

Example

26

檔案處理：隨機存取

商品基本資料包含產品名稱與價錢，寫一程式提供以下功能：新增資料、顯示資料與查詢資料。商品基本資料以結構表示，並以二進位檔案儲存商品資料。查詢資料時輸入欲查詢第幾筆資料後顯示查詢到的商品資料。

一、學習目標

試想以下種情形：檔案裏面有 1000 筆資料，現在要查詢第 700 筆的資料。依照目前所學的方式，要先從檔案的開頭讀取 699 筆資料之後，第 700 筆才是要查詢的資料。另一種情形：現在要修改第 800 筆的資料，則要先從檔案讀取所有的 1000 筆資料並儲存到陣列變數裡面，接著修改第 800 筆的資料，最後再將這 1000 筆資料重新寫入檔案。這樣的處理方式並沒有效率；若檔案裡有數萬筆資料的時候，這樣的處理方式不僅非常耗費時間，也容易在處理的過程中產生問題。

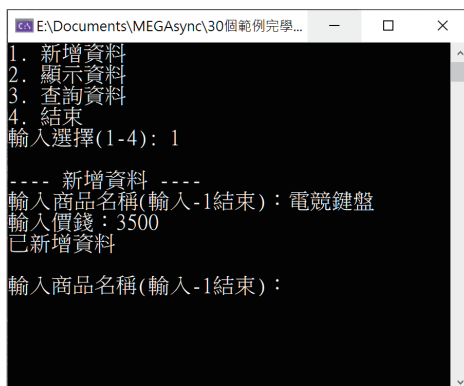
使用隨機存取的方式可以處理上述的問題：建立或是開啓檔案之後，指定檔案裡要讀寫資料的索引位置，便可以讀寫此位置的資料。使用隨機存取的方式，通常搭配二進位檔案，並且所儲存的每一筆資料都是固定的長度；如此才能計算檔案裡要讀寫資料的正確索引位置。

二、執行結果

下圖左為選單畫面，一共有 4 個選項：「新增資料」、「顯示資料」、「查詢資料」與「結束」。下圖右為「新增資料」的畫面：輸入商品名稱與價錢。若輸入的商品名稱為 -1，則結束輸入資料。每次輸入一筆商品資料之後，便會立即儲存到檔案中。

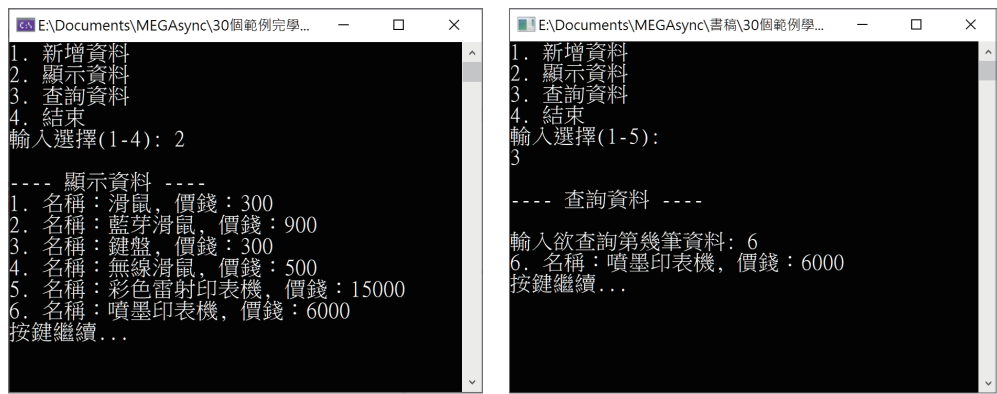


```
E:\Documents\MEGAsync\30個範例完學...
1. 新增資料
2. 顯示資料
3. 查詢資料
4. 結束
輸入選擇(1-4):
```



```
E:\Documents\MEGAsync\30個範例完學...
1. 新增資料
2. 顯示資料
3. 查詢資料
4. 結束
輸入選擇(1-4): 1
---- 新增資料 ----
輸入商品名稱(輸入-1結束): 電競鍵盤
輸入價錢: 3500
已新增資料
輸入商品名稱(輸入-1結束):
```

下圖左為「顯示資料」的畫面：開啓檔案並從檔案裡逐筆讀取資料顯示；畫面中有多筆不同的滑鼠商品資料。下圖右為查詢資料的畫面：輸入查詢第 6 筆資料後，顯示搜尋到的商品名稱與價錢。



26-1 隨機讀取資料

隨機讀取資料的意思是：能從檔案中讀取指定位置的資料。要能做到如此的功能，檔案必須符合 2 個條件：

- 1. 資料儲存為二進位檔案。
- 2. 每筆儲存的資料長度相同。

因為每筆資料的長度固定，所以某一筆資料在檔案中的位置才能被計算出來。

檔案索引位置

例如：二進位檔案中儲存 `int` 型別的資料：50、28、32、11、90、45、20；這些資料在檔案中的位置如下所示：

0	4	8	12	16	20	24
50	28	32	11	90	45	20

檔案的索引位置從 0 開始，因為 `int` 型別資料的長度為 4 個位元組，因此可以輕易計算出某筆資料在檔案中的位置。例如要尋找 32 這個數值：32 是第 3 個數值，所以在檔案中的位置為：

