



# C 語言

## 計數器控制的 重複結構



# 程式設計的三大結構

- 所有的程式皆由三種結構所構成
- 循序結構
- 選擇結構
  - 單一選擇敘述式：if
  - 雙重選擇敘述式：if...else
  - 多選一敘述式：swith
- 重複結構
  - while
  - do ...while
  - for

循序

選擇

重複



可以讓程式設計師在**某一種條件持續為真**時，重複執行同一項目的動作

# 回想過去的問題...

寫一個程式，輸入一個整數，如果是偶數，則印出此數是偶數，如果是奇數則印出此數是奇數。此程式可執行2次。

輸入範例：

78

77

輸出範例：

78是偶數

77是奇數

```
#做第一次
scanf("%d",&num);
if(num%2==0){
    printf("%d是偶數\n",num );
}else{
    printf("%d是奇數\n",num );
}
#做第二次
scanf("%d",&num);
if(num%2==0){
    printf("%d是偶數\n",num );
}else{
    printf("%d是奇數\n",num );
}
```

# 如果是執行次數是1000次？

- 找出重複主體

```
scanf( "%d", &num );  
if( num%2==0 ){  
    printf( "%d是偶數\n", num );  
}else{  
    printf( "%d是奇數\n", num );  
}
```

- 在某一種條件持續為真( $i \leq 1000$ )時，重複執行同一項目的動作(奇數偶數判斷)
- 變數*i*用來做次數的控制，必須設定初始值，否則執行結果可能不正確，沒有設定初始值是一個邏輯錯誤的範例

```
i=1  
while ( i<=1000 ){  
  
    重複的主體  
  
    i=i+1; # 每做一次，次數 + 1  
}
```

# 重複結構-while

```
while(條件表達式){  
    語句1;  
    語句2;  
    ...  
}
```

作為是否執行重複的**控制條件**表達式，表達式如果成立則執行迴圈。

**重複**部分由多個語句構成，應由一對括號括起來，以構成一個語句塊的形式。

# 練習

寫一個程式，顯示「HaHaHa!」10次。

輸出範例：

HaHaHa!

HaHaHa!

HaHaHa!

.....



# HaHaHa寫10次

```
int main(){
    int i=0;
    while(i<10){
        printf("HaHaHa!\n");
        i=i+1;
    }
    return 0;
}
```

控制條件要記得設定初始值

控制條件表達式，i必須小於10

重複主體由執行主體與條件改變兩個語句所構成



# 練習

寫一個程式，我說幾次，就顯示幾次「HaHaHa!」



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

```
int main(){
```

```
    int i=0;
```

```
    int n;
```

```
    scanf("%d",&n);
```

```
    while(i<n){ → 控制條件表達式，i 必須小於 n
```

```
        printf("HaHaHa!\n");
```

```
        i=i+1;
```

```
    }
```

```
    return 0;
```

```
}
```

# 練習

有10個學生的一個班級，你已經有這次測驗的成績(0到100的整數)，請算出班成績



# 參考答案解析

```
#include <stdio.h>
int main(){
    int counter=1; /*計數器，通常設定0或1，是需要而定*/
    int grade; /*成績*/
    int total=0; /*總分，如果不清0，算出來會包含先前存在total記憶體的值*/
    float average; /*平均*/
    while(counter<=10){ /*在計數器小於等於10時*/
        scanf( "%d" ,&grade); /*輸入每一筆成績*/
        total=total+grade; /*加總*/
        counter=counter+1; /*做完一次次數增加1*/
    } /*迴圈結束*/
    average=(float)total/10; /*計算平均，但是整數除以整數必須強迫轉型浮點數*/
    printf( "%.2f\n" ,average); /*顯示平均*/
    return 0;
}
```



# C 語言

## 延伸的概念



# 強制型別轉換(cast)運算子

整數除以整數時，  
如果想要得到浮點數  
的結果

```
average = (float)sum / n;  
printf("%.2f\n", average);
```

- 在C裡，兩個整數相除的小數部分將被捨棄
- 為了能夠保留計算結果的小數部分，我們必須產生一個暫時的浮點數來進行計算
- C提供單元強制型別轉換運算子，如上圖第01行，含有(float)這個強制型別轉換運算子，它會為它的運算元sum產生一個暫時的浮點數拷貝。

# 指定精準度

- %f這個轉換指定詞，沒有指定精準度，列印時會被預設6，就是%.6f
- .2指定了此值被列印時的精準度，意思是印到小數點後2位
- 但記憶體裡的值並不會被修改
- 浮點數並不一定100%的精準，例如我們讀出98.6，真正的值可能是98.5999546321666，但將值看成98.6是較具實用性的

# 程式寫作風格提醒

```
#include <stdio.h>

int main(){
    int i=0;
    int n;
    scanf("%d",&n);
    while(i<n){
        printf("HaHaHa!\n");
        i=i+1;
    }
    return 0;
}
```

寫迴圈時，迴圈內的語句  
相對應於while縮兩格





# C 語言

## 累加問題詳細拆解



# 練習

請撰寫一程式，計算所有輸入整數的總和。  
輸入的第 1 個整數，代表後面輸入整數的個數。  
接下來，逐一輸入各個整數。

輸入範例：

6  
1  
2  
3  
5  
6  
7

輸出範例：

24



# 首先：需要有一個控制次數的變數

程式實作，需要輸入一個值(以 **count** 變數表示)，一共有多少資料筆資料要輸入。

# 當輸入的次數小於count時(以6為例)，開始輸入資料



收數官  
(以1.2.3.5.6.7等數說明)

收數官的工作
1
2
3
5
6
7



控管官  
(次數到6，就不能再收了)

控管官的工作
0+1
1+1
2+1
3+1
4+1
5+1



加法官

加法官的工作
0+1=>1
1+2=>3
3+3=>6
6+5=>11
11+6=>17
17+7=>24