



第14章 檔案處理

程式設計與生活－使用C語言



Shi-Huang Chen

Spring 2013



樹德科技大學 資訊工程學系
Dept. of CSIE, Shu-Te University

第14章 檔案處理

14-1 檔案類型

14-2 檔案存取



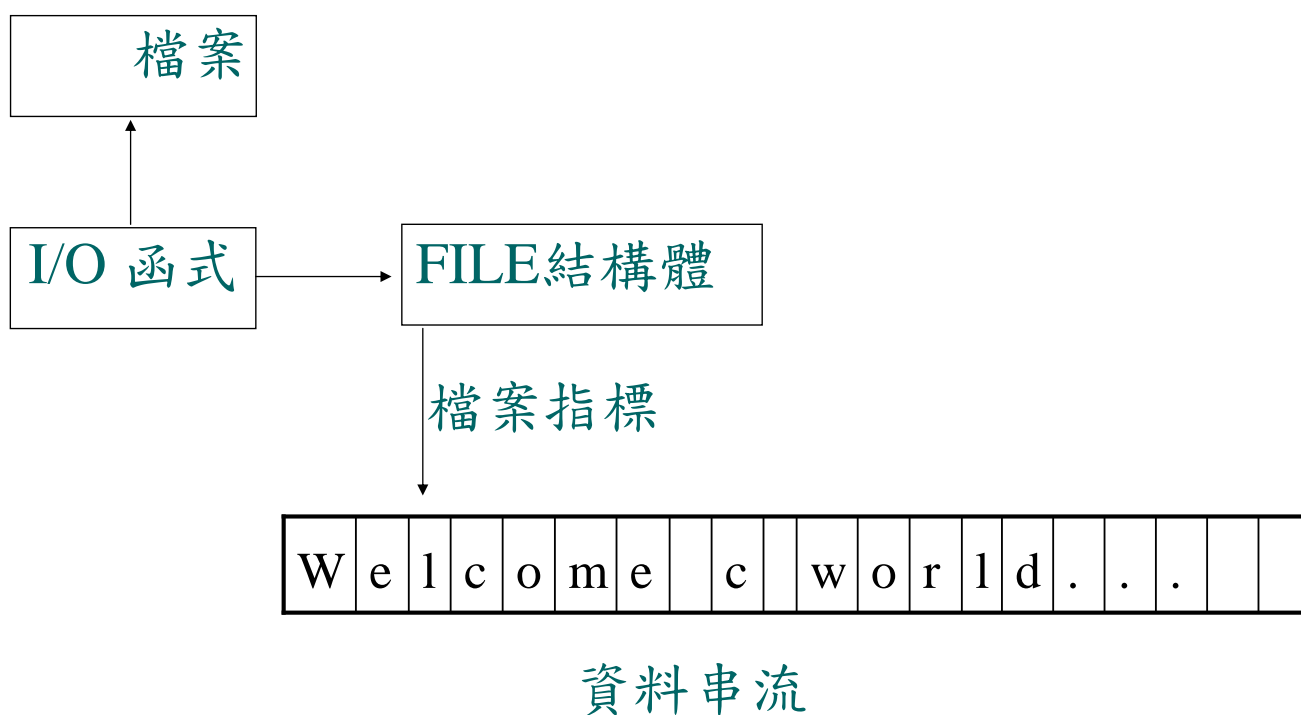
檔案

- 在第3章基本輸出函式及輸入函式中，談到資料除了可以儲存在變數之外，還可以檔案的形式儲存在硬體裝置中
 - 兩者之間的最大差異為保存時間。若資料儲存在變數，則程式結束時，其所佔用的記憶體空間會被釋放，且資料無法永久保存；
 - 若資料儲存在檔案，則不會隨程式結束而消失。



- 檔案是由眾多字元所組成的集合體，C語言將它視為一種資料串流(stream)
 - 與資料串流有關的資訊，是記錄在一個資料型態為檔案結構（FILE）的指標變數，其作用為資料串流與檔案溝通的橋樑
 - 透過這個檔案指標變數，就能對資料串流進行讀寫的動作。
 - 因此，所謂檔案處理就是資料串流處理。