$$(3 - 1) \times 4 = 8$$

因此，可以將上述的計算方式予以通式化，如下所示；其中，`no` 為要讀取的第幾筆資料，`sizeof()` 函式裡的引數可以為資料型別或是變數。

$$(no - 1) \times sizeof(\text{變數或資料型別})$$

例如：要讀第 5 筆資料，則從檔案裡讀取資料的位置為：

$$(5 - 1) \times sizeof(int)$$

讀取與設定檔案索引位置

`fstream` 類別提供了 2 組用於讀取與設定檔案索引位置的函式：`seekg()`、`tellg()`、`seekp()`、`tellp()`。用於讀取資料的檔案使用 `seekg()` 與 `tellg()`；即 `ifstream` 類別所開啓的檔案，或是使用 `ios::in` 開檔模式的檔案。用於寫入資料的檔案則使用 `seekp()` 與 `tellp()`；即 `ofstream` 類別所開啓的檔案，或是使用 `ios::out` 開檔模式的檔案。

函式	說明
<code>seekg(a)</code>	將檔案的資料讀取位置移至 <code>a</code> ； <code>a</code> 為整數。
<code>seekg(a,b)</code>	將檔案的資料讀取位置，移至相對於 <code>b</code> 的位置 <code>a</code> 。 <code>a</code> 為整數， <code>b</code> 為 <code>ios_base::seekdir</code> 型別。
<code>tellg()</code>	取得目前檔案內的資料讀取位置，回傳整數數值。
<code>seekp(a)</code>	將檔案的資料寫入位置移至 <code>a</code> ； <code>a</code> 為整數。
<code>seekp(a,b)</code>	將檔案的資料寫入位置，移至相對於 <code>b</code> 的位置 <code>a</code> 。 <code>a</code> 為整數， <code>b</code> 為 <code>ios_base::seekdir</code> 型別。
<code>tellp()</code>	取得目前檔案內的資料寫入位置，回傳整數數值。

`ios_base::seekdir` 用於 `seekg()` 與 `seekp()` 函式設定移動檔案索引位置的起始位置；一共有以下 3 種起始位置：

常數	說明
<code>ios_base::beg</code>	從檔案開頭；也可以寫成 <code>ios::beg</code> 。
<code>ios_base::cur</code>	從目前的位置；也可以寫成 <code>ios::cur</code> 。
<code>ios_base::end</code>	從檔案的尾端；也可以寫成 <code>ios::end</code> 。

例如，檔案 `file` 已經開啓，現欲從目前的索引位置再移動 50 個位元組的位置開始讀取資料：

```
file.seekg(50, ios_base::cur);
```

或是要將檔案的索引位置重新移至檔案開頭：

```
file.seekg(0, ios_base::beg);
```

隨機讀取基本資料型別

基本資料型別諸如：`int`、`float`、`double`、`char` 等，都是以單個變數的方式儲存於檔案，所以儲存於檔案中的長度也等於基本資料型別的長度；例如 `int` 型別的變數長度等於 4 個位元組、`char` 型別的變數長度等於 1 個位元組。

因此，在讀取資料時的讀取長度就等於其資料型別的長度；例如：讀取 `int` 型別的資料，則使用 `sizeof(int)`、讀取 `double` 型別的資料則使用 `sizeof(double)`。

練習 1：讀取指定位置的資料

將 1、2、3...10 寫入檔案，並可指定要從檔案讀取第幾個數值。

■ 解說

要能夠從檔案中指定讀取特定位置的資料；因此，此檔案為二進位檔案。寫入的資料為 1-10 的整數；所以寫入的資料為 `int` 型別。例如要從二進位檔案 `file` 中寫入 `int` 型別的變數 `num`，如下所示：

```
file.write((char*)&num, sizeof(int));
```

要從檔案的索引位置 `no` 讀取資料，並儲存到變數 `num`，則如下所示：

```
file.seekg((no - 1) * sizeof(int), ios::beg);
file.read((char*)& num, sizeof(int));
```

`seekg()` 的移動起始位置為 `ios::beg`；因此，每一次都會從檔案的開頭計算索引位置。

■ 執行結果

```
讀取第幾筆數字 (1-10) : 4
第 4 筆資料 = 4
讀取第幾筆數字 (1-10) : 7
第 7 筆資料 = 7
讀取第幾筆數字 (1-10) : 1
第 1 筆資料 = 1
```

程式碼列表

```

1  #include <iostream>
2  #include <fstream>
3  using namespace std;
4
5  int main()
6  {
7      fstream file;
8      int no;
9      int num;
10
11     //----- 將 1,2,3...10 寫入檔案 -----
12     file.open("data", ios::binary | ios::out);
13     if(!file)
14     {
15         cout << " 無法建立檔案 " << endl;
16         exit(0);
17     }
18
19     for (int i = 1; i <= 10; i++)
20         file.write((char*)&i, sizeof(int));
21
22     file.close();
23
24     //-----
25     while (true)
26     {
27         cout << " 讀取第幾筆數字 (1-10) : ";
28         cin >> no;
29         if (no == -1)
30             exit(0);
31
32         if (no < 1 || no>10)
33         {
34             cout << " 輸入錯誤 " << endl;
35             exit(0);
36         }
37
38         file.open("data", ios::binary | ios::in);
39         if (!file)
40         {
41             cout << " 無法開啓檔案 " << endl;

