2017

程式設計

加強班

程式設計與實習(一)

BY 孫茂勛

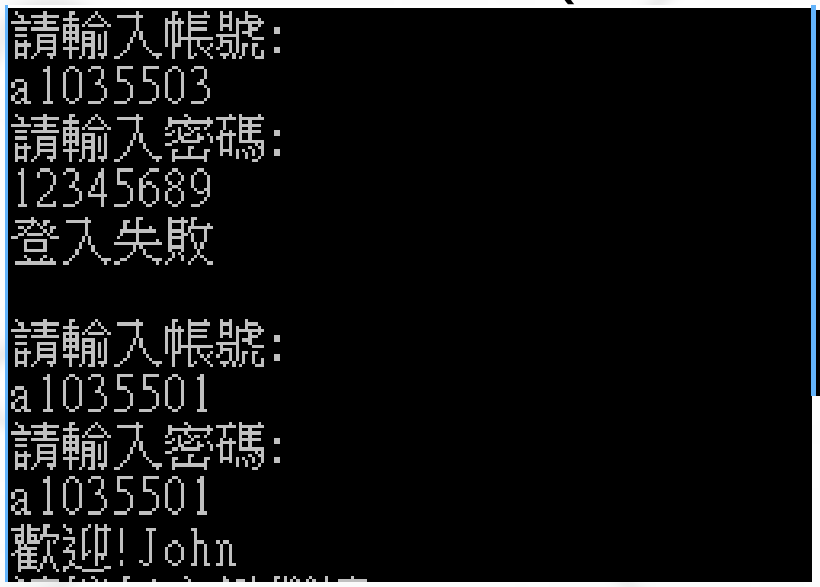
Email:JOHN85051232@GMAIL.COM

練習

Q：模擬一般正常的登入功能，每個人都有自己的帳號密碼，登入成功後根據不同的人會印出

＂歡迎! 使用者名字＂

- 至少設計3個以上不同的使用者(每個使用者有不同的名稱、帳號、密碼)



我們不說課本的CH9的東西～

- 第九章在上什麼?

%c	以字元 方式輸出
%d	10 進位整數輸出
%o	以8進 位整數方式輸出
%u	無號整數輸出
%x, %X	將整 數以16進位方式輸出
%f	浮點 數輸出
%e, %E	使用科學記號顯示浮點數
%g, %G	浮點數輸出，取%f或%e（%f或%E），看哪個表示精簡
%%	顯示 %
%s	字串輸出

想想看

- Q：一個學生的資料有學生成績、姓名、學號、性別.....
- 要如何去寫一個程式來記錄每位學生的這些資料?

```
Char student_name[10] = {0};
Char student_sex[10] = {0};
Char student_grade[10] = {0};
Char student_ssn[10] = {0};
```

如果今天要記錄10位學生呢?

```
Char student_name[10][10] = {0};
Char student_sex[10][10] = {0};
Char student_grade[10][10] = {0};
Char student_ssn[10][10] = {0};
//用strcpy來紀錄第一位學生的資料
strcpy(student_name[0],"john");
strcpy(student_sex[0],"boy");
strcpy(student_grade[0],"1");
strcpy(student_ssn[0],"A1035501");
....
```

