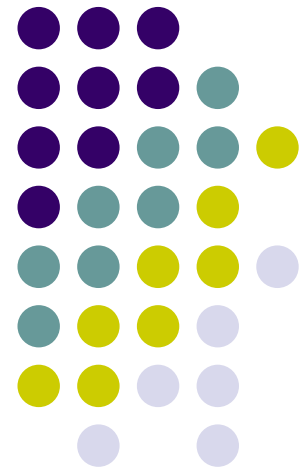


第十二章 檔案處理

檔案的觀念與操作的方式
有緩衝區與無緩衝區的檔案處理函數
二進位檔案的使用方式



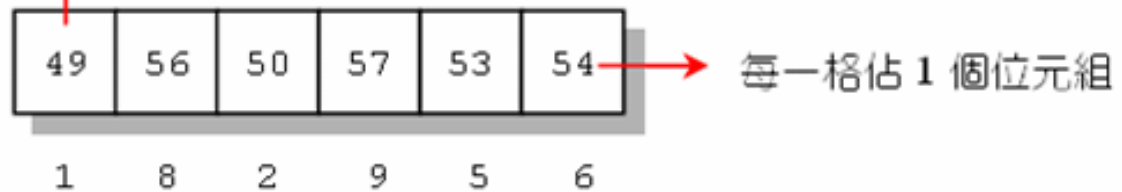


檔案的分類 (1/2)

- 檔案依儲存方式，可分為文字檔與二進位檔

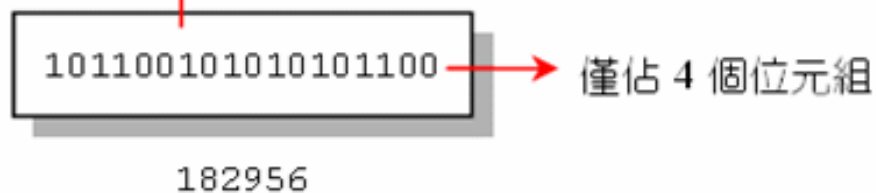
將數值 182956
以文字檔儲存

以 ASCII 碼儲存



將數值 182956 以二
進位的格式儲存

以二進位碼儲存

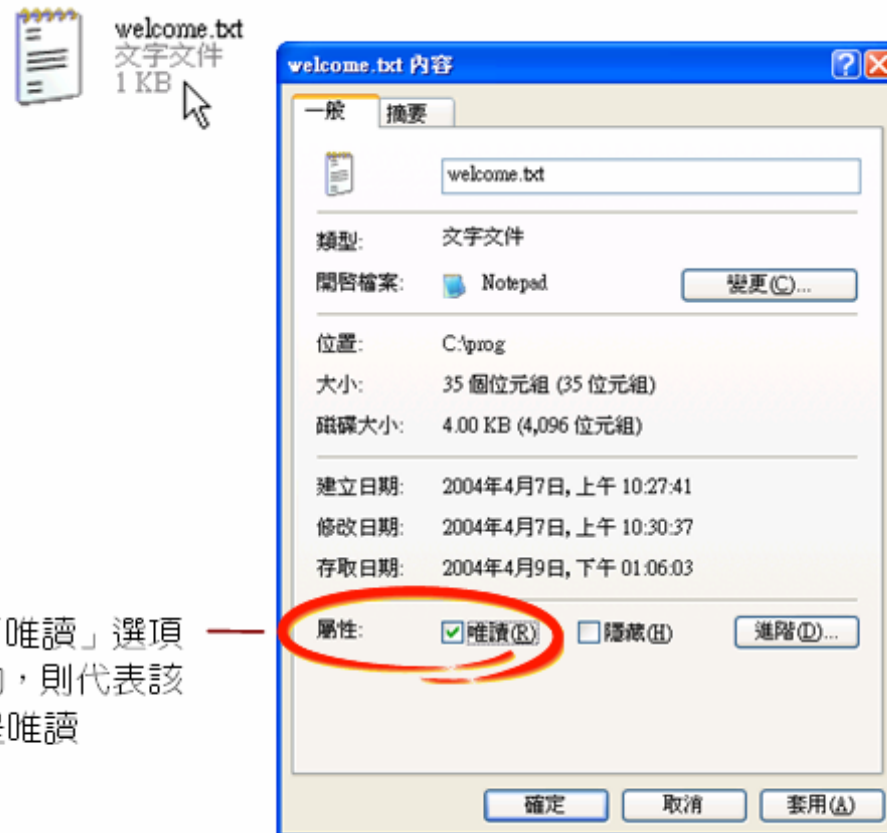


- 文字檔以列為單位，列與列之間為 '\n' 字元；二進位檔裡面沒有額外的 '\n' 字元



檔案的分類 (2/2)