```

```

42         exit(0);
43     }
44
45     file.seekg((no - 1) * sizeof(int), ios::beg);
46     file.read((char*)& num, sizeof(int));
47     file.close();
48     cout << " 第 " << no << " 筆資料 = " << num << endl;
49 }
50
51 system("pause");
52 }

```

程式講解

1. 程式碼第 1-3 行引入需要的標頭檔與宣告使用 `std` 命名空間。
2. 程式碼第 7-9 行宣告 3 個變數：`fstream` 類別的檔案變數 `file`、整數變數 `no` 用於指定要讀取的第幾筆資料、整數變數 `num` 用於儲存從檔案讀取的資料。
3. 程式碼第 12-17 行使用 `open()` 函式建立檔案 `data`，開檔模式為 `ios::binary|ios::out`；因此，此檔案作為寫入資料所用的二進位檔案。若檔案建立失敗則顯示錯誤訊息後結束程式。第 19-20 行使用 `for` 重複敘述將迴圈變數 `i` 寫入檔案；變數 `i` 的變化為 1-10，因此第 20 行使用 `write()` 函式將變數 `i` 寫入檔案，即是將數字 1-10 寫入檔案。第 22 行使用 `close()` 函式關閉檔案。
4. 程式碼第 25-49 行為 `while` 無窮迴圈。第 27-36 行讀取使用者所輸入的值並儲存於變數 `no`；若輸入的值為 -1 則結束程式。因為檔案內只有 10 筆資料，因此若輸入的值小於 1 或大於 10 則表示輸入錯誤。
5. 程式碼第 38-43 行使用 `open()` 開啓檔案 `data`，開檔模式為 `ios::binary|ios::in`，所以此檔案作為提供讀取資料之二進位檔案；若無法開啓檔案則顯示錯誤訊息並結束程式。第 45 行使用 `seekg()` 函式以及 `ios::beg` 常數，設定檔案索引位置為檔案開頭算起的第 $(no-1) \times \text{sizeof}(\text{int})$ 位元組的位置。
6. 程式碼第 46 行使用 `read()` 函式讀取資料並儲存到變數 `num`。第 47 行使用 `close()` 函式關閉檔案。第 48 行顯示讀取的變數 `num` 的值。

字元陣列資料

儲存至二進位檔案中的資料若是字元陣列型別的字串資料，使用 `seekg()` 函式設定檔案的索引位置時，由於字串本身是固定長度的字元陣列，則設定檔案的索引位置的方式應為：

$$(no - 1) \times \text{sizeof}(\text{字元陣列的長度})$$

或是：

$$(no - 1) \times \text{sizeof}(\text{字串變數})$$

例如：二進位檔案 `file` 中儲存字串資料，這些字串都是長度等於 10 的字元陣列。因此，要從檔案中移動索引位置到第 `no` 筆的字串資料位置，則如下所示：

$$(no - 1) \times \text{sizeof}(\text{char}) \times 10$$

若字串資料宣告為字元陣列變數 `str`，則可以簡化為：

```
char str[30];
(no - 1) × sizeof(str)
```

練習 2：讀取指定索引位置的陣列資料

寫一程式可以持續輸入任意字串；輸入字串後立即將字串儲存於二進位檔案，並可以指定要從檔案讀取某筆字串。

■ 解說

爲了要能使用隨機存取的方式，從檔案中讀取特定索引位置的字串資料，因此要採用二進位檔案；並且字串要以固定長度的字元陣列的方式儲存。假設每個字元陣列型別的字串長度等於 30，則從檔案讀取指定索引位置的字串如下所示：

```
1 fstream file;
2 char str[30];
3 int no;
4     :
5 file.seekg((no-1)*sizeof(str), ios::beg);
6 file.read(str, sizeof(str));
```

程式碼第 2 行的變數 `str` 用於儲存從檔案讀取的字串資料，其長度等於 30 個字元。第 3 行變數 `no` 爲指定要從檔案讀取的第幾筆字串。第 5 行使用 `seekg()` 函式設定檔案的索引位置 $(no-1) \times \text{sizeof}(\text{str})$ 。第 6 行讀取資料並儲存至變數 `str`。

■ 執行結果

```
輸入字串 ( 長度小於 30 個字元 ) : This is a book.
輸入字串 ( 長度小於 30 個字元 ) : 你好，早安。
輸入字串 ( 長度小於 30 個字元 ) : Hello, how are you?
輸入字串 ( 長度小於 30 個字元 ) : -1
讀取第幾筆數字 (1-3) : 3
第 3 筆資料 = Hello, how are you?
讀取第幾筆數字 (1-3) : 2
第 2 筆資料 = 你好，早安。
```