這樣的寫法很沒有效率、不夠明確、也較不容易寫。

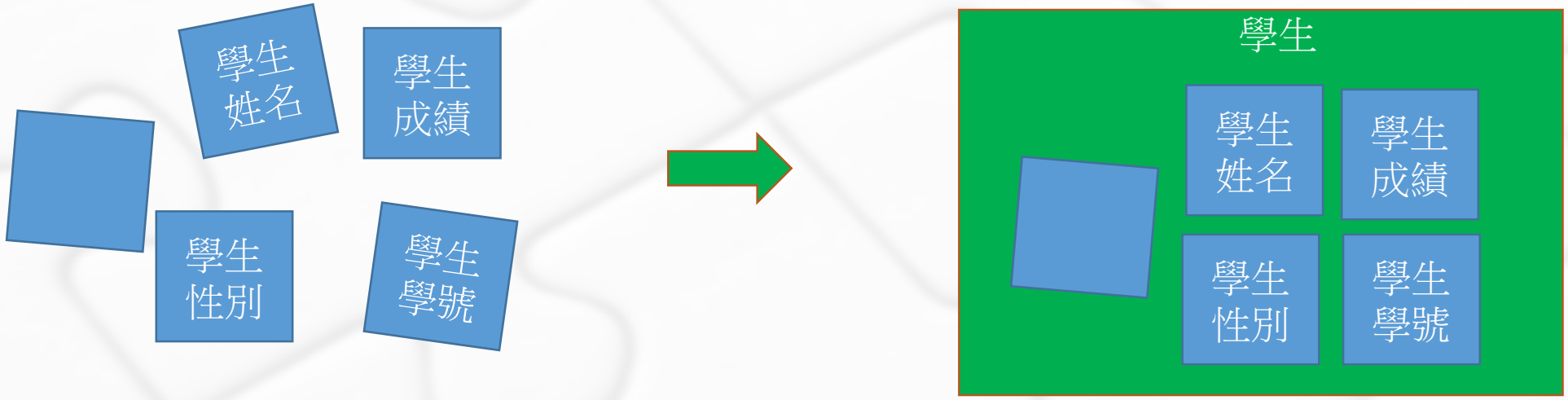
- 到目前為止我們的程式碼都是圍繞著變數在撰寫的。
- 有些情況，我們會想要以更大的單位來進行程式碼撰寫。

- Ex：當你想操控的是一台汽車(物件)，而不是汽車的零件(變數)。

OOP(補充)

物件導向(Object-Oriented Programming)

- 大二上學期會教
- 把變數結合再一起變成物件(Object)的概念
- 可以先想成把許多變數用一個更大的變數包裝再一起

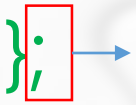


Struct

Struct(結構)

- 一種資料結構
- 可以包含不同資料型態的變數
- Struct要定義後才能使用，定義結構時不用做變數的初始化
- 定義方式：

```
struct 名稱
{
};
```



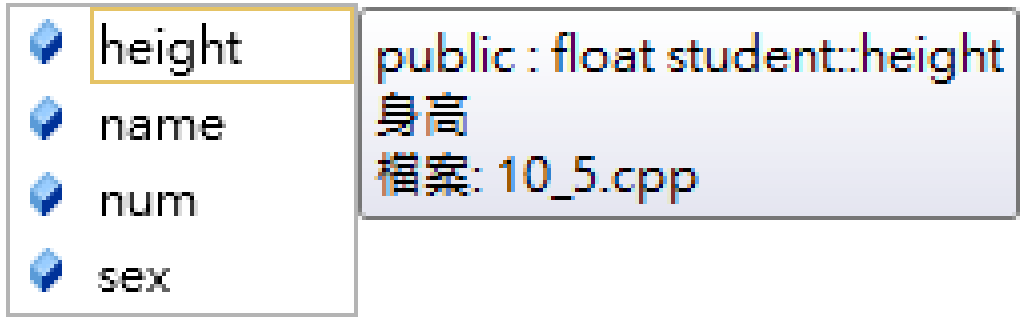
Struct

如何使用Struct裡面的變數?

- Visual Studio下針對該變數按.會自動出現該struct內的變數
- 不是每個編譯器都會有這個功能

```
    printf("%d %s %c %f\n",s1.)

    system("pause");
    return 0;
}
```



Struct

- 使用的3個步驟：

1.定義結構

2.宣告結構變數

3.使用結構變數

Struct

```c
struct student  1.定義結構
{
    int num; //號碼
    char name[10]; //姓名
    char sex; //性別
    float height; //身高
};

int main()
{
    student s1 = {1,"john",'b',190};    2.宣告結構變數
    student s2 = {2,"tom",'g',150};

    printf("%d %s %c %f\n",s1.num,s1.name,s1.sex,s1.height);  3.使用結構變數
    printf("%d %s %c %f\n",s2.num,s2.name,s2.sex,s2.height);

    system("pause");
    return 0;
}
```

```
1 john b 190.000000
2 tom g 150.000000
請按任意鍵繼續 . . .
```