○檔案處理流程



14-1 檔案類型

○檔案有下列兩種類型：

- 文字檔（Text file）：資料中的每一個字元是
以其所對應的ASCII碼來儲存

- 一般文書編輯軟體（例：windows 內建的 記事本
軟體），是以文字檔方式儲存資料



○二進制檔（Binary file）：資料中的每一個字元是直接以二進位的格式儲存

○一般執行檔、圖形檔及影像聲音檔，都是以二進制檔方式儲存

○二進制檔無法使用文書編輯軟體處理，若強用文書編輯軟體開啟，則看到的資料是一堆無法了解的亂碼



例：說明8以文字檔及二進制檔儲存的差異？

解說：

1. 因為8所對應的ASCII碼為56，所以8以文字檔儲存為00111000
2. 因為8以二進制表示為00001000，所以8以二進制檔儲存為00001000



檔案存取方式

○檔案依儲存方式分成下列兩種類型：

○循序存取(Sequential Access)：

- 資料寫入檔案時，是附加在檔案的尾端；
- 讀取資料時，是由檔案的開端由前往後一筆一筆讀出。
- 以這種方式存取資料的檔案稱為循序檔，常用於文字檔



○隨機存取 (Random Access)：

- 資料是以一筆記錄（結構資料型態）為單位寫入檔案，且每一筆記錄的長度相同
- 可利用目前資料記錄所在位置，算出實際資料的位置並取得資料。
- 以這種方式存取資料的檔案稱為隨機存取檔，常用於二進制檔



14-2 檔案存取

- 對檔案內的資料進行存取時，**首先必須將檔案開啟**，然後才能進行存取工作。
- 在存取工作完成後，**必須將檔案關閉**，避免造成檔案內的資料在電腦系統不穩的狀態下流失
- 有關檔案的I/O（輸入/輸出）處理，C語言都是藉由stdio.h函式庫中的I/O函式來達成



- 檔案處理的步驟如下：
 - 步驟1：利用fopen()函式，開啟指定的檔案。
 - 步驟2：利用資料存取函式（fscanf()函式、fprintf()函式等等）進行存取工作。
 - 步驟3：利用fclose()函式，關閉指定的已開啟檔案



14-2-1 開檔函式fopen()

函式名稱	fopen()
函式原型	FILE *fopen(const char *filename, const char *mode);
功 能	以mode模式開啟filename檔案。
傳 回	1. 若開啟檔案成功，則傳回一個FILE指標。 2. 若開啟檔案失敗，則傳回NULL。 (NULL值為0)
原型宣告 所在的標頭檔	stdio.h



〔 說明 〕

- 1.fopen()函式被呼叫時，需傳入兩個參數，第一個參數（filename）代表要開啟的檔案名稱（含路徑）；第二個參數（mode）代表檔案是以何種模式開啟
- 2.第一個參數（filename）與第二個參數（mode）的資料型態均為const char*，表示必須使用字元陣列名稱或字串常數



3. `fopen()` 函式被呼叫後，會傳回一個指向檔案開頭的檔案(FILE)指標，程式可藉由這個FILE指標，存取檔案內的資料。

- 若檔案(FILE)指標為NULL，則表示無法開啟檔案；
- 若檔案(FILE)指標不為NULL，則表示開啟檔案成功，且檔案指標會指向檔案的第一個字元



4. 若filename檔案與程式檔位於同一資料夾，則可省略路徑。

- 若filename為字串常數，則必須以雙引號將檔案名稱括起來。
- 若filename有指定檔案路徑，且路徑中一定有\字元（C中的跳脫字元），則必須在\字元前再加上一個\字元（參考範例1）