程式碼列表

```
1  #include <iostream>
2  #include <fstream>
3  #include <string>
4  using namespace std;
5
6  int main()
7  {
8      fstream file;
9      int no,num=0;
10     string str;
11     char wstr[30],rstr[30];
12
13     //-----
14     file.open("data", ios::binary | ios::out);
15     if (!file)
16     {
17         cout << " 無法建立檔案 " << endl;
18         exit(0);
19     }
20
21     while (true)
22     {
23         cout << " 輸入字串 ( 長度小於 30 個字元 ) : ";
24         getline(cin, str);
25
26         if (str=="-1")
27         {
28             file.close();
29             break;
30         }
31         else
32         {
33             strncpy_s(wstr, str.c_str(), 29);
34             file.write(wstr, sizeof(wstr));
35             if (!file.good())
36             {
37                 cout << " 字串寫入錯誤 " << endl;
38                 file.close();
39                 exit(0);
40             }
41             num++; // 字串數量加 1
```



```

42     }
43 }
44
45 file.close();
46
47 //-----
48 while (true)
49 {
50     cout << " 讀取第幾筆數字 (1-" << num << ") : ";
51     cin >> no;
52     if (no == -1)
53         exit(0);
54
55     if (no < 1 || no>num)
56     {
57         cout << " 輸入錯誤 " << endl;
58         continue;
59     }
60
61     file.open("data", ios::binary | ios::in);
62     if (!file)
63     {
64         cout << " 無法開啓檔案 " << endl;
65         exit(0);
66     }
67
68     file.seekg((no-1)*sizeof(rstr), ios::beg);
69     file.read(rstr, sizeof(rstr));
70     file.close();
71
72     cout << " 第 " << no << " 筆資料 = " << rstr << endl;
73 }
74
75 system("pause");
76 }

```

程式講解

1. 程式碼第 1-4 行引入所需要的標頭檔與宣告使用 `std` 命名空間。
2. 程式碼第 9 行宣告整數變數 `no` 與 `num`；`no` 用於指定要讀取第幾筆的字串資料，`num` 則用於儲存輸入了多少筆的字串資料。第 11 行宣告字元陣列型別的變數 `wstr` 與 `rstr`，其長度都等於 30；分別作為寫入檔案的字串資料與儲存從檔案讀取的字串資料。

3. 程式碼第 14-19 行使用 `open()` 函式建立檔案 `data`，開檔模式為 `ios::binary|ios::out`；因此，此檔案作為提供寫入資料的二進位檔案。若無法建立檔案則顯示錯誤訊息並結束程式。
4. 程式碼第 21-43 行為 `while` 無窮迴圈，用於持續寫入字串到檔案；結束輸入字串之後會執行第 45 行使用 `close()` 函式關閉檔案。

第 23-24 行顯示提示輸入的訊息，並將輸入的字串資料儲存於變數 `str`。第 26-42 行為 `if...else` 判斷敘述；第 26-30 行判斷若輸入的資料等於 `"-1"`，表示不再輸入資料，所以使用 `close()` 函式關閉檔案 `data` 並離開 `while()` 重複敘述；否則執行第 32-42 行將資料寫入檔案。

第 33 行先使用 `strncpy_s()` 函式將字串 `str` 的前 29 個字元轉換為字元陣列形式的資料，並儲存到字元陣列變數 `wstr`。第 34 行使用 `write()` 函式將變數 `wstr` 寫入檔案。第 35-40 行使用 `good()` 函式判斷若寫入資料失敗，則關閉檔案並結束程式。第 41 行將變數 `num` 累加 1 表示多了一筆字串資料。

5. 第 48-73 行為 `while` 無窮迴圈，用於開啓檔案並從檔案中讀取指定索引位置的資料。第 50-53 行顯示要讀取第幾筆資料的提示訊息，並將輸入的資料儲存於變數 `no`；若輸入 `-1` 則結束程式。第 55-59 行判斷輸入的資料小於 1 或是大於資料筆數 `num`，則顯示錯誤訊息並重新輸入資料。
6. 程式碼第 61-66 行使用 `open()` 函式開啓檔案 `data`，開檔模式為 `ios::binary|ios::in`；因此，此檔案作為提供讀取資料的二進位檔案。若無法開啓檔案則顯示錯誤訊息並結束程式。
7. 程式碼第 68 行使用 `seekg()` 函式，從檔案開頭 `ios::beg` 移動索引位置到 $(no-1)*sizeof(rstr)$ ，第 69 行使用 `read()` 函式讀取一筆字串資料並儲存到變數 `rstr`，第 70 行使用 `close()` 關閉檔案，第 72 行顯示讀取的資料。

26-2 隨機寫入資料

隨機寫入資料與隨機讀取資料的方式與步驟相同：檔案必須為二進位檔案、需使用 `seekp()` 移動檔案索引位置，再使用 `write()` 函式將資料寫入檔案的指定索引位置。

然而，隨意的將資料寫入檔案的任意位置，並沒有多大的意義。因為資料通常是依照一定的先後時間或順序寫入檔案；因此，除非有特別的理由與應用，才需要特別指定檔案的寫入位置。

修改檔案內的資料便是隨機寫入的應用。當檔案內儲存很多的資料，例如：數百筆、數千筆甚至數萬筆的資料，此時不可能把檔案內所有的資料先讀到陣列、修改陣列裡的資料、然後再重新把這麼多的資料重新寫入檔案。因此，最好的方法是直接指定要修改哪筆資料之後，直接將新資料寫入檔案指定的索引位置。

