C/C++基礎程式設計

結構、檔案處理(Structure, File)

講師：張傑帆
CSIE, NTU

人這輩子沒辦法做太多事，所以每一件事情都要做到精采絕倫。
In your life you only get to do so many things. Right now we've chosen to do this. So let's make it great. -Steve Jobs

課程大綱

- 結構
- 檔案概論
- 作業

思考一下

- 如果你想用程式存一個人的姓名、身高、體重你會怎麼寫？

```
#include <stdio.h>
int main()
{
    char name[80];
    int  height;
    int  weight;

    scanf("%s",&name);
    scanf("%d",&height);
    scanf("%d",&weight);

    return 0;
}
```

思考一下

- 那2個人呢？

```
#include <stdio.h>
int main()
{
    char p1name[80];
    int  p1height;
    int  p1weight;

    char p2name[80];
    int  p2height;
    int  p2weight;

    scanf("%s",&p1name);
    scanf("%d",&p1height);
    scanf("%d",&p1weight);

    scanf("%s",&p2name);
    scanf("%d",&p2height);
    scanf("%d",&p2weight);

    return 0;
}
```

思考一下

- 那10個人呢？
- 因此我們需要更方便管理的變數來儲存資料：
  結構化資料

```
#include <stdio.h>
int main()
{
    char p1name[80];
    int  p1height;
    int  p1weight;

    char p2name[80];
    int  p2height;
    int  p2weight;

    ......
    ......
    ......
    ......
    ......
    ......

    return 0;
}
```

結構 (structure)

- 通常一個簡單之變數或陣列不足以用來儲存複雜之記錄。
- 我們用過了int, float, double，也可以使用陣列，若如果我們想要讓一個變數擁有許多不同的形態呢？
- C語言中有結構體之架構，允許使用者宣告資料實體將不同形式之元素儲存一起。
- 結構是一種由程式設計師自訂之資料型態。

結構的定義

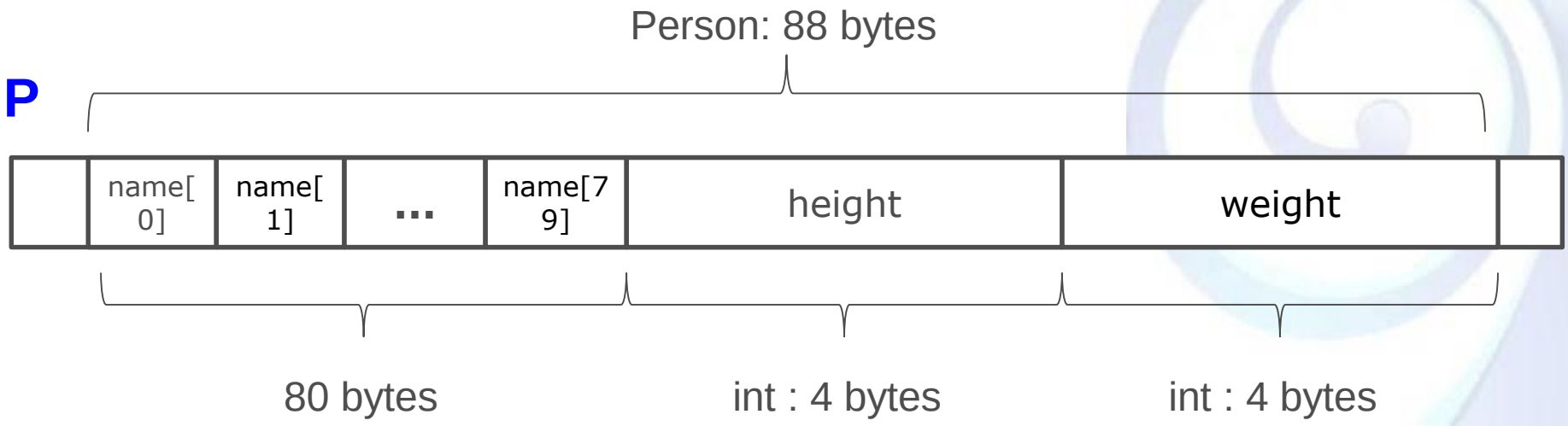
- 定義一個結構之語法
  - struct 結構名稱標籤
    {
        資料型態　資料變數成員1;
        資料型態　資料變數成員2;
        . . . . . .
    };

```
struct Person
{
        char name[80];
        int  height;
        int  weight;
};
```

- 結構的**標籤名稱(tag)**
  - 可看成是一個程式設計師自訂的新型態。
- 結構的**成員(member)**
  - 其資料型態可以使用 **int, float** 及 **char**。也可用陣列與指標或是另外一個結構