5. 檔案的開啟模式，如下表所示：

mode模式	作用(開啟成功)	開啟失敗
"r"	開啟一個唯讀的文字檔	傳回NULL
"w"	開啟一個可寫入的文字檔後， 先清除文字檔內容	會產生一個可寫入的文字檔



mode模式	作用(開啟成功)	開啟失敗
"a"	開啟一個可寫入的文字檔。 資料寫入是加在檔尾	會產生一個可寫入的文字檔
"r+"	開啟一個可讀寫的文字檔	傳回NULL
"w+"	開啟一個可讀寫的文字檔後， 先清除文字檔內容	會產生一個可讀寫入的文字檔



mode模式	作用(開啟成功)	開啟失敗
"a+"	開啟一個可讀寫的文字檔。資料要寫入是加在檔尾，作用與"a"一樣	會產生一個可讀寫入的文字檔
"rb"	開啟一個唯讀的二進制檔	傳回NULL
"wb"	開啟一個可寫入的二進制檔後，先清除二進制檔內容	會產生一個可寫入的二進制檔

mode模式	作用(開啟成功)	開啟失敗
"ab"	開啟一個可寫入的二進制檔。資料寫入是加在檔尾	會產生一個可寫入的二進制檔
"r+b"	開啟一個可讀寫的二進制檔	傳回NULL
"w+b"	開啟一個可讀寫的二進制檔後，先清除二進制檔內容	會產生一個可讀寫入的二進制檔
"a+b"	開啟一個可讀寫的二進制檔。資料要寫入是加在檔尾，作用與"ab"一樣	會產生一個可讀寫入的二進制檔



14-2-2 關檔函式fclose()

函式名稱	fclose()
函式原型	FILE *fclose(FILE * fptr);
功 能	關閉檔案指標fptr所指向的檔案
傳 回	1. 若關閉檔案成功，則傳回0 2. 若關閉檔案失敗，則傳回-1
原型宣告 所在的標頭檔	stdio.h



〔 說明 〕

1. fclose()函式被呼叫時，需傳入一個參數(fptr)，代表要關閉的資料串流之檔案指標，其資料型態為FILE*，表示必須使用檔案指標
2. 檔案處理完畢後不再使用時，一定要使fclose()函式來關閉檔案，並將緩衝區內的資料寫入檔案內，否則緩衝區資料會流失
3. 當檔案開啟模式變更時，也必須使用fclose()函式來關閉檔案後，再重新開啟檔案(例:先寫後讀)



範例1：寫一程式，輸入一文字檔名稱，然後以唯讀方式開啟該文字檔，然後再將它關閉。（假設有一test.txt檔案）

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main (void)
{
    char filename[30];
    //宣告一個指向"檔案"的檔案指標結構變數
    FILE *fptr;
    printf("輸入要開啟的文字檔名稱:");
    scanf("%s",filename);
```

```
fptr=fopen(filename,"r");//以唯讀方式,開啟檔案
if (fptr==NULL)
{
    printf ("%s檔案無法開啟!\n",filename);
    system("pause");
    exit(1); //exit( )函式作用為強迫結束程式，
    //並將( )中的參數值傳給作業系統，若參數值
    //不等於0，則表示程式執行時發生錯誤
}
printf("%s檔案已開啟!\n", filename);
printf("%s檔案已開啟!\n", filename);
//關閉fptr所指向的檔案
if (fclose(fptr)== -1)
{
    printf ("%s檔案無法關閉!\n", filename);
    system("pause");
    exit(1);
}
```