Struct

```c
struct student
{
    int num; //號碼
    char name[10]; //姓名
    char sex; //性別
    float height; //身高
}s1 = {1,"john",'b',190},s2 = {2,"tom",'g',150};

int main()
{

    printf("%d %s %c %f\n",s1.num,s1.name,s1.sex,s1.height);
    printf("%d %s %c %f\n",s2.num,s2.name,s2.sex,s2.height);

    system("pause");
    return 0;
}
```

Struct

- Struct的字串如果是用字元陣列的方式宣告，宣告完無法直接更改內容，必須要透過strcpy / strncpy。

```
struct student
{
    int num; //號碼
    char name[10]; //姓名
    char sex; //性別
    float height; //身高
};
```

```
student s1 = {1,"john",'b',190};
student s2 = {2,"tom",'g',150};

s1.name = "JOHN";//error
strcpy(s1.name,"JOHN");
```

- Struct的字串如果是用字元指標的方式宣告，可以直接更改。

```
struct student
{
    int num; //號碼
    char *name; //姓名
    char sex; //性別
    float height; //身高
};
```

Struct

- 相同的Struct變數可以指定(=)
- (複習一下陣列無法直接指定，必須透過Loop)

學完struct後

Q：模擬一般正常的登入功能，每個人都有自己的帳號密碼，登入成功後根據不同的人會印出

＂ 歡迎! 使用者名字 ＂

- 至少設計3個以上不同的使用者(每個使用者有不同的名稱、帳號、密碼)

- 試著用Struct來修改剛剛的練習吧！

Union

Union(聯合)

- 一種資料結構
- Union中可以放很多個不同資料型態的資料
- 編譯器會從那些型態中，找需要最大空間的資料型態，以它的大小來配置空間。
- Union中所有成員都是存取同一塊空間。

Union

- 初始化、宣告、使用方法和struct都相同。
- Union中一次只會儲存一個成員變數的值。
- 要使用該成員時，切記要先給過值才可使用。

Union

```
union number{
    int x;
    double y;
};

int main(void){
    union number value = {10,1.5};//出現錯誤，因為union中，一次只能給
                                  //一個成員值，不能同時給兩個

    system("pause");
    return 0;
}
```

Union

```
int main(void){
    union number value ;
    value.x = 10;
    printf("value中的 x %d 有給值，所以會正常輸出\n",value.x );
    printf("value中的 y %f 沒有給值，所以輸出不正確的值\n",value.y );

    value.y = 1.5;
    printf("value中的 x %d 沒有值，所以輸出不正確的值\n",value.x );
    printf("value中的 y %f 有給值，所以會正常輸出\n",value.y );


    system("pause");
    return 0;
}
```

Union

- 為什麼要有這個?
- 早期為了節省記憶體空間
- http://pydoing.blogspot.tw/2010/06/c-union.html

Enum

Enum(列舉)

- 一種資料結構
- 宣告

```
enum months{jan ,feb,mar,apr,may,jun,jul,aug,sep,oct,nov,decm};
```