結構

- 範例:
  - struct Person
    {
        char name[80];
        int height;
        int weight;
    };
  - struct Person p; //這行在main裡面



結構的宣告與使用

- 宣告一個結構實體之語法
  - struct 結構名稱標籤 結構實體名稱;
  - struct 結構名稱標籤 結構實體名稱 = {初始值1, 初始值2, …};
- 結構中成員的使用之語法
  - 結構實體之成員的直接存取
    - 結構實體.成員名稱
  - 結構指標之成員的直接存取
    - 結構指標->成員名稱 = (*結構指標).成員名稱

結構

- 在C語言中，有提供一個比較好的方法來自己定義出一群變數的組合

```
#include <stdio.h>
struct Person
{
    char name[80];
    int  height;
    int  weight;
};
int main()
{
    struct  Person p1;
    struct  Person p2;

    scanf("%s",&p1.name);
    scanf("%d",&p1.height);
    scanf("%d",&p1.weight);
    printf("Name:%s\nHeight:%d cm\nWeight:%d kg\n",p1.name,p1.height,p1.weight);

    scanf("%s",&p2.name);
    scanf("%d",&p2.height);
    scanf("%d",&p2.weight);
    printf("Name:%s\nHeight:%d cm\nWeight:%d kg\n",p2.name,p2.height,p2.weight);
    return 0;
}
```

結構的指標

- 範例: 使用結構指標存取另一個結構的資料

```
#include <stdio.h>

struct Person
{
    char name[80];
    int  height;
    int  weight;
};

int main()
{
    struct  Person p1;
    struct  Person *p2;
    p2 = &p1;
    gets(p2->name);
    scanf("%d",&p2->height);
    scanf("%d",&p2->weight);
    return 0;
}
```

結構陣列

- 結構也可宣告成陣列，方便做重覆控制

```
#include <stdio.h>

#define PEOPLE 2
struct Person
{
    char name[80];
    int  height;
    int  weight;
};

int main()
{
    struct Person p[PEOPLE];
    int i;
    for (i=0; i < PEOPLE; i++ )
    {
        scanf("%s", &p[i].name );
        scanf("%d", &p[i].height );
        scanf("%d", &p[i].weight );
        printf("Name:%s\nHeight:%d\nWeight:%d\n",
            p[i].name, p[i].height, p[i].weight);
    }
    return 0;
}
```

typedef

- typedef可以把所有C語言中的型態改用程式設計師自己取的名字來取代

```
#include <stdio.h>

typedef unsigned char UINT8;
typedef unsigned short UINT16;
typedef unsigned long  UINT32;

int main()
{
    UINT8 a=1;
    UINT16 b=2;
    UINT32 c=3;
    return 0;
}
```

typedef

- 用“ typedef ”來自訂一個型態，常用在結構上

```
#include <stdio.h>
#define PEOPLE 2
struct _Person
{
    char name[80];
    int  height;
    int  weight;
};
typedef struct _Person Person;

int main()
{
    Person p[PEOPLE];
    int i;
    for (i=0; i < PEOPLE; i++ )    {
        scanf("%s", &p[i].name );
        scanf("%d", &p[i].height);
        scanf("%d", &p[i].weight );
        printf("Name:%s\nHeight:%d\nWeight:%d\n",
                p[i].name, p[i].height, p[i].weight);
    }
    return 0;
}
```

小練習

- 請寫一程式可以連續儲存3名公司職員的姓名、電話與職員編號，並可依職員編號查尋。

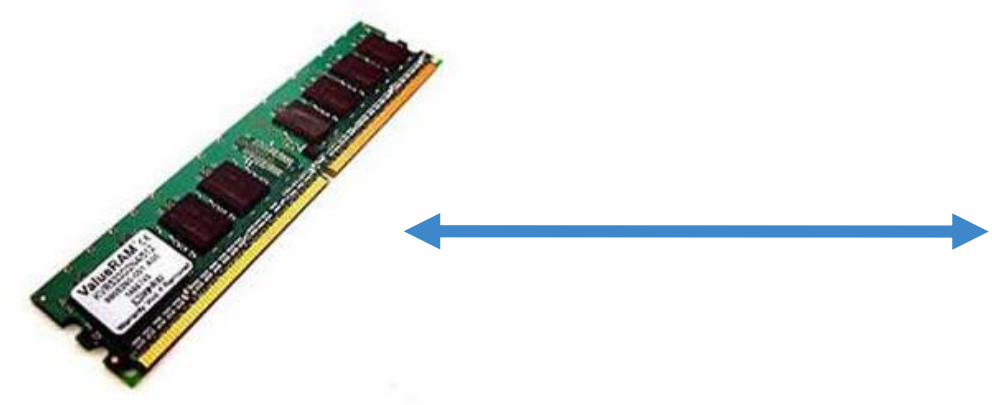
```
struct Employee
{
        char  Name[20];
        char  Phone[10];
        int  Id;
};
```