```
printf ("%s檔案已關閉!\n", filename);  
system("pause");  
return 0;  
}
```

執行
結果

輸入要開啟的文字檔名稱: **test.txt**
test.txt檔案已開啟!
test.txt檔案已關閉!



〔 程式解說 〕

程式第14列 if (fptr==NULL)

可以改成 if (fptr==0)



14-2-3 檔案I/O（輸入/輸出）函式

- 檔案成功開啟後，若要存取檔案內的資料，必要藉由C語言的stdio.h函式庫之I/O存取函式才能達成。
- I/O存取函式分成下列兩類：



- 有緩衝區的I/O存取函式(Buffered I/O)：
 - 又稱為標準I/O存取函式，是指存取檔案中的資料時，會在記憶體中建立一塊緩衝區(buffer)用來存放檔案的資料之函式。
 - 讀取緩衝區內的資料時，若緩衝區內有資料，則直接讀取緩衝區內的資料；否則將檔案的資料存入緩衝區內。
 - 而寫入資料時，資料是先寫入緩衝區內，若緩衝區空間已滿，則會將緩衝區內的資料寫入檔案中



○無緩衝區的I/O存取函式(Unbuffered I/O)：

- 又稱為系統I/O存取函式，是指存取檔案中的資料時，是直接對磁碟機內的檔案資料做讀取與寫入動作之函式




- 由於記憶體存取速度比磁碟機快，因此，利用有緩衝區的I/O存取函式存取檔案中的資料，比無緩衝區的I/O存取函式來得快，但安全性卻是無緩衝區的I/O存取函式比較好
- 本章只針對標準I/O存取函式的應用做介紹，至於系統I/O存取函式的應用，請有興趣的讀者自行參考相關的書籍

函式名稱	fgetc()
函式原型	int fgetc(FILE *fptr);
功 能	從檔案指標fptr所指向的檔案中，讀取一個字元，讀取後，檔案指標fptr會移往下一個字元所在的位址
傳 回	<ol style="list-style-type: none"> 1. 若成功讀取一個字元，則傳回字元所對應的ASCII碼 2. 若檔案指標fptr指在檔案尾端，則傳回EOF（End Of File, EOF值為-1）
原型宣告 所在的標頭檔	stdio.h



〔 說明 〕

- fgetc()函式被呼叫時，需傳入一個參數（fptr），代表所要讀取資料的資料串流之檔案指標，其資料型態為FILE*，表示必須使用檔案指標



範例2：寫一程式，開啟test.txt文字檔，
然後輸出其內容及所佔的空間(byte)。(假
設文字檔test.txt的內容如下：今年農曆大
年初一是2012/1/23星期一)



```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main (void)
{
    char ch;
    int filesize=0; //計算檔案所佔的空間(byte)
    FILE *fptr;
    fptr=fopen("test.txt","r");
    //開啟本資料夾下的test.txt檔案
    if (fptr==NULL)
```

```
{
    printf ("test.txt檔案無法開啟!\n");
    exit(1);
}
while (1)
{
    ch=fgetc(fptr);
    if (ch != EOF)
    {
        printf("%c",ch);
        filesize ++;
    }
    else
        break;
}
printf("\ntest.txt文字檔所佔的空間為");
printf("%d byte\n", filesize);
```





```
//關閉本資料夾下的test.txt檔案
if (fclose(fptr)==-1)
{
    printf("test.txt檔案無法關閉!\n");
    exit(1);
}

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

執行 結果	今年農曆大年初一是2012/1/23 星期一 test.txt文字檔所佔的空間為34 byte
----------	---



樹德科技大學 資訊工程學系
Dept. of CSIE, Shu-Te University

〔 程式解說 〕


1. 因test.txt檔案的第一列佔28bytes(含換列字元)，第二列佔6bytes(不含換列字元)，所以所佔的空間為34bytes
2. 程式第24列 if (ch != EOF)
可以改成 if (ch != -1)

函式名稱	fputc()
函式原型	int fputc(int c, FILE *fptr);
功 能	將一個字元寫入檔案指標fptr所指向的檔案中
傳 回	1. 若c字元成功寫入檔案串流，則傳回c字元所對應的ASCII碼 2. 若傳回EOF（EOF值為-1），則表示寫入過程中出現錯誤
原型宣告 所在的標頭檔	stdio.h



〔說明〕

1. fputc()函式被呼叫時，需傳入兩個參數：第一個參數（c），代表要寫入的字元（或字元所對應的ASCII碼）；第二個參數（fptr），代表要寫入的資料串流之檔案指標
2. 第一個參數（c）的資料型態為int，表示可使用整數變數、整數常數、字元變數或字元常數；第二個參數（fptr）的資料型態為FILE*，表示必須使用檔案指標




