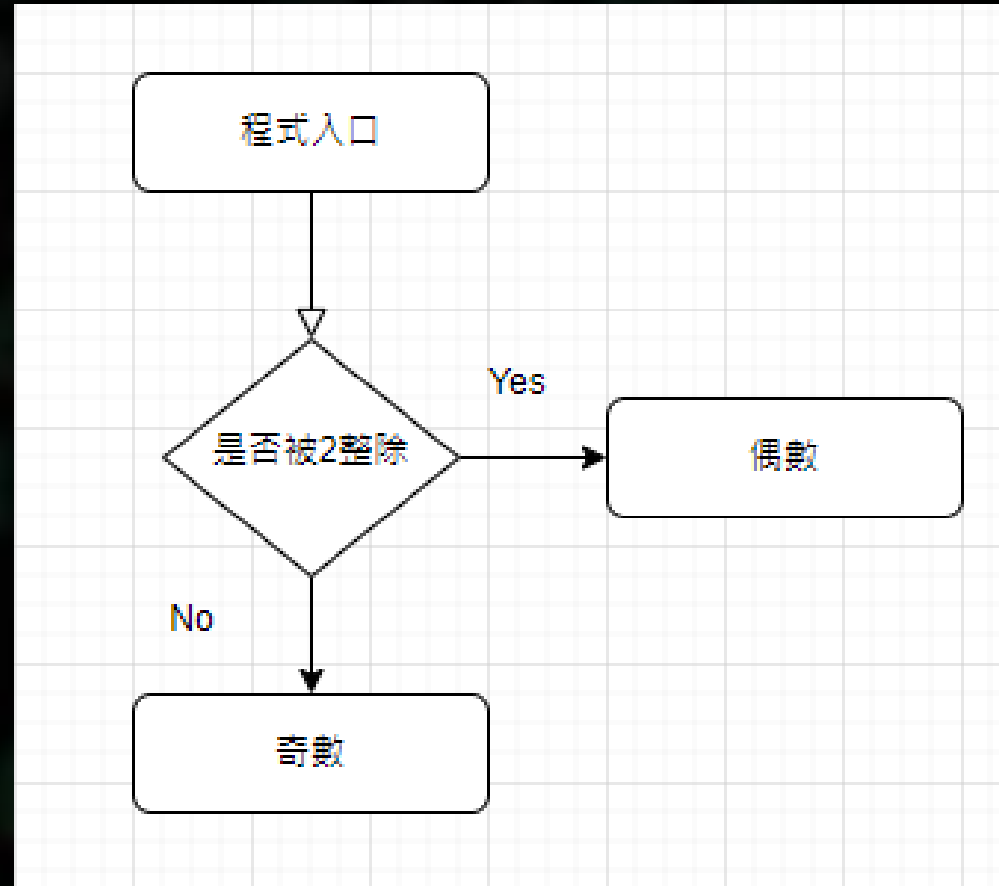
- 1.c : 簡單計算機
- 2.c : 溫度轉換機
- 3.c : 餘數練習
- 4.c : 關係 & 邏輯運算子
- 5.c : 計算考試成績
- 6.cpp : c 與 c++ 的計算圓面積

Try it !!!