- 檔案依存取權限，可分為一般檔案與唯讀檔 (read-only)
- 查看檔案屬性是否為唯讀：



如果「唯讀」選項被打勾，則代表該檔案是唯讀



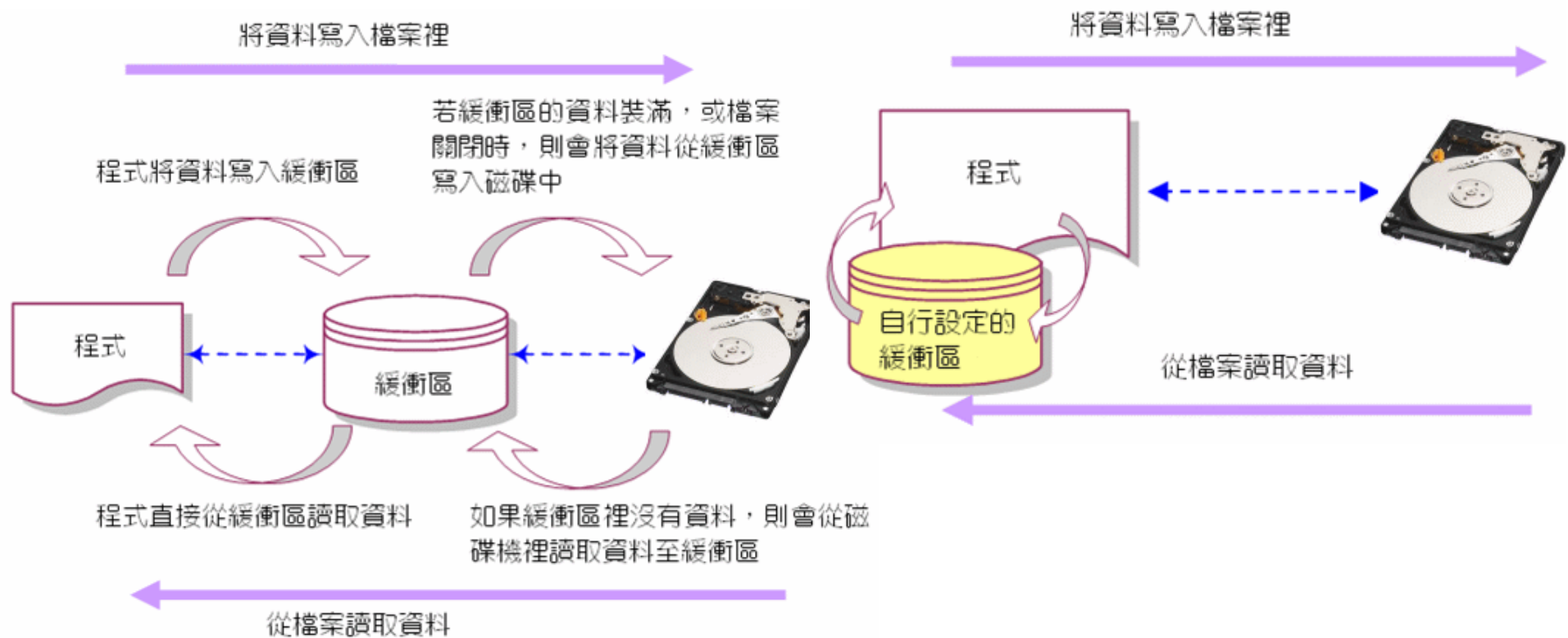
有緩衝區與無緩衝區的檔案處理

- 有緩衝區的檔案處理：

stdio

- 無緩衝區的檔案處理：

conio, system calls





檔案處理函數 fopen()

- 有緩衝區的檔案處理函數

FILE *指標變數; /* 宣告指向檔案結構的指標 */
指標變數 = fopen("欲開啓檔案名稱", "存取模式");