範例3：（承範例2）寫一程式，開啟test.txt文字檔，然後一個字元一個字元輸入，直到按下Enter鍵才結束輸入，並將這些字元加在test.txt文字檔內容的後面。



```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main (void)
{
    char ch;
    FILE *fptr; //以新增的方式,開啟test.txt檔案
    fptr=fopen("test.txt","a");
    if (fptr==NULL)
    {
        printf ("test.txt檔案無法開啟!\n");
        exit(1);
    }
```

```
printf("請輸入一段文字,");
printf("並以按Enter鍵作為輸入之結束:\n");
while(1)
{
    ch=getche();
    if (ch != '\r')
    {
        if (fputc(ch , fptr) == EOF)
        {
            printf("寫入時,出現錯誤:\n");
            break;
        }
    }
    else
        break;
}
```





```
//關閉本資料夾下的test.txt檔案
if (fclose(fptr)==-1)
{
    printf ("test.txt檔案無法關閉!\n");
    exit(1);
}

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

執行 結果	請輸入一段文字,並以按Enter鍵作為輸入之結束: ,有七天假
----------	---



〔 程式解說 〕

1. 程式執行後test.txt檔案內容如下：

今年農曆大年初一是2012/1/23

星期一,有七天假

2. 程式第27列 if (fputc(ch , fptr) == EOF)

可以改成

if (fputc(ch , fptr) == -1)

函式名稱	fgets()
函式原型	char *fgets(char *str , int length , FILE *fptr);
功 能	從檔案指標fptr所指向的檔案中，讀取長度為length-1之字串，並儲存在字元指標str所指向的記憶體位址
傳 回	1. 若成功讀取字串，則傳回所讀取字串，且將所讀取字串存入字元指標str所指向的記憶體位址。一次最多讀取字串byte數為length-1 2. 若檔案指標fptr指在檔案尾端，則傳回NULL（NULL值為0），且字元指標str所指向的記憶體位置維持上次的內容
原型宣告 所在的標頭檔	stdio.h



〔 說明 〕

1. fgets()函式被呼叫時，需傳入三個參數：第一個參數（str），用來儲存所讀取的字串；第二個參數（length），代表所要讀取的字串之長度-1；第三個參數（fptr），代表要讀取的資料串流之檔案指標



2. 第一個參數（str）的資料型態為char*，表示可使用字元陣列變數或字元指標變數；第二個參數（length）的資料型態為int，表示可使用整數變數或整數常數；第三個參數（fptr）的資料型態為FILE*，表示必須使用檔案指標



範例4：（承範例3）寫一程式，開啟test.txt文字檔，將其內容一次一列輸出。

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main (void)
{
    char str[81];
    FILE *fptr;
    fptr=fopen("test.txt","r");
    //開啟本資料夾下的test.txt檔案
    if (fptr==NULL)
    {
        printf ("test.txt檔案無法開啟!\n");
        exit(1);
    }
}
```



```
printf("test.txt文字檔的內容為:\n");

while(fgets(str,81,fptr) != NULL)
    printf("%s",str);

printf("\n");//關閉本資料夾下的test.txt檔案
if (fclose(fptr)==-1)
{
    printf ("test.txt檔案無法關閉!\n");
    exit(1);
}
system("pause");
return 0;
}
```



執行 結果	test.txt文字檔的內容為: 今年農曆大年初一是2012/1/23 星期一,有七天假
----------	--

〔程式解說〕

1. 程式第21列中，`fgets(str,81,fptr) != NULL`
表示一次從檔案中讀取長度最多為80(=81-1)bytes的字串，如果資料已到檔尾或遇到enter鍵，則結束讀取
第一次讀取的資料為
今年農曆大年初一是2012/1/23
第二次讀取的資料為
星期一,有七天假
2. 程式第22列 `while(fgets(str,81,fptr) != NULL)`
可以改成 `while(fgets(str,81,fptr) != 0)`



函式名稱	<code>fputs()</code>
函式原型	<code>int fputs(const char *str, FILE *fp);</code>
功 能	將一個字串寫入檔案指標fptr所指向的檔案中
傳 回	1. 若字串str成功寫入檔案串流，則傳回0 2. 若傳回EOF，則表示寫入過程中出現錯誤
原型宣告 所在的標頭檔	<code>stdio.h</code>