練習 3：修改檔案中的資料

寫一程式可以持續輸入任意字串；輸入字串後立即將字串儲存於二進位檔案。結束輸入字串之後，可以直接修改檔案中的資料。

■ 解說

本練習用於示範隨機寫入檔案的方式：直接設定檔案中的索引位置，並將新的資料覆蓋舊的資料，藉以完成修改資料的功能。

此外，新增資料與修改資料都是寫入檔案的操作，因此會有相似的程式敘述；所以可以在同一個自訂函式中完成此 2 種類似但不同的功能。

■ 執行結果

第 1-3 行輸入 3 個字串並儲存於檔案；第 4 行輸入 "-1" 結束輸入字串。第 5-7 行顯示儲存於檔案中的 3 個字串。第 8-9 行輸入欲修改資料的編號 3，與輸入新的字串資料。第 10-12 行顯示修改後儲存於檔案中的資料；可見到第 3 筆資料已從原來的字串 "pencil" 改為新的字串 "orange"。

```
輸入字串 ( 長度小於 30 個字元，輸入 -1 結束 ) : hello
輸入字串 ( 長度小於 30 個字元，輸入 -1 結束 ) : apple
輸入字串 ( 長度小於 30 個字元，輸入 -1 結束 ) : pencil
輸入字串 ( 長度小於 30 個字元，輸入 -1 結束 ) : -1
hello
apple
pencil
輸入欲修改第幾筆資料 : 3
輸入字串 ( 長度小於 30 個字元，輸入 -1 結束 ) : orange
hello
apple
orange
```

■ 程式碼列表

```
1 #include <iostream>
2 #include <string>
3 #include <fstream>
4 using namespace std;
5
6 //----- 讀取資料 -----
7 bool readData()
```

```

8 {
9     fstream file;
10    char str[30];
11
12    file.open("data", ios::binary | ios::in);
13    if (!file)
14    {
15        cout << " 無法開啓檔案 " << endl;
16        return false;
17    }
18    else
19    {
20        while (file.read(str, sizeof(str)))
21            cout << str << endl;
22
23        file.close();
24        return true;
25    }
26 }
27
28 //----- 輸入資料與修改資料 -----
29 // mode=0: 新增資料  mode=1: 修改資料
30 void addData(int mode,int &num, int openMode)
31 {
32     fstream file;
33     char wstr[30];
34     string str;
35     int no;
36
37     file.open("data", openMode);
38     if (!file)
39     {
40         cout << " 無法建立檔案 " << endl;
41         return;
42     }
43
44     if (mode == 0)
45         num = 0;
46     else
47     {
48         cout << " 輸入欲修改第幾筆資料 : ";
49         cin >> no;

```

```

50     cin.ignore(80, '\n');
51     if (no<1 || no>num)
52     {
53         cout << " 輸入錯誤 ";
54         return;
55     }
56 }
57
58 do{
59     cout << " 輸入字串 ( 長度小於 30 個字元，輸入 -1 結束 ) : ";
60     getline(cin, str);
61
62     if (str == "-1")
63     {
64         file.close();
65         break;
66     }
67     else
68     {
69         strncpy_s(wstr, str.c_str(), 29);
70         if (mode==1)
71             file.seekp((no - 1) * sizeof(wstr), ios::beg);
72         file.write(wstr, sizeof(wstr));
73         if (!file.good())
74         {
75             cout << " 字串寫入錯誤 " << endl;
76             file.close();
77             exit(0);
78         }
79         if(mode==0)
80             num++; // 字串數量加 1
81     }
82 } while (mode == 0);
83
84 file.close();
85 }
86
87 //-----
88 int main()
89 {
90     int num = 0;
91

```

```

92     // 新增資料
93     addData(0, num, ios::binary | ios::out);
94     readData();
95     // 修改資料
96     addData(1, num, ios::binary | ios::out|ios::in);
97     readData();
98
99     system("pause");
100 }

```

程式講解

1. 程式碼第 1-4 行引入所需要的標頭檔與宣告使用 `std` 命名空間。
2. 程式碼第 7-26 行為自訂函式 `readData()` 的程式本體，此函式用於顯示檔案中的所有資料。第 12-17 行使用 `open()` 函式開啓檔案 `data`，開檔模式為 `ios::binary|ios::in`，所以此檔案作為提供讀取資料的二進位檔案；若無法開啓檔案則顯示錯誤訊息並回傳 `false`，否則執行第 19-25 行讀取檔案中的資料。

第 20-21 行為 `while` 重複敘述；第 20 行使用 `while` 重複敘述讀取檔案中的字串資料，並儲存於字元陣列變數 `str`，第 21 行顯示從檔案讀取的資料 `str`。讀取所有資料之後，第 23-24 行使用 `close()` 函式關閉檔案，並回傳 `true` 表示讀取檔案成功。

3. 程式碼第 30-85 行為自訂函式 `addData()`，用於新增資料與修改資料，並帶有 3 個參數。第 1 個參數 `mode` 用於區分要執行新增資料還是修改資料的功能，第 2 個參數 `num`（使用參考呼叫）若是在新增資料模式下，用於儲存新增資料之後的資料總筆數；若是在修改資料的模式之下，參數 `num` 則表示資料的總筆數。第 3 個參數 `openMode` 為開檔模式，因為新增資料與修改資料所使用的開檔模式並不相同。