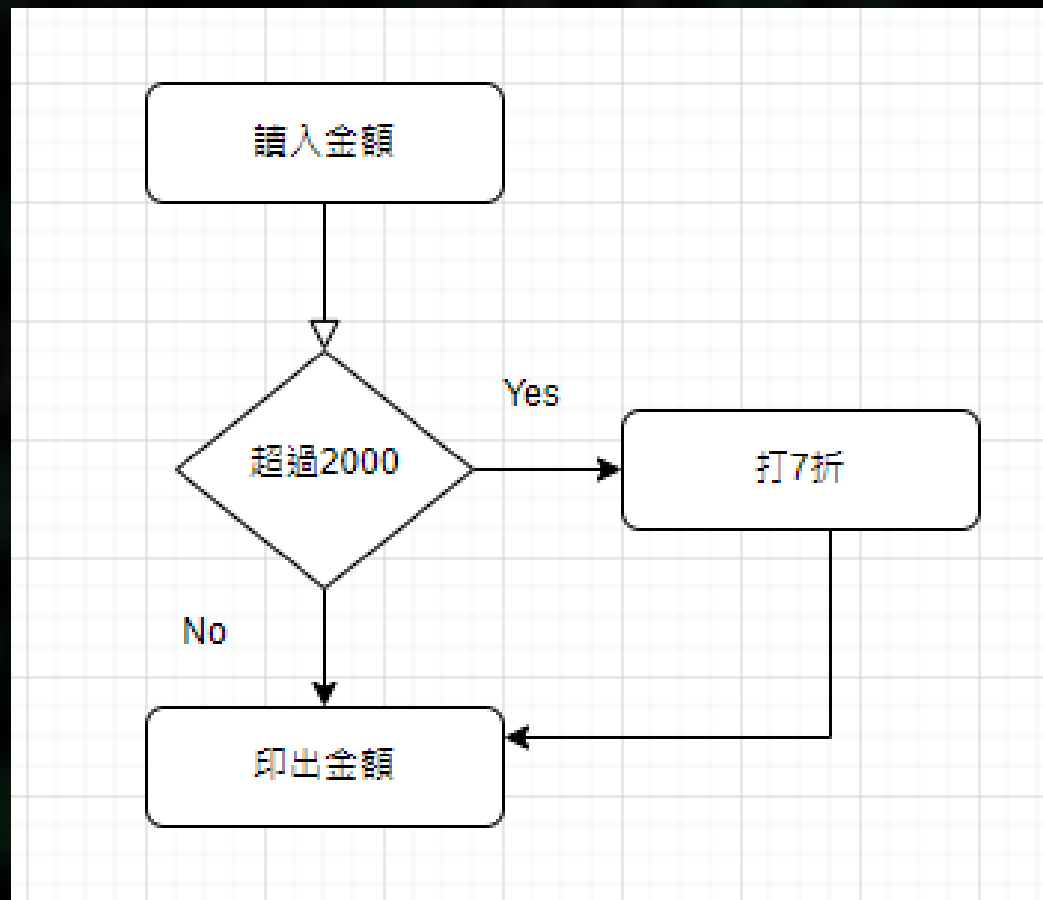
流程敘述



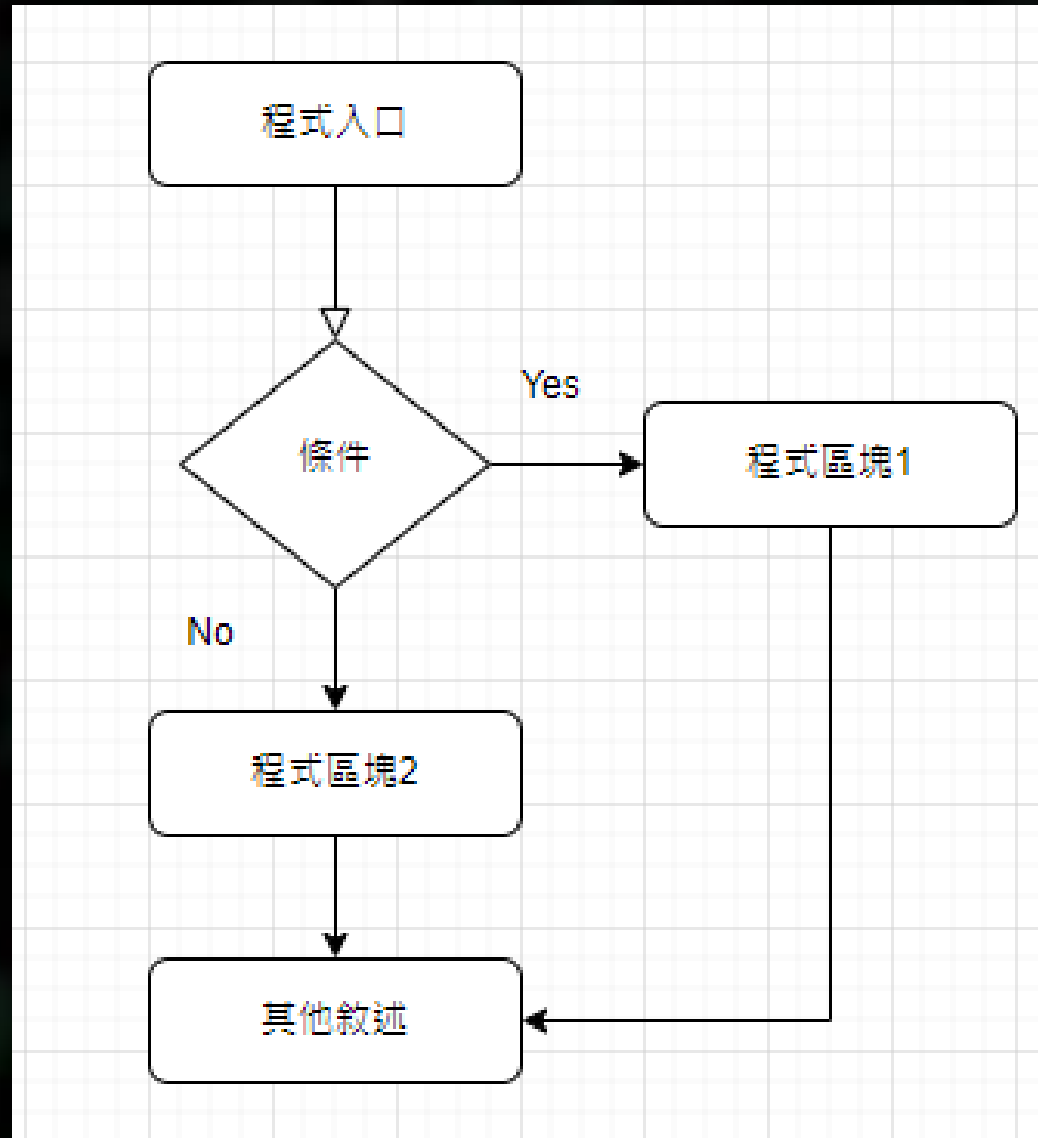
```
if (條件式) {  
    程式區塊  
}