存取模式	代碼	說明
讀取資料	r	開啟檔案以供讀取。在開啟前，此檔案必須先存在於磁碟機內。如果檔案不存在，則開檔函數 fopen() 開檔失敗，將無法執行
寫入資料	w	開啟檔案以供寫入。如果檔案已經存在，則該檔案的內容將被覆蓋掉。如果檔案不存在，則系統會自行建立此檔案
附加於檔案之後	a	開啟一個檔案，可將資料寫入此檔案的末端。如果檔案不存在，則系統會自行建立此檔案



開啓檔案的範例

- 開啓檔案 abc.txt 以讀取資料：

```
FILE *fptr; /* 宣告指向檔案的指標fptr */
fptr=fopen("abc.txt","r"); /* 開啓檔案abc.txt以供讀取 */
if(fptr!=NULL) /* 判別檔案是否開啓成功 */
{
    /* 檔案開啓成功時，所要執行的程式碼 */
}
else
{
    /* 檔案開啓失敗時，所要執行的程式碼 */
}
```



常用的檔案處理函數 (1/2)

- `stdio.h` 標頭檔中所宣告的檔案處理函數：

函數功能	格式及說明
開啟檔案	<code>FILE *fopen(const char *filename, const char *mode);</code> 開啟指定的檔案，並指定存取模式。 <code>fopen()</code> 的第一個引數為檔案名稱字串，第二個引數為存取模式的代表。 <code>fopen()</code> 的傳回值為檔案指標，開檔失敗傳回 <code>NULL</code>
關閉檔案	<code>int fclose(FILE *fptr);</code> 關閉由 <code>fptr</code> 所指向的檔案，關檔成功傳回 <code>0</code>
讀取字元	<code>int getc(FILE *fptr);</code> 由 <code>fptr</code> 所指向的檔案讀取一個字元，傳回值為被讀取的字元
寫入字元	<code>int putc(int ch, FILE *fptr);</code> 將字元 <code>ch</code> 寫入由 <code>fptr</code> 所指向的檔案



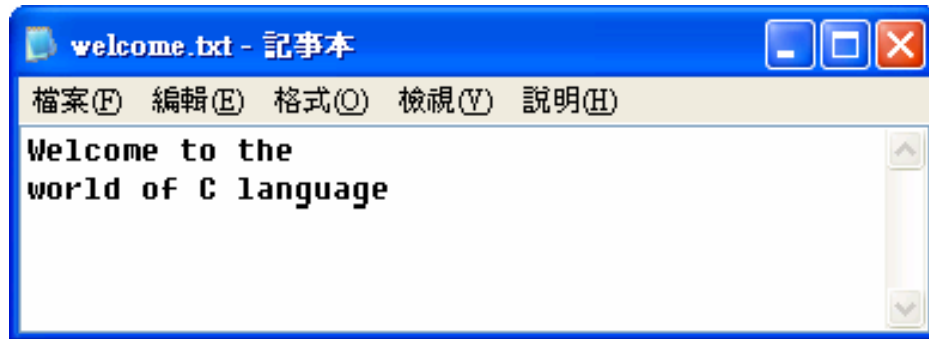
常用的檔案處理函數 (2/2)

函數功能	格式及說明
讀取字串	<pre>char *fgets(char *str,int maxchar,FILE *fptr);</pre> 從 fptr 所指向的檔案裡讀取最多 maxchar 個字元，然後將它寫入字元陣列 str 中。若讀取失敗，或已讀到檔尾，則傳回 NULL
寫入字串	<pre>int fputs(const char *str,FILE *fptr);</pre> 將字串 str 寫入 fptr 所指向的檔案
檢查檔案是否結束	<pre>int feof(FILE *fptr);</pre> 檢查 fptr 所指向的檔案是否已讀取到檔案結束的位置。若尚未到達檔尾，則傳回 0 ；若已到檔尾，則傳回非 0 的值
區塊輸入	<pre>size_t fread(void *p,size_t s,size_t cnt,FILE *fptr);</pre> 由檔案讀取 cnt 個資料項目，存放到指標 p 所指向的位址中，每一個資料項目的大小為 s 個位元組，傳回值為讀取資料的個數
區塊輸出	<pre>size_t fwrite(const void *p,size_t s,size_t cnt,FILE *fptr);</pre> 將 cnt 個大小為 s 個位元組的資料，寫入指標 p 所指向的位址中，傳回值為成功寫入資料的個數