第 37-42 行依據所傳入的開檔模式 `openMode` 開啓檔案，若開啓檔案失敗則顯示訊息並返回呼叫者。

第 44-56 行為 `if...else` 判斷敘述。第 44 行判斷若是要新增資料，則先將參數 `num` 設定為 0，表示目前尚未新增任何資料；否則表示要執行修改資料，則執行第 47-56 行的程式敘述。第 48-49 行顯示修改資料的輸入提示訊息、讀取輸入的資料並儲存於變數 `no`。第 51-55 行判斷若輸入的資料編號超過範圍則顯示錯誤訊息並返回呼叫者。

4. 第 58-82 行為 `do...while` 後測式的重複敘述。第 59-60 行顯示輸入字串的提示訊息並讀取輸入的字串，將輸入的字串儲存於變數 `str`。第 62-81 行為 `if...else` 判斷敘述，第 62-66 行判斷若輸入的資料等於 `"-1"` 表示停止新增資料，使用 `close()` 函式關閉檔案並離開 `do...while()` 重複敘述；否則執行第 68-81 行程式敘述。

第 69 行使用 `strncpy_s()` 函式將字串 `str` 轉換為字元陣列並儲存於字元陣列變數 `wstr`，第 70-71 行判斷若參數 `mode` 等於 1 表示要執行修改資料，所以使用函式 `seekp()` 將檔案的索引位置移到正確的寫入位置 $(no-1)*sizeof(wstr)$ 。

第 72 行使用 `write()` 函式將字串 `wstr` 寫入檔案。第 73-78 行使用 `good()` 函式判斷若資料寫入檔案失敗，則顯示錯誤訊息、關閉檔案並返回呼叫者。若資料成功寫入檔案則執行第 79-80 行，先判斷若是在新增資料的模式下，則將資料筆數 `num` 加 1。

因為是後測式的 `do...while` 重複敘述，因此第 82 行判斷若是在新增資料的模式下，則再返回第 58 行重複執行 `do...while` 重複敘述。

5. 第 90 行宣告整數變數 `num`，此變數用於記錄一共新增了多少筆的資料，初始值設定為 0。第 93 行呼叫自訂函式 `addData()` 新增資料，並傳入 3 個引數。第 1 個引數 0 表示要執行新增資料的功能，第 2 個引數 `num` 等到新增資料結束之後，會等於新增資料的數量。第 3 個引數為新增資料所需要的開檔模式。第 94 行呼叫 `readData()` 顯示檔案裡的所有資料。

第 96 行呼叫自訂函式 `addData()` 修改資料，並傳入 3 個引數。第 1 個引數 1 表示要執行修改資料的功能，第 2 個引數 `num` 為資料的數量。第 3 個引數為修改資料所需要的開檔模式。第 97 行呼叫 `readData()` 顯示修改資料後檔案裡的所有資料。

三、範例程式解說

1. 建立專案，程式碼第 1-5 行引入需要的標頭檔與宣告使用 `std` 命名空間。

```

1 #include <iostream>
2 #include <fstream>
3 #include <string>
4 #include <conio.h>
5 using namespace std;
```

2. 程式碼第 7-11 行定義商品基本資料的結構 `_PRODUCT`，包含 2 個變數成員：產品名稱 `name` 與價錢 `price`。

```

7 struct _PRODUCT
8 {
9     char name[20];
10    int price;
11 };
```

3. 程式碼第 14-22 行為自訂函式 `showMenu()` 的程式本體，用於顯示功能選單。

```

14 void showMenu()
15 {
16     system("cls");
17     cout << "1. 新增資料 " << endl;
18     cout << "2. 顯示資料 " << endl;
19     cout << "3. 查詢資料 " << endl;
20     cout << "4. 結束 " << endl;
21     cout << "輸入選擇 (1-4): ";
22 }

```

4. 程式碼第 25-63 行為自訂函式 `addData()` 的程式本體，用於新增產品基本資料，並將之儲存於檔案。第 28 行宣告 `_PRODUCT` 型別的變數 `product`，用於儲存輸入的產品基本資料，第 29 行宣告布林型別的變數 `fgRun`，用於控制是否執行第 40-60 行的 `while` 重複敘述。第 33-38 行使用 `open()` 函式建立檔案 `data`，開檔模式為 `ios::binary|ios::out| ios::app`；因此，此檔案作為附加資料的二進位檔案。

```

25 void addData()
26 {
27     fstream file;
28     _PRODUCT product;
29     bool fgRun = true;
30
31     cout << "\n---- 新增資料 ----";
32
33     file.open("data", ios::binary | ios::out|ios::app);
34     if (!file.is_open())
35     {
36         cout << "無法建立 / 開啓檔案 " << endl;
37         fgRun = false;
38     }

```

第 40-60 行為 `while` 重複敘述，當變數 `fgRun` 等於 `true` 時會重複執行 `while` 重複敘述。第 42-43 行顯示輸入的提示訊息與讀取輸入的產品名稱 `product.name`。第 44-59 行為 `if...else` 判斷敘述，第 44-45 行判斷若輸入的產品名稱等於 `"-1"` 則將變數 `fgRun` 設定為 `false` 表示停止輸入資料；否則執行第 47-59 行。