課程大綱

- 結構
- 檔案概論
- 作業

檔案處理

- 從檔案讀進資料或將結果存入檔案之中
- **fscanf**
  - 從檔案讀進資料
- **fprintf**
  - 將結果存入檔案



檔案處理

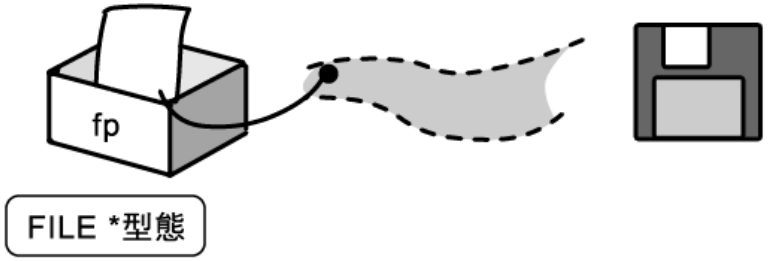
- 範例：從input.txt讀一字串到str，再把字串從str寫到output.txt

```
#include <stdio.h>
int main()
{
    FILE *in, *out;                          //in, out為檔案指標
    char str[80];

    in = fopen("input.txt", "r");        // 開啟input.txt
    fscanf(in, "%s", &str);               // 讀檔
    fclose(in);                               // 關閉input.txt

    out = fopen("output.txt", "w"); // 開啟output.txt
    fprintf(out, "%s", str);              // 寫
    fclose(out);                              //關閉output.txt
    return 0;
}
```

fopen(): 開檔



- 檔案使用前需要先開啟，開檔結果需要**檔案指標FILE**紀錄相關資訊，之後即可使用該指標讀寫檔
- 使用格式：
  - 檔案指標 = fopen(“**檔名**”, “**模式**”);
    - 檔名：檔案指標，指的是欲開啟的檔案名稱。
    - 模式：檔案使用模式，指的是檔案被開啟之後，它的使用方式。

模式	
“r”	開啟一個文字檔(text)，供程式讀取。
"w"	開啟一個文字檔(text)，供程式將資料寫入此檔案內。如果磁碟內不包含這個檔案，則系統會自行建立這個檔案。如果磁碟內包含這個檔案，則此檔案內容會被覆蓋而消失。
"a"	開啟一個文字檔(text)，供程式將資料寫入此檔案的末端。如果此檔案不存在，則系統會自行建立此檔案。
"rb"	開啟一個二元檔(binary)，供程式讀取。
"wb"	開啟一個二元檔，供程式將資料寫入此檔案內。如果磁碟內不包含這個檔案，則系統會自行建立這個檔案。如果磁碟內包含這個檔案，此檔案內容會被蓋過而消失。
"ab"	開啟一個二元檔(binary)，供程式將資料寫入此檔案末端，如果此檔案不存在，則系統會自行建立此檔案。

判斷檔案開啟是否正確

- 使用fopen若開檔失敗，會回傳一個NULL結果
- 常用來判斷檔案是否存在!

```
#include <stdio.h>
int main()
{
    FILE *in, *out;
    char str[80];
    in = fopen("input.txt", "r");
    if(in == NULL)
    {
        printf("failed to open file!\n");
        return 0;
    }
    out = fopen("output.txt", "w");
    fscanf(in, "%s", &str);
    fprintf(out, "%s", str);
    fclose(in);
    fclose(out);
    return 0;
}
```

fclose(): 關檔

- 用於關閉檔案，如果fclose( )執行失敗，它的傳回值是非零值。
- 使用格式：
  - fclose(檔案指標);
- 在C語言中關閉檔案主要有三個目的：
  - 檔案在關閉前會將檔案緩衝區資料寫入磁碟檔案內，否則檔案緩衝區資料會遺失。
    (因為磁碟IO會比較慢，所以需要緩衝區)
  - 檔案需要換個模式開啟時 (例:先寫後讀)。
  - 解除已開啟檔案的鎖定狀態。