檔案處理函數的練習 (1/2)

- 利用有緩衝區的檔案處理函數，讀取文字檔 welcome.txt



```
/* prog12_1 OUTPUT---  
Welcome to the  
world of C language  
總共有 34 個字元  
-----*/
```

```
01  /* prog12_1, 顯示檔案內容，並計算字元數 */  
02  #include <stdio.h>  
03  #include <stdlib.h>  
04  int main(void)  
05  {  
06      FILE *fptr;          /* 宣告指向檔案的指標 fptr */  
07      char ch;             /* 宣告字元變數 ch，用來接收讀取的字元 */  
08      int count=0;        /* 宣告整數 count，用來計算檔案的字元數*/  
09
```



檔案處理函數的練習 (2/2)

```

10  fptr=fopen("c:\\prog\\welcome.txt","r");      /* 開啟檔案 */
11  if(fptr!=NULL) /* 如果 fopen() 的傳回值不為 NULL, 代表檔案開啟成功 */
12  {
13      while( (ch=getc(fptr)) !=EOF) /* 判斷是否到達檔尾 */
14      {
15          printf("%c",ch); /* 一次印出一個字元 */
16          count++;
17      }
18      fclose(fptr); /* 關閉所開啟的檔案 */
19      printf("\n 總共有%d 個字元\n",count);
20  }
21  else /* 檔案開啟失敗 */
22      printf("檔案開啟失敗!!\n");
23
24  system("pause");
25  return 0;
26  }

```

空白與換行字元也列入字元數的計算，因此總字元數為34

W	e	l	c	o	m	e		t	o		t	h	e	\n
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

w	o	r	l	d		o	f		C		l	a	n	g	u	a	g	e
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34



檔案拷貝的範例

```

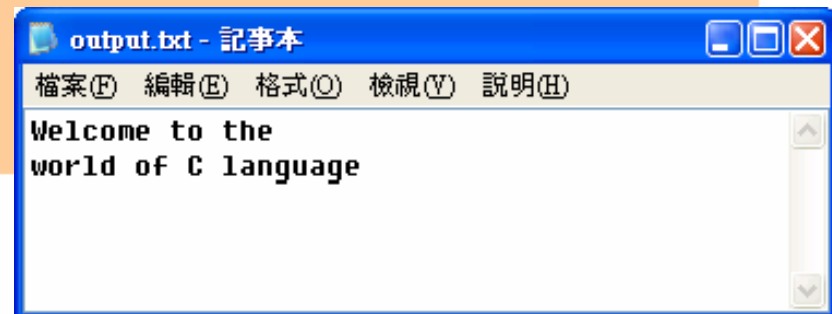
01  /* prog12_2, 拷貝檔案內容到其它的檔案 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06      FILE *fptr1, *fptr2;      /* 宣告指向檔案的指標 fptr1 與 fptr2 */
07      char ch;
08      fptr1=fopen("c:\\prog\\welcome.txt", "r"); /* 開啟可讀取的檔案 */
09      fptr2=fopen("c:\\prog\\output.txt", "w"); /* 開啟可寫入的檔案 */
10      if((fptr1!=NULL) && (fptr2!=NULL))      /* 如果開檔成功 */
11      {
12          while((ch=getc(fptr1))!=EOF)        /* 判斷是否到達檔尾 */
13              putc(ch, fptr2);                /* 將字元 ch 寫到 fptr2 所指向的檔案 */
14          fclose(fptr1);                       /* 關閉 fptr1 所指向的檔案 */
15          fclose(fptr2);                       /* 關閉 fptr2 所指向的檔案 */
16          printf("檔案拷貝完成!!\n");
17      }
18      else
19          printf("檔案開啟失敗!!\n");
20      system("pause");
21      return 0;
22  }