〔說明〕

1. `fputs()` 函式被呼叫時，需傳入二個參數：第一個參數（`str`），代表要寫入的字串；第二個參數（`fptr`），代表要寫入的資料串流之檔案指標
2. 第一個參數（`str`）的資料型態為 `const char *`，表示可使用字元陣列變數、字元指標變數或字串常數；第二個參數（`fptr`）的資料型態為 `FILE *`，表示必須使用檔案指標



函式名稱	<code>feof()</code>
函式原型	<code>int feof(FILE *fp);</code>
功 能	判斷檔案指標 <code>fptr</code> 是否指向檔案尾端
傳 回	1. 若檔案指標 <code>fptr</code> 指在檔案尾端，則傳回 16 2. 若檔案指標 <code>fptr</code> 不是指在檔案尾端，則傳回 NULL (NULL 值為 0)
原型宣告 所在的標頭檔	<code>stdio.h</code>



範例5：寫一程式，將學習程式設計的心得報告存入learn_c.txt檔案中，並將其內容從檔案中輸出（註：每列最多80bytes，要結束時，請在該列的最前面按下Ctrl+Z）


```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    FILE *fptr;
    int i;
    char string[81]; //設定一長度為80Byte的字串
    //以寫入方式,開啟本資料夾下的learn_c.txt
    fptr = fopen("learn_c.txt","w");


    //如果fptr指標指向空的,代表檔案開啟失敗
    if (fptr == NULL)

    {
        printf("開啟檔案失敗!\n");
        exit(1);
    }

    printf("輸入學習程式設計的心得報告:\n");
    printf("要結束時,請在該列的最前面按Ctrl+Z\n");
    //在一列最前面,鍵盤輸入不是Ctrl+Z(結束字元)
    //或 while(gets(string)!=0)
    while (gets(string) != NULL)
    {
        fputs(string,fptr); //將字串string,就寫入檔案裡
        fputc('\n', fptr); //將換列字元寫入檔案
    }

    if (fclose(fptr)==-1)
    {
        printf("檔案無法關閉!\n");
        exit(1);
    }
}
```





```

fptr=fopen("learn_c.txt","r");
if (fptr==NULL) {
    printf("無法開啟learn_c.txt!\n");
    system("pause");
    exit(1);
}
printf("learn_c.txt檔案內容為\n");
i=0;
while (fgets(string,80,fptr) != NULL) {
    printf("第%d列資料為:%s",i+1,string);
    i++;
}
if (fclose(fptr) == -1) {
    printf("檔案無法關閉!\n");
    exit(1);
}
system("PAUSE");
return 0;
}

```

執行結果

輸入學習程式設計的心得報告:

要結束時,請在該列的最前面按Ctrl+Z

多數的初學者,對學習程式設計的恐懼與排斥,

主要原因有下列兩點:

1.對所要處理的問題之程序不懂

2.上機練習時間不夠

(按Ctrl+Z)

learn_c.txt檔案內容為

第1列資料為:多數的初學者,對學習程式設計的恐懼與排斥,

第2列資料為:主要原因有下列兩點

第3列資料為: 1.對所要處理的問題之程序不懂

第4列資料為: 2.上機練習時間不夠



〔 程式解說 〕

程式第24列到28列，可以改成gets(string);



```
//若在一列的最前面，按下Ctrl+Z(結束字元)，  
//則表示鍵盤(stdin)資料串流的指標在檔案  
//尾端（EOF），故feof(stdin)為真，  
while (!feof(stdin))  
{  
    fputs(string , fptr); //將字串string,就寫入檔案  
    fputc('\n' , fptr); //將換列字元寫入檔案  
    gets(string);  
}
```



其中

```
while (!feof(stdin))
```

又可以改成

```
while (feof(stdin) == NULL)
```

或可以改成

```
while (feof(stdin) == 0)
```

意思都為「不在標準輸入裝置(鍵盤)尾端」。

函式名稱	fprintf()
函式原型	int fprintf(FILE *fptr, const char *format, [series]);
功 能	將資料串列series分別以指定的format格式 寫入檔案指標fptr所指向的檔案
傳 回	1. 若成功寫入資料，則傳回寫入資料的 byte數 2. 若寫入資料出現錯誤，則傳回EOF
原型宣告 所在的標頭檔	stdio.h