關檔時檔案才真正寫入

- 通常，在呼叫fclose之前，檔案的內容不會真正的被寫到磁碟機上
- 若要強迫馬上寫入，可以呼叫 fflush(FILE *)來做到

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

int main()
{
    FILE *output;
    int i;
    output=fopen("test.txt", "w");
    for (i=0; i<10; i++ )
    {
        fprintf( output, "%d\n",i);
        //fflush(output);
    }
    fclose(output);
    return 0;
}
```

相對路徑與絕對路徑

- **相對路徑** 相對於現在目錄的路徑表示法，因此「相對路徑」所指到的檔案或目錄，會隨著現在目錄的不同而改變
- **假設當前目錄於E:\test\file**
  - ./　表示當前路徑，相當於E:\test\file
  - ../ 表示當前路徑的上一級路徑，相當於E:\test
  - ./data 表示當前路徑下一級路徑，相當於E:\test\file\data

相對路徑與絕對路徑

- **絕對路徑**
- 指的是一個絕對的位置，並不會隨著現在目錄的改變而改變
- EX:
  - E:\test
  - E:\test\file
  - E:\test\file\data

fprintf(): 寫檔

- 將資料，以格式化方式寫入某檔案內。
- 使用格式：
  - fprintf(檔案指標,"格式化輸出內容", 參數1,參數2, ...參數n);
    - 格式化輸出內容: 可加入列印格式、控制字元、修飾子
    - 參數: 為對應格式之變數
- 此格式化輸出內容與參數的使用，格式和printf( )使用格式相同。

fprintf():寫檔

- 範例：將陣列個數與內容寫入指定名稱之檔案

```
#include <stdio.h>
int main()
{
    FILE *file;
    int a[5] = {10, 20, 30, 40, 50};
    char str[80];
    int i, n=5;

    //開檔
    printf("input file name: ");
    scanf("%s", &str);
    file = fopen(str, "w");
    //寫檔
    fprintf(file, "%d\n", n);
    for(i=0; i<n; i++)
        fprintf(file, "%d ", a[i]);
    //關檔
    fclose(file);
    return 0;
}
```

fscanf(): 讀檔

- 從某個檔案讀取資料。
- 使用格式：
  - fscanf(檔案指標, "格式化輸入內容", &參數1, &參數2, ...&參數n);
    - 格式化輸入內容: 可加入列印格式~~、控制字元、修飾子~~
    - 參數: 對應格式之變數
- 此格式化輸入內容與參數的使用，格式和scanf( )使用格式相同。

fscanf(): 讀檔

- 範例：
  將指定名稱之檔案的陣列個數與內容讀出並印到螢幕上

```
#include <stdio.h>
int main()
{
    FILE *file;
    int a[100];
    char str[80];
    int i, n;

    //開檔
    printf("input file name: ");
    scanf("%s", &str);
    file = fopen(str, "r");
    if(file == NULL)
    {
        printf("open file fail!\n");
        return 0;
    }
    //讀檔
    fscanf(file, "%d", &n);
    for(i=0; i<n; i++)
        fscanf(file, "%d ", &a[i]);
    //輸出
    printf("%d\n", n);
    for(i=0; i<n; i++)
        printf("%d ", a[i]);
    printf("\n");
    //關檔
    fclose(file);
    return 0;
}
```

小練習

- 請將上述讀檔與寫檔的例子修改成浮點數版(小數點)

判斷檔案是否結束

- feof(檔案指標);
- 範例:寫作一個讀檔程式將檔案內容印到螢幕上

```
#include <stdio.h>
int main()
{
    FILE *input;
    char str[80];
    char ch;

    scanf("%s", &str);
    input=fopen(str,"r");
    if ( input==NULL )
    {
        printf("open input.txt fail!\n");
        return 0;
    }
    while(1)
    {
        fscanf(input, "%c", &ch);
        if( feof(input) )
            break;
        printf("%c", ch);
    }
    return 0;
}
```

課程大綱

- 結構
- 檔案概論
- 作業

作業

- 製作一個通訊錄程式
- 功能
  - 輸入’i’ 可輸入姓名, 電話, email
  - 輸入’l’ 印出目前輸入所有人員之內容
  - 輸入’s’ 輸入檔名, 將所有人員之內容存入檔案
  - 輸入’o’輸入檔名, 將所有人員之內容從檔案讀出
  - 輸入’q’離開程式
  - 輸入之人數上限為50人
- 輸入格式
  - 輸入輸出介面與下面範例相同，檔案格式不拘
- 參考範例:
  - http://homepage.ntu.edu.tw/~d02922022/C/Demo/Addressbook.exe

延申閱讀

- union 共同空間
- enum 列舉型態
- 讀寫CSV檔
- 二進位檔 fwrite()、fread()
- 隨機存取 fseek()

- 使用fscanf()函數、fprintf()函數，可以進行固定條件的資料流輸出、輸入。
- 使用fgets()函數、fputs()函數，可以進行單行的資料流輸出、輸入。
- 使用fgetc()函數、putc()函數，可以進行單一字元的資料流輸出、輸入。
- 使用fopen()函數、fclose()函數，可以進行檔案的開啟、關閉。
- 使用fread()函數、fwrite()函數，可以進行指定資料大小的輸出、輸入。
- 使用fseek()函數，可以移動檔案位置。