```

```

/* prog12_2 OUTPUT--
檔案拷貝完成!!
-----*/

```



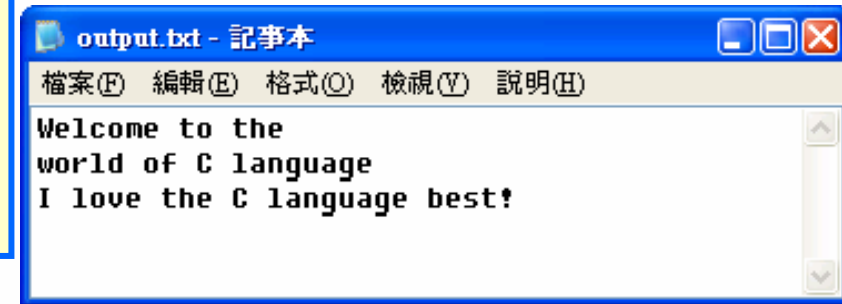


將字串附加到檔案的範例

```

0  文字檔案一般不使用 fwrite() 而是先  0  加入字串結束字元 str[i] = 0;
0  然後使用
0  fputs(str, fptr);
0  或是
0  fprintf(fptr, "%s", str);
08  {
09     FILE *fptr;
10     char str[MAX],ch; /* 宣告字元陣列 str，用來儲存由鍵盤輸入的字串 */
11     int i=0;
12     fptr=fopen("c:\\prog\\output.txt","a");
13     printf("請輸入字串，按 ENTER 鍵結束輸入:\n");
14     while((ch=getche())!=ENTER && i<MAX) /* 按下的鍵不是 ENTER 且 i<MAX */
15         str[i++]=ch; /* 一次增加一個字元到字元陣列 str 中 */
16     putc('\n',fptr); /* 寫入換行字元 */
17     fwrite(str,sizeof(char),i,fptr);
18     fclose(fptr); /* prog12_3 OUTPUT-----
19     printf("\n 檔案附加完成!!\n"); 請輸入字串，按 ENTER 鍵結束輸入:
20     system("pause"); I love the C language best!
21     return 0; 檔案附加完成!!
22 } -----*/

```





利用 fread() 函數讀取檔案內容

文字檔案一般不使用 fread() 而是使用

fgets(str, MAX-1, fptr); /* 一次讀取一行 */

或是

fscanf(fptr, "%s", str); /* 以 white space 為分隔單位讀入 */

```

01
02
03
04
05  int main(void)                                world of C language
06  {                                              I love the C language best!
07      FILE *fptr;                                -----*/
08      char str[MAX];
09      int bytes;                                  /* 存放 fread() 成功讀取的字元數 */
10      fptr=fopen("c:\\prog\\output.txt","r");
11      while(!feof(fptr))                          /* 如果還沒讀到檔尾 */
12      {
13          bytes=fread(str,sizeof(char),MAX,fptr);
14          if(bytes<MAX)
15              str[bytes]='\0';
16          printf("%s\n",str);                      /* 印出檔案內容 */
17      }
18      fclose(fptr); /* 關閉檔案 */
19      system("pause");
20      return 0;
21  }

```



無緩衝區的檔案處理函數

- 利用作業系統的 `open()` 函數可開啓無緩衝區的檔案

`open("檔案名稱", 開啓模式, 存取屬性);` 注意: 不是 C 的標準函數庫

檔案開啟模式		說 明
基本模式	<code>O_RDONLY</code>	開啟的檔案只供讀取，不能寫入資料
	<code>O_WRONLY</code>	開啟的檔案只供寫入，不能讀取資料
	<code>O_RDWR</code>	開啟的檔案可供讀取與寫入資料
修飾模式	<code>O_CREAT</code>	若開啟的檔案不存在，則建立新檔；若存在，則此功能無效
	<code>O_APPEND</code>	開啟的檔案可供寫入，寫入時不會蓋掉原有的內容，而是附加在其後，若與 <code>O_RDONLY</code> 一起使用，則此功能無效
	<code>O_BINARY</code>	開啟一個二進位檔案 (binary file)
	<code>O_TEXT</code>	開啟文字檔案



檔案開啓模式與存取屬性

- 有多種開啓模式時，可以利用OR「|」運算子串接模式

O_WRONLY /* 開啓舊檔，此檔只供寫入，不能讀取 */

O_WRONLY | O_APPEND /* 開啓舊檔，此檔可以附加資料，但不能讀取 */

O_WRONLY | O_CREAT | O_APPEND /* 若檔案不存在，則建立可附加資料的新檔 */

O_RDONLY | O_TEXT /* 開啓已存在的文字檔，且只供讀取 */

open(“檔案名稱”, **O_CREAT**, 存取屬性);