〔說明〕

1. `fprintf()` 函式被呼叫時，需傳入三個參數：第一個參數（`fptr`），代表要寫入的資料串流之檔案指標；第二個參數（`format`），代表資料以何種格式寫入的檔案中；第三個參數（`series`），代表要寫入的資料串列




2. 第一個參數（`fptr`）的資料型態為 `FILE *`，表示必須使用檔案指標；第二個參數（`format`），它的資料型態為 `const char *`，表示必須使用字串常數、字元陣列變數名稱或字元指標變數名稱；第三個參數（`series`）可以是無、一個或多個的變數或常數
3. 第二個參數（`format`）及第三個參數（`series`），可參考 3-1-1 節「標準廣泛輸出函式 `printf()`」

範例6：寫一程式，將下列資料寫入animal.txt中。

動物	年齡	身高
馬	2	165
狗	3	35
貓	4	25

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    FILE *fptr;
    int i;
    char animal[81]; //設定一長度為80Byte的字串
    int age,height;
    float total_age=0,total_height=0;
```

```
//以寫入的方式,開啟資料夾下的animal.txt
fptr = fopen("animal.txt","w");
if (fptr == NULL)
{
    printf("開啟檔案失敗!\n");
    exit(1);
}
fprintf(fptr,"動物\t年齡\t身高\n");
for (i=1;i<=3;i++)
{
    printf("輸入第%d種動物名稱,年齡及身高:\n", i);
    scanf("%s %d %d", animal, &age, &height);
    total_age= total_age+age;
    total_height= total_height+height;
    //將字串animal,整數,age,height分別以
    //"%s %d %d"的格式且以\t定位方式寫入
    //檔案指標 fptr 所指向的檔案animal.txt,
    //並將換列字元 ('\n') 一併寫入
```

```
fprintf(fptr,"%s\t%d\t%d\n", animal, age, height);
}
if (fclose(fptr)==-1)
{
printf ("檔案無法關閉!\n");
exit(1);
}
system("PAUSE");
return 0;
}
```


執行 結果	輸入第1種動物名稱，年齡及身高: 馬 2 165
	輸入第2種動物名稱，年齡及身高: 狗 3 35
	輸入第3種動物名稱，年齡及身高: 貓 4 25



〔 程式解說 〕

程式執行後animal.txt內容如下:

動物	年齡	身高
馬	2	165
狗	3	35
貓	4	25



函式名稱	fscanf()
函式原型	int fscanf(FILE *fptr, const char *format, series);
功 能	以指定的format格式，從檔案指標fptr所指向的檔案中讀取資料，分別存放在series變數位址串列中
傳 回	1. 若成功讀取資料，則傳回所讀取的資料項之數目 2. 若檔案指標在檔尾或讀檔發生錯誤，則傳回EOF
原型宣告 所在的標頭檔	stdio.h



〔 說明 〕

1. fscanf()函式被呼叫時，需傳入三個參數：第一個參數（fptr），代表要讀取的資料串流之檔案指標；第二個參數（format），代表資料以何種格式從檔案中讀取出來；第三個參數（series），代表讀出之資料所要存放的變數位址串列



2. 第一個參數 (fptr) 的資料型態為FILE*，表示必須使用檔案指標；第二個參數 (format)，它的資料型態為const char*，表示必須使用字串常數、字元陣列變數名稱或字元指標變數名稱；第三個參數(series)可以是一個或多個的變數位址
3. 第二個參數 (format) 及第三個參數(series)，可參考3-2-1節「標準廣泛輸入函式scanf()」



範例7：（承範例6）寫一程式，計算animal.txt檔案中動物的平均年齡及身高，並將結果寫入檔案中。

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    FILE *fptr;
    char animal[81]; //設定一長度為80Byte的字串
    int age,height;
    float total_age=0,total_height=0;
    //以寫入的方式,開啟資料夾下的animal.txt
    fptr = fopen("animal.txt","a+");
    if (fptr == NULL)
    {
        printf("開啟檔案失敗!\n");
        exit(1);
    }
}
```



```
//標題不是計算的部份，所以先將其讀取出來
fscanf(fptr,"%s %s %s",animal,animal,animal);
//若檔案指標不是指在檔尾，再讀取年齡及身高
while (fscanf(fptr,"%s",animal) != EOF)
{
    fscanf(fptr,"%d %d",&age,&height);
    total_age= total_age+age;
    total_height= total_height+height;
}
printf("平均年齡%.1f",total_age/3);
printf("平均身高%.1f\n",total_height/3);
fprintf(fptr, "平均年齡%.1f",total_age/3);
fprintf(fptr, "平均身高%.1f\n",total_height/3);
if (fclose(fptr)==-1)
```

```
{
    printf ("檔案無法關閉!\n");
    exit(1);
}
system("PAUSE");
return 0;
}
```