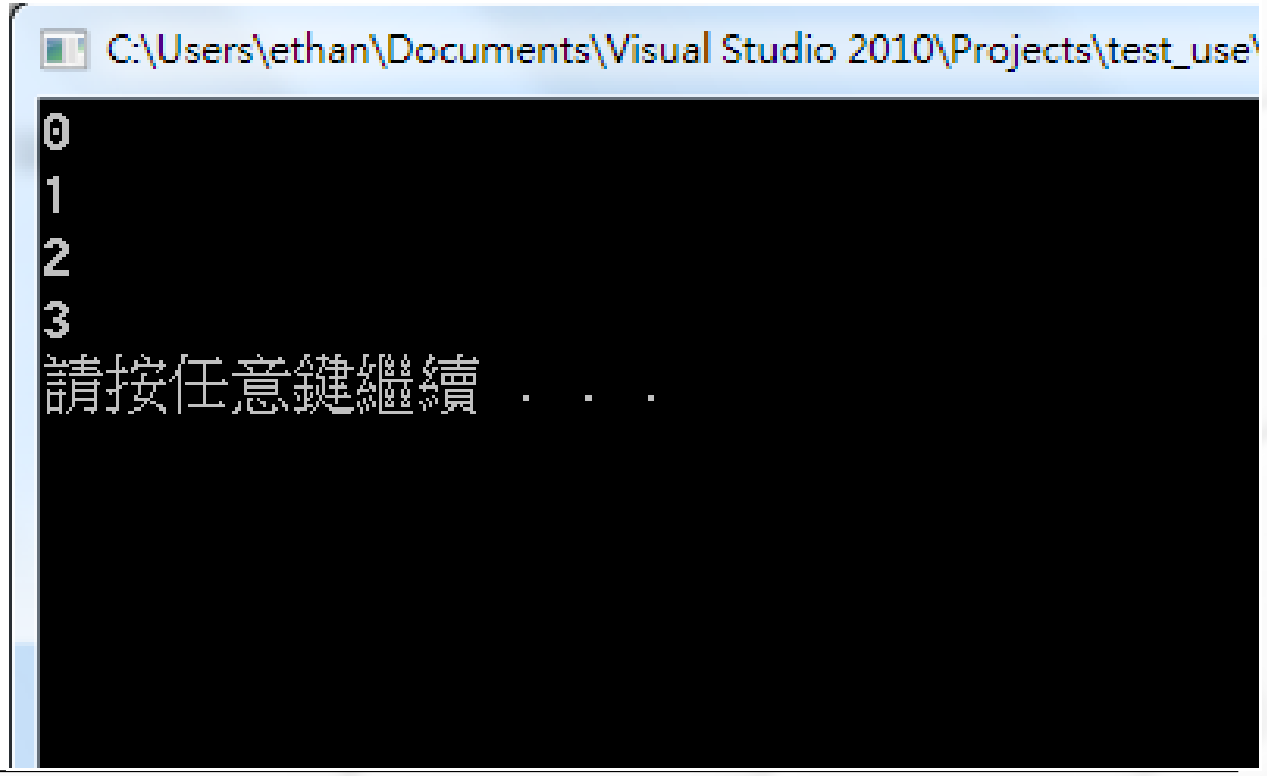
- 每一個成員對應一個整數，基本上由0開始。
- 可以自己定義成員的值，那後面的成員就會接著遞增1。

Enum

```
enum months{jan ,feb,mar,apr,may,jun,jul,aug,sep,oct,nov,decm};
int main(void){

    printf("%d\n" , jan);
    printf("%d\n" , feb);
    printf("%d\n" , mar);
    printf("%d\n" , apr);

    system("pause");
    return 0;
}
```



Enum

```c
enum months{jan = 1 ,feb,mar,apr,may,jun,jul,aug,sep,oct,nov,decm};
int main(void){

    printf("%d\n" , jan);
    printf("%d\n" , feb);
    printf("%d\n" , mar);
    printf("%d\n" , apr);

    system("pause");
    return 0;
}
```

```
1
2
3
4
請按任意鍵繼續 . . .
```

Enum

- 相當於 #define 的概念，只是使用enum來宣告，可以將相關的資料放在一起，自動幫你累加，比較方便。

```
//enum months{jan = 1 ,feb,mar,apr,may,jun,jul,aug,sep,oct,nov,decm};
#define jan 1
#define feb 2
#define mar 3
#define apr 4
int main(void){
    printf("%d\n" , jan);
    printf("%d\n" , feb);
    printf("%d\n" , mar);
    printf("%d\n" , apr);

    system("pause");
    return 0;
}
```

THANK YOU