表 12.3.2 存取權限的模式

存取屬性	說明
S_IWRITE	新建立的檔案可供寫入
S_IREAD	新建立的檔案只供讀取（即屬性為唯讀）
S_IREAD S_IWRITE	新建立的檔案，可供讀取與寫入資料



無緩衝區的檔案處理函數

表 12.3.3 無緩衝區的檔案處理函數

函數功能	格式
開啟檔案	<code>int open(const char *filename, int oflag[, int pmode]);</code> 開啟指定的檔案及開啟模式，傳回值為檔案代號，開檔失敗時傳回-1
關閉檔案	<code>int close(int handle);</code> 關閉指定的檔案，關檔成功傳回 0，關檔失敗傳回 1
開新檔案	<code>int creat(const char *filename, int pmode);</code> 建立一個存取屬性為 pmode 的檔案，傳回值為檔案代號，開檔失敗時傳回-1
讀取資料	<code>int read(int handle, char *buffer, unsigned count);</code> 讀取資料，最多一次讀取 count 位元組，並存放到位址為 buffer 的變數裡。傳回值為實際讀取資料的位元組，若是傳回-1，表示讀取失敗
寫入資料	<code>int write(int handle, char *buffer, unsigned count);</code> 將位址為 buffer 的變數內容寫入檔案中，最多可一次寫入 count 位元組，傳回值為實際寫入資料的位元組，若是傳回-1，表示寫入失敗



檔案處理函數的練習 (1/2)

- 下面的範例是以無緩衝區的檔案函數來複製檔案內容：

```
01  /* prog12_5, 複製檔案內容 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  #include <fcntl.h>
05  #include <io.h>
06  #include <sys/stat.h>
07  #define SIZE 512          /* 設定 read() 一次可讀取的最大位元組為 512 */
08  int main(void)
09  {
10      char buffer[SIZE];
11      int f1,f2,bytes;
12
13      f1=open("c:\\prog\\welcome.txt",O_RDONLY|O_TEXT);
14      f2=creat("c:\\prog\\output2.txt",S_IWRITE);
15
```

注意:和作業系統相關,不是 C 的標準函數庫



檔案處理函數的練習 (2/2)

```
16     if((f1!=-1)&&(f2!=-1))      /* 測試檔案是否開啟成功 */
17     {
18         while(!eof(f1))        /* 如果還沒有讀到檔案末端 */
19         {
20             bytes=read(f1,buffer,SIZE);    /* 從 f1 讀取資料 */
21             write(f2,buffer,bytes);        /* 將資料寫入檔案 f1 中 */
22         }
23         close(f1);
24         close(f2);
25         printf("檔案拷貝完成!!\n");
26     }
27     else
28         printf("檔案開啟失敗!!\n");
29
30     system("pause");
31     return 0;
32 }
```

```
/* prog12_5 OUTPUT---
檔案拷貝完成!!
-----*/
```

```
output2.txt - 記事本
檔案(F) 編輯(E) 格式(O) 檢視(V) 說明(H)
Welcome to the
world of C language
```



使用 `fopen()` 處理二進位檔

- 使用 `fopen()` 函數開啓二進位檔案時的存取模式：

表 12.4.1 二進位檔案的存取模式

存取模式	代碼	說明
二進位檔的讀取	rb	開啟一個僅供讀取資料的二進位檔案 (binary file)
二進位檔的寫入	wb	開啟一個僅供寫入資料的二進位檔案
二進位檔的附加	ab	開啟一個可以附加資料的二進位檔案

```
/* 開啓可供附加資料的二進位檔案test.bin */  
FILE *fptr;  
fptr=fopen("test.bin", "ab");
```



輸出資料到二進位檔案的範例

```
01  /* prog12_6, 輸入資料到二進位檔案 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06      double a=3.14,b=6.28;
07      int arr[]={12,43,64};
08      FILE *fptr;
09
10      fptr=fopen("c:\\prog\\number.bin","wb");      /* 開啟檔案 */
11      fwrite(&a,sizeof(double),1,fptr);          /* 寫入變數 a 的值 */
12      fwrite(&b,sizeof(double),1,fptr);          /* 寫入變數 b 的值 */
13      fwrite(arr,sizeof(int),3,fptr);            /* 寫入陣列 arr 的所有元素 */
14      fclose(fptr);                               /* 關閉
15      printf("檔案寫入完成!!\n");
16      system("pause");
17      return 0;
18 }
```

```
/* prog12_6 OUTPUT---
檔案寫入完成!!
-----*/
```