```



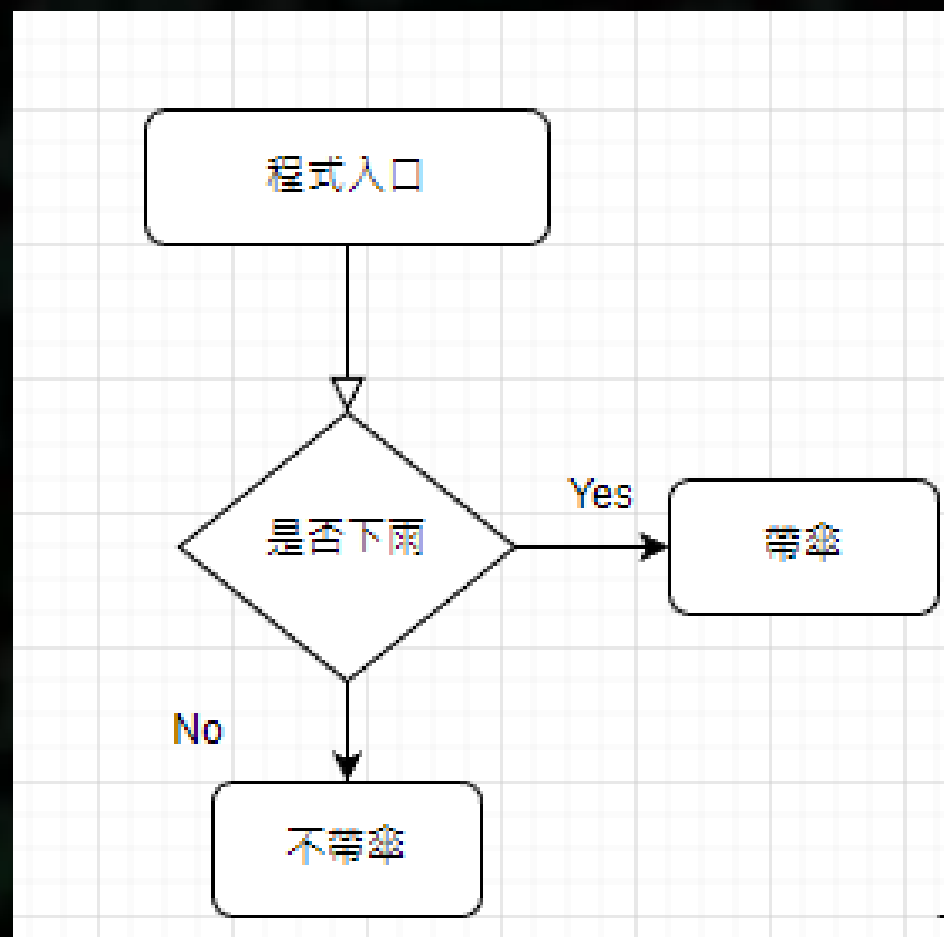
米勒百貨公司周年慶，
公司決定在周年慶期間對消費超過2000元的顧客打7折，
請設計一個收銀台程式，輸入顧客購買金額後，
再計算實際要付的錢