第 48-49 行顯示輸入的提示訊息與讀取輸入的產品價錢 `product.price`。第 51 行將產品的基本資料 `product` 寫入檔案，第 52-58 行判斷若寫入資料失敗，則顯示錯誤訊息並將變數 `fgRun` 設定為 `false`；否則顯示資料寫入成功的訊息。

```

40     while (fgRun)
41     {
42         cout << "\n 輸入商品名稱 ( 輸入 -1 結束 ) : ";
43         cin >> product.name;
44         if (strcmp(product.name, "-1") == 0)
45             fgRun = false;
46         else
47         {
48             cout << " 輸入價錢 : ";
49             cin >> product.price;
50
51             file.write((char*)& product, sizeof(_PRODUCT));
52             if (!file.good())
53             {
54                 cout << " 無法寫入資料 " << endl;
55                 fgRun = false;
56             }
57             else
58                 cout << " 已新增資料 " << endl;
59         }
60     }
61
62     file.close();
63 }

```

5. 程式碼第 66-84 行為自訂函式 `showData()` 的程式本體，此函式用於顯示檔案中的所有產品基本資料。第 69 行宣告 `_PRODUCT` 型別的變數 `product`，用於儲存從檔案讀取的資料。第 70 行宣告變數 `no`，表示這是從檔案讀取的第幾筆資料。第 74 行使用 `open()` 函式開啓檔案 `data`，開模式為 `ios::binary|ios::in`，表示此檔案作為讀取資料之二進位檔案。

第 75-83 行為 `if...else` 判斷敘述，第 75-76 行判斷若無法開啓檔案則顯示錯誤訊息，否則執行 78-83 行的程式敘述。第 79 行使用 `while` 重複敘述與 `read()` 函式從檔案中讀取資料，並儲存於變數 `product`。第 80-81 行顯示所讀取的資料 `product`。讀取資料結束後第 82 行使用 `close()` 關閉檔案。

```

66 void showData()
67 {
68     fstream file;
69     _PRODUCT product;
70     int no = 1;

```

```

71
72     cout << "\n---- 顯示資料 ----" << endl;
73
74     file.open("data", ios::binary | ios::in);
75     if (!file.is_open())
76         cout << " 無法開啓檔案 " << endl;
77     else
78     {
79         while (file.read((char*)& product, sizeof(_PRODUCT)))
80             cout << no++ << ". 名稱：" << product.name << \
81                 ", 價錢：" << product.price << endl;
82         file.close();
83     }
84 }

```

6. 程式碼第 87-119 行為自訂函式 `queryData()`，用於查詢檔案中的資料。第 90 行宣告 `_PRODUCT` 型別的變數 `product`，用於儲存從檔案中讀取的資料。第 91 行宣告整數變數 `no`，用於儲存欲查詢的是第幾筆資料。

第 95 行使用 `open()` 函式開啓檔案 `data`，開檔模式為 `ios::binary|ios::in`，表示此檔案作為讀取資料之二進位檔案。第 96-100 行判斷若無法開啓檔案則顯示錯誤訊息並且返回呼叫者。第 102-103 行顯示輸入搜尋商品的提示訊息，並將輸入的資料儲存到變數 `no`。

第 105-117 行為 `if...else` 判斷敘述。第 105-106 行判斷若輸入的變數 `no` 小於 1，則顯示錯誤訊息；否則執行 108-117 行程式敘述。第 109 行使用函式 `seekg()` 將檔案的索引位置移動至 `(no-1)*sizeof(_PRODUCT)`。第 110 行使用 `read()` 函式從檔案中讀取資料後儲存到變數 `product`。第 112-116 行使用函式 `fail()` 判斷若能成功讀取資料，則第 113-114 行顯示此商品的基本資料，否則第 116 行顯示錯誤訊息。資料查詢結束後，第 118 行使用 `close()` 函式關閉檔案。

```

87 void queryData()
88 {
89     fstream file;
90     _PRODUCT product;
91     int no = 1;
92
93     cout << "\n---- 查詢資料 ----" << endl;

```

```

94
95     file.open("data", ios::binary | ios::in);
96     if (!file.is_open())
97     {
98         cout << " 無法開啓檔案 " << endl;
99         return;
100    }
101
102    cout << "\n 輸入欲查詢第幾筆資料 : ";
103    cin >> no;
104
105    if (no < 1)
106        cout << " 輸入錯誤 \n";
107    else
108    {
109        file.seekg((no - 1) * sizeof(_PRODUCT), ios::beg);
110        file.read((char*)& product, sizeof(_PRODUCT));
111
112        if(!file.fail())
113            cout << no << ". 名稱：" << product.name << \
114                ", 價錢：" << product.price << endl;
115        else
116            cout << " 超過範圍 \n";
117    }
118    file.close();
119 }

```

7. 開始於 `main()` 主函式中撰寫程式。程式碼第 123 行宣告整數變數 `sel`，用於儲存使用者所輸入的功能編號。第 125-149 行為 `while` 無窮迴圈，第 127 行呼叫自訂函式 `showMenu()` 顯示功能選單，第 128 行讀取使用者所輸入的功能編號，並儲存於變數 `sel`。