請注意: 作業系統不區分檔案是文字或是二進位, 是應用程式自己以文字模式或是二進位模式去存取檔案的內容, 二進位檔案通常以 **fread** 或是 **fwrite** 順序讀/寫



讀取二進位檔案的範例

```
01 /* prog12_7, 讀取二進位檔案的內容 */
02 #include <stdio.h>
03 #include <stdlib.h>
04 int main(void)
05 {
06     double a,b;
07     int i,arr[3];
08     FILE *fptr;
09     fptr=fopen("c:\\prog\\number.bin","rb"); /* 開啟檔案 */
10     fread(&a,sizeof(double),1,fptr); /* 把讀取的資料設定給 a 存放 */
11     fread(&b,sizeof(double),1,fptr); /* 把讀取的資料設定給 b 存放 */
12     fread(arr,sizeof(int),3,fptr); /* 把讀取的資料設定給陣列 arr 存放 */
13
14     printf("a=%4.2f\n",a);
15     printf("b=%4.2f\n",b);
16     for(i=0;i<3;i++)
17         printf("arr[%d]=%d\n",i,arr[i]);
18     fclose(fptr); /* 關閉檔案 */
19     system("pause");
20     return 0;
21 }
```

```
/* prog12_7 OUTPUT---
a=3.14
b=6.28
arr[0]=12
arr[1]=43
arr[2]=64
-----*/
```



利用 write() 寫入二進位檔 (1/2)

- 下面的程式是把結構變數的內容寫入二進位檔中：

```
01  /* prog12_8, 輸入資料到二進位檔案 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  #include <fcntl.h>
05  #include <io.h>
06  #include <sys/stat.h>
07  int main(void)
08  {
09      int fl;
10      struct data          /* 定義結構 data */
11      {
12          char name[10];
13          int math;
14      }student={"Jenny",96}; /* 宣告結構變數 data, 並設定初值 */
15
```

注意: 和作業系統相關, 不是 C 的標準函數庫



利用 write() 寫入二進位檔 (2/2)

```
16     f1=open("c:\\prog\\score.bin",O_CREAT|O_WRONLY|O_BINARY,S_IREAD);
17     if((f1!=-1))          /* 檔案開啟成功 */
18     {
19         write(f1,&student,sizeof(student));
20         close(f1);
21         printf("資料已寫入檔案!!\n");
22     }
23     else
24         printf("檔案開啟失敗!!\n");
25
26     system("pause");
27     return 0;
28 }
```

```
/* prog12_8 OUTPUT--
```

```
資料已寫入檔案!!
```

```
-----*/
```



利用 read() 讀取二進位檔 (1/2)

- 下面的程式利用read() 函數讀取二進位檔案：

```
01  /* prog12_9, 讀取二進位檔案的內容 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  #include <fcntl.h>
05  #include <io.h>
06  #include <sys/stat.h>
07  int main(void)
08  {
09      int fl;
10      struct data
11      {
12          char name[10];
13          int math;
14      }student;          /* 宣告結構變數 student */
15      fl=open("c:\\prog\\score.bin",O_RDONLY | O_BINARY);
16
```

```
/* prog12_9 OUTPUT---
student.name=Jenny
student.math=96
-----*/
```

注意: 和作業系統相關, 不是 C 的標準函數庫



利用 `read()` 讀取二進位檔 (2/2)

```
17     if ((f1 != -1))          /* 檔案開啟成功 */
18     {
19         read(f1, &student, sizeof(student)); /* 讀取資料並給 student 存放 */
20         printf("student.name=%s\n", student.name);
21         printf("student.math=%d\n", student.math);
22         close(f1);
23     }
24     else /* 檔案開啟失敗 */
25         printf("檔案開啟失敗!!\n");
26
27     system("pause");
28     return 0;
29 }
```

```
/* prog12_9 OUTPUT--
```

```
student.name=Jenny
```

```
student.math=96
```

```
-----*/
```