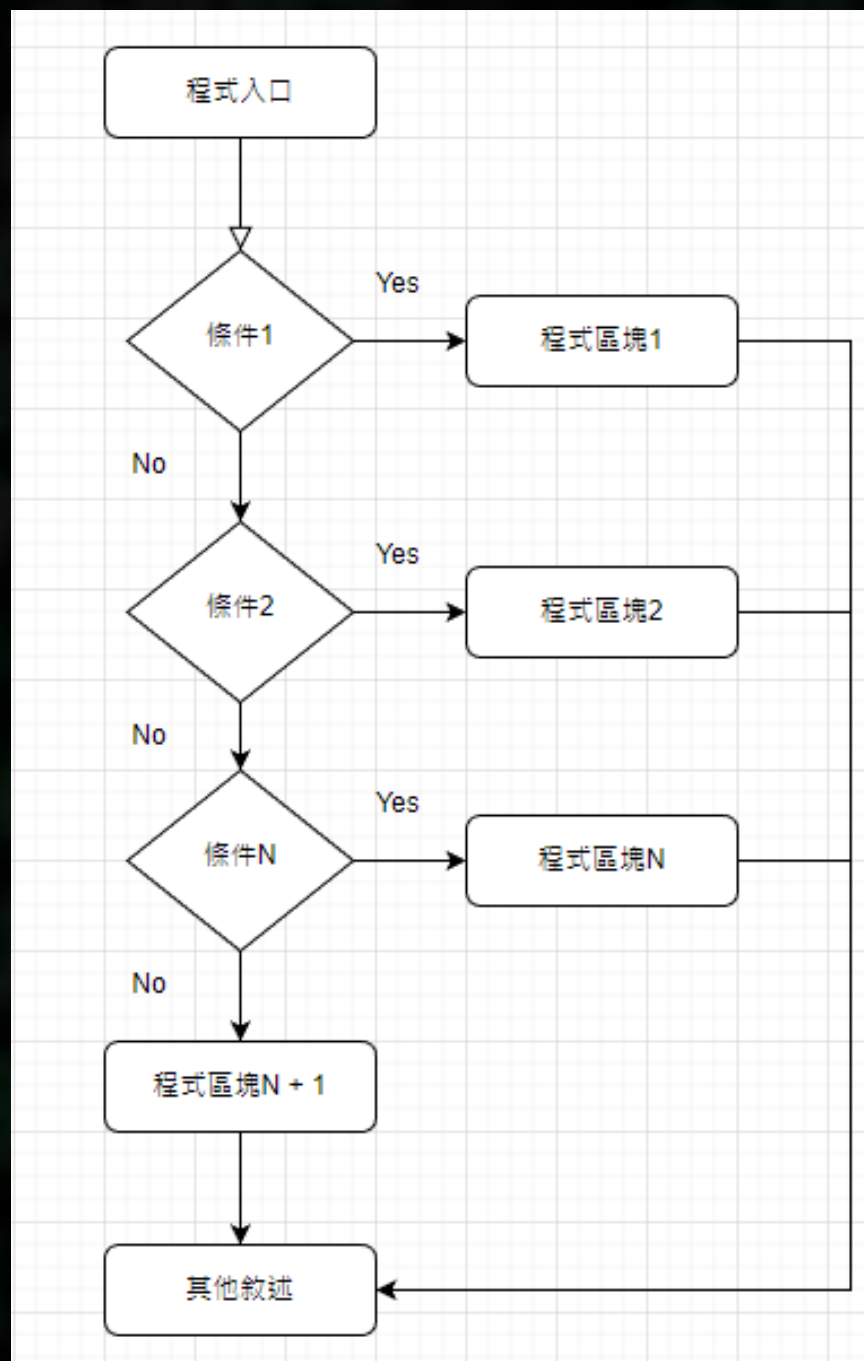
```
if (條件式) {  
    程式區塊 1  
} else {  
    程式區塊 2  
}  
//若條件成立，則執行程式區塊1  
// 反之，執行程式區塊2
```