第 130-146 行為 `switch...case` 選擇敘述，並根據變數 `sel` 執行相對應的功能。第 132-133 為「新增資料」的功能，呼叫自訂函式 `addData()`。第 135-136 為「顯示資料」的功能，呼叫自訂函式 `showData()`。第 138-139 為「查詢資料」的功能，呼叫自訂函式 `queryData()`。程式碼第 141-142 行為「結束」的功能，呼叫函式 `exit()` 結束程式。第 144-145 行為 `default` 區塊；若輸入錯誤的選項則顯示錯誤訊息。第 147-148 行等待使用者按任一鍵後繼續執行。

```
123 int sel;
124
125 while (true)
126 {
127     showMenu();
128     cin >> sel;
129
130     switch (sel)
131     {
132         case 1: addData();
133             break;
134
135         case 2: showData();
136             break;
137
138         case 3: queryData();
139             break;
140
141         case 4: exit(0);
142             break;
143
144         default: cout << " 輸入錯誤，";
145             break;
146     }
147     cout << " 按鍵繼續 ...";
148     while (!_kbhit());
149 }
150
151 system("pause");
```

重點整理

1. 隨機存取需要將資料儲存為二進位檔案，並且儲存於檔案的每筆資料需要相同的長度。
2. `seekp()` 與 `tellp()` 函式使用於寫入資料的檔案；`seekg()` 與 `tellg()` 函式使用於讀取資料的檔案。

程式碼列表

```

1  #include <iostream>
2  #include <fstream>
3  #include <string>
4  #include <conio.h>
5  using namespace std;
6
7  struct _PRODUCT
8  {
9      char name[20];
10     int price;
11 };
12
13 //----- 顯示選單 -----
14 void showMenu()
15 {
16     system("cls");
17     cout << "1. 新增資料 " << endl;
18     cout << "2. 顯示資料 " << endl;
19     cout << "3. 查詢資料 " << endl;
20     cout << "4. 結束 " << endl;
21     cout << "輸入選擇 (1-4): ";
22 }
23
24 //----- 新增資料 -----
25 void addData()
26 {
27     fstream file;
28     _PRODUCT product;
29     bool fgRun = true;
30
31     cout << "\n---- 新增資料 ----";
32
33     file.open("data", ios::binary | ios::out|ios::app);
34     if (!file.is_open())
35     {
36         cout << "無法建立 / 開啓檔案 " << endl;
37         fgRun = false;
38     }
39
40     while (fgRun)
41     {

```

```

42     cout << "\n 輸入商品名稱 ( 輸入 -1 結束 ) : ";
43     cin >> product.name;
44     if (strcmp(product.name, "-1") == 0)
45         fgRun = false;
46     else
47     {
48         cout << " 輸入價錢 : ";
49         cin >> product.price;
50
51         file.write((char*)& product, sizeof(_PRODUCT));
52         if (!file.good())
53         {
54             cout << " 無法寫入資料 " << endl;
55             fgRun = false;
56         }
57         else
58             cout << " 已新增資料 " << endl;
59     }
60 }
61
62 file.close();
63 }
64
65 //----- 顯示資料 -----
66 void showData()
67 {
68     fstream file;
69     _PRODUCT product;
70     int no = 1;
71
72     cout << "\n---- 顯示資料 ----" << endl;
73
74     file.open("data", ios::binary | ios::in);
75     if (!file.is_open())
76         cout << " 無法開啓檔案 " << endl;
77     else
78     {
79         while (file.read((char*)& product, sizeof(_PRODUCT)))
80             cout << no++ << ". 名稱 : " << product.name << "\n", 價錢 : " << product.price << endl;
81         file.close();
82     }
83 }

```

```

84 }
85
86 //----- 查詢資料 -----
87 void queryData()
88 {
89     fstream file;
90     _PRODUCT product;
91     int no = 1;
92
93     cout << "\n---- 查詢資料 ----" << endl;
94
95     file.open("data", ios::binary | ios::in);
96     if (!file.is_open())
97     {
98         cout << " 無法開啓檔案 " << endl;
99         return;
100    }
101
102    cout << "\n 輸入欲查詢第幾筆資料： ";
103    cin >> no;
104
105    if (no < 1)
106        cout << " 輸入錯誤 \n";
107    else
108    {
109        file.seekg((no - 1) * sizeof(_PRODUCT), ios::beg);
110        file.read((char*)& product, sizeof(_PRODUCT));
111
112        if(!file.fail())
113            cout << no << ". 名稱：" << product.name << \
114                ", 價錢：" << product.price << endl;
115        else
116            cout << " 超過範圍 \n";
117    }
118    file.close();
119 }
120
121 int main()
122 {
123     int sel;
124
125     while (true)

```

```
126     {
127         showMenu();
128         cin >> sel;
129
130         switch (sel)
131         {
132             case 1: addData();
133                     break;
134
135             case 2: showData();
136                     break;
137
138             case 3: queryData();
139                     break;
140
141             case 4: exit(0);
142                     break;
143
144             default: cout << " 輸入錯誤，";
145                     break;
146         }
147         cout << " 按鍵繼續 ...";
148         while (!_kbhit());
149     }
150
151     system("pause");
152 }
```

本章習題

1. 增加範例 26 的功能：可以修改產品資料。
2. 增加範例 26 的功能：指定顯示某筆編號的資料。
3. 將多筆長度等於 5 的字串陣列寫入二進位檔案。寫一程式輸入要查詢第幾筆資料，並從檔案中讀取與顯示此筆資料。