執行 結果	平均年齡3.0,平均身高75.0
----------	------------------

函式名稱	fseek()
函式原型	int fseek(FILE *fptr , long offset , int postion);
功 能	移動檔案指標fptr的位置。將檔案指標fptr從位置postion移動offset個bytes
傳 回	1. 若成功設定位置，則傳回0 2. 若設定位置失敗，則傳回非0的數
原型宣告 所在的標頭檔	stdio.h



〔 說明 〕

1. fseek()函式被呼叫時，需傳入三個參數：
第一個參數（fptr），代表要讀取的資料串流之檔案指標；第二個參數（offset），代表從postion處移動offset個bytes；第三個參數（postion），代表檔案指標從檔案的什麼位置開始移動



2. 第一個 (fptr) 的資料型態為FILE *，表示必須使用檔案指標；第二個參數(offset)，它的資料型態為long，表示必須使用長整數；第三個參數(postion)，它的資料型態為int，表示必須使用整數



3. postion有下列三種選項：

- SEEK_SET ：檔案開頭
- SEEK_CUR ：檔案目前位置
- SEEK_END ：檔案尾端

4. 若offset為正，則表示往後移動；否則往前移動

函式名稱	rewind()
函式原型	void rewind (FILE *fptr);
功 能	將檔案指標fptr的位置，移至檔案的開頭位置
傳 回	無
原型宣告 所在的標頭檔	stdio.h



〔 說明 〕


1. rewind()函式被呼叫時，需傳入參數（fptr），代表要讀取的資料串流之檔案指標
2. 參數（fptr）的資料型態為FILE*，表示必須使用檔案指標
3. 當檔案內容被全部讀取過一次後，檔案指標會停在檔尾。若要從頭讀取資料時，則必須先將檔案指標設定在檔案的開頭處，才能再次讀取

函式名稱	ftell()
函式原型	long ftell (FILE *fptr);
功 能	取得目前檔案指標fptr的位置
傳 回	1. 若成功取得，則傳回檔案指標fptr的位置。即，檔案指標在檔案的第幾個byte的位置(檔案位置，從0開始) 2. 若取得失敗，則會傳回 -1
原型宣告 所在的標頭檔	stdio.h

範例15：寫一程式，開啟data.txt文字檔，然後一個字元一個字元輸出，並顯示該字元在檔案內的位置（第幾個byte）。(假設data.txt的內容為：1+2=3)




```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main (void)
{
    char ch;
    FILE *fptr;
    fptr=fopen("data.txt","r");
    if (fptr==NULL)
    {
        printf ("data.txt檔案無法開啟!\n");
        exit(1);
    }
}
```

```
while (1)
{
    ch=fgetc(fptr);
    if (ch != EOF)
    {
        //取得目前檔案指標fptr的位置
        printf("檔案的第%dbyte ",ftell(fptr));
        printf("的字元為%c\n",ch);
    }
    else
        break;
}

//關閉資料夾下的data.txt檔案
if (fclose(fptr)==-1)
{
    printf ("data.txt檔案無法關閉!\n");
    exit(1);
}
```



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



執行 結果	檔案的第0byte的字元為1 檔案的第1byte的字元為+ 檔案的第2byte的字元為2 檔案的第3byte的字元為= 檔案的第4byte的字元為3
----------	--