請寫出一個程式，
由使用者輸入今天是否下雨，
若下雨，則由程式提醒帶雨傘；反之則不帶傘



```
if (條件式 1) {  
    程式區塊 1  
} else if (條件式 2){  
    程式區塊 2  
...  
}else{  
    程式區塊 N  
}  
//若條件1成立，則執程式區塊1  
//若條件2成立，則執程式區塊2  
//若條件都不成立，執程式區塊N
```



請依據使用者輸入的成績，判定成績等第

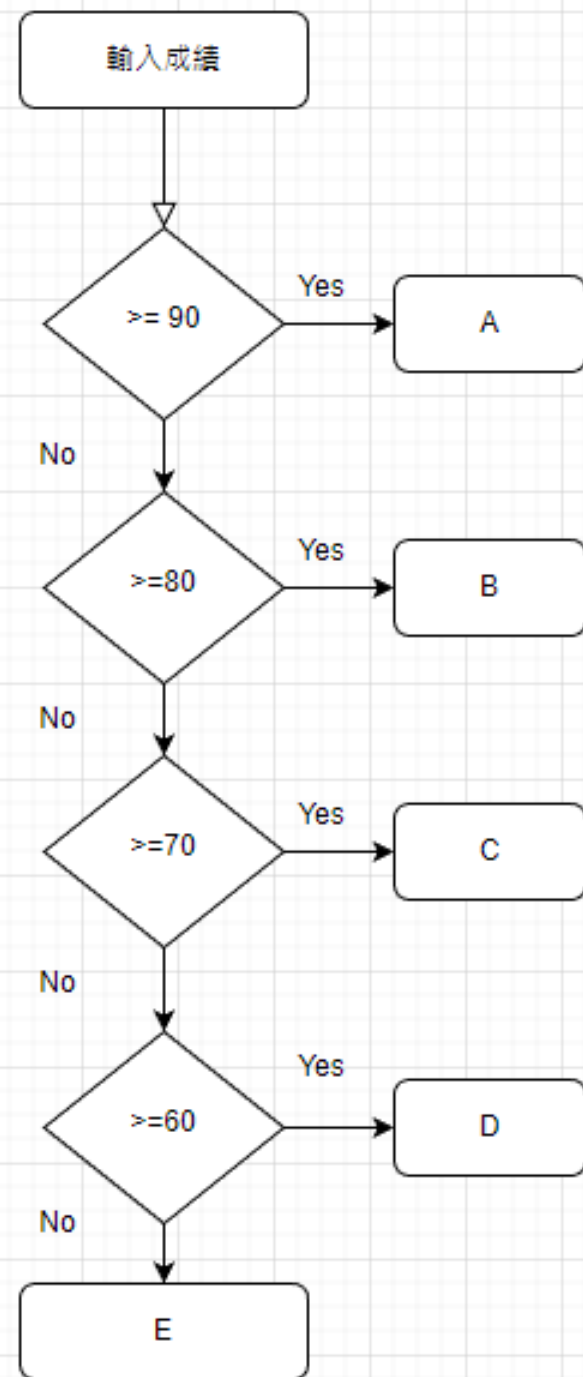
90以上：A

80 - 89：B

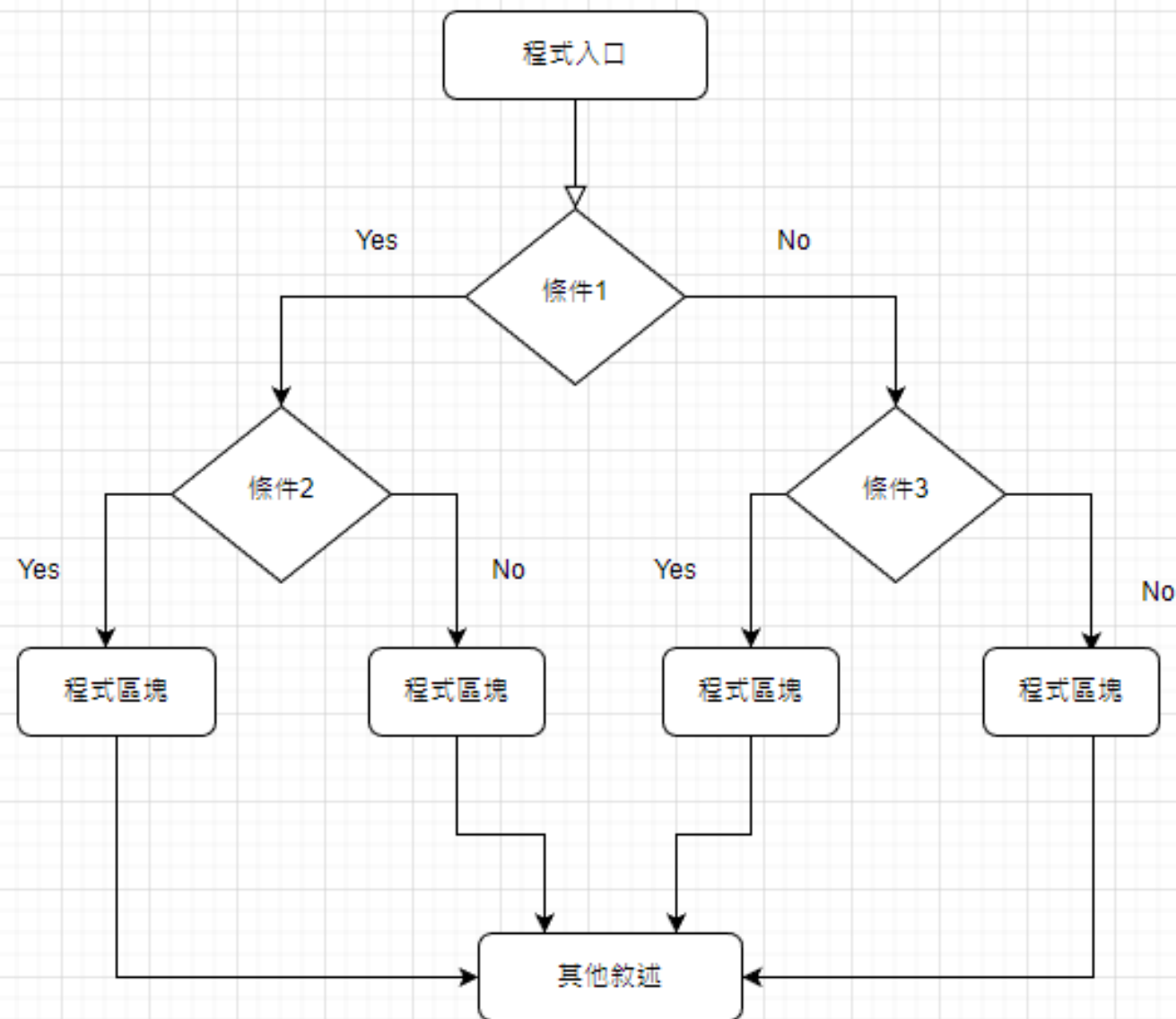
70 - 78：C

60 - 69：D

未滿60：E



```
if (條件式 1.1) {  
    //<若 condition 1.1 為真，則執行此區塊>  
    if(條件式 2){  
        //<若 條件式 2 為真，則執行此區塊>  
    }else{  
        //<若 條件式 2 為假，則執行此區塊>  
    }  
} else if (條件式 1.2){  
    #<若 condition 1.2 為真，則執行此區塊>  
    if(條件式 3){  
        //<若 條件式 3 為真，則執行此區塊>  
    }else{  
        //<若 條件式 3 為假，則執行此區塊>  
    }  
}
```

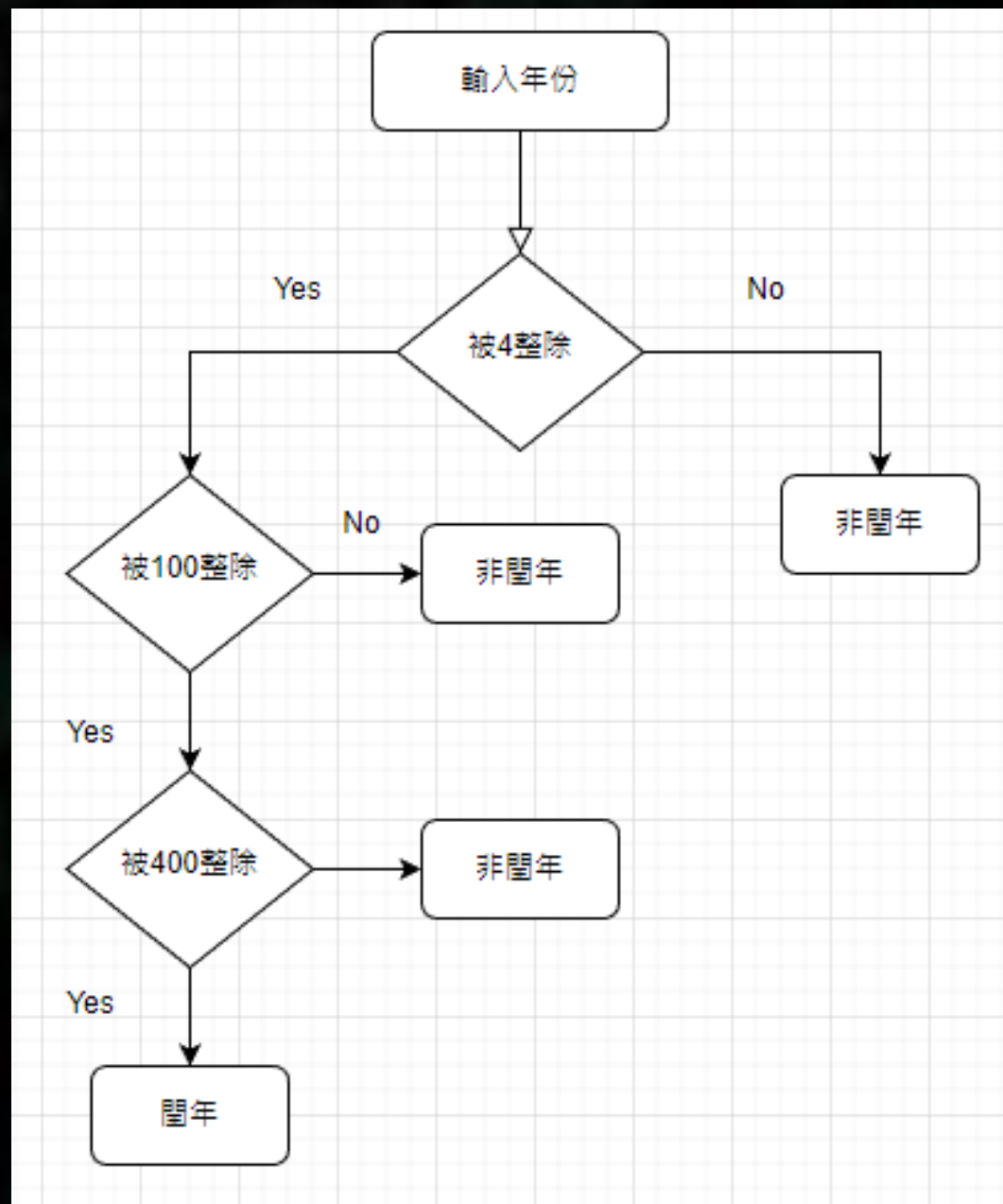


請寫出一個程式，讓使用者輸入西元年分，
判斷該年是否為閏年
判斷方式：四年一閏，逢百不閏，逢四百又閏

ex:

輸入 2000 : 顯示 Leap Year

輸入 2100 : 顯示 Not leap Year



```
switch(變數或運算式){  
    case 值1:  
        程式區塊 1;  
        break;  
    case 值2:  
        程式區塊 2;  
        break;  
    ...  
    case 值N:  
        程式區塊 N;  
        break;  
    default:  
        程式區塊 N+1;  
}
```

