第四章　陣列資料結構

4-1 資料結構

既然電腦是處理資料的能手，勢必也有儲存大量資料的能力，到底資料怎麼儲存才好？這牽涉到的問題很廣。從最基本的觀點來看，電腦除了能儲存大量資料外，還必須具有處理這些資料的能力，譬如，從中搜尋所要的資料，或可以容易的插入、刪除或更新資料內容。如果資料不多時，隨便儲存都不會產生大礙，然而當資料超過千萬筆時，如何可快速處理這些資料，這可就非常困難。然而為了讓資料可以被快處理，這就牽涉儲存方式，一般稱之為『資料結構』。

資料結構有許多種方法，譬如：陣列、鏈路、樹狀、B-Tree、B+-Tree 等等方法。B-Tree 以上的資料結構處理方法並不容易，但他們的搜尋速度最快，較適合巨量儲存環境；陣列結構的處理方法最簡單，但搜尋速度最慢，較適合小資料量的儲存。相對應的利用 B-Tree 資料結構所建立的資料庫系統，遠比利用陣列建立的貴很多。本書僅利用陣列要介紹資料結構的程式編寫技巧，至於較複雜的儲存方式，留給專門課程介紹。

4-2 無序陣列結構

4-2-1 無序陣列結構簡介

陣列資料結構的基本構想是利用二維陣列，陣列中每一行為一筆資料。基本上每一筆資料都有一個關鍵欄位，此關鍵欄位在所有其他資料中不可重複，又稱為『主鍵』。如果資料儲存於陣列中沒有依照主鍵的大小排列，則稱為『無序陣列結構』，反之，有依照主鍵的大小排列，則稱為『有序陣列結構』。圖 4-1 為兩者的比較，同樣都是儲存文具庫存資料，並以序號為主鍵，無序陣列沒有照次序儲存；有序陣列則有依照序號大小排列。為了簡化程式設計，以下的範例僅利用主鍵來說明。

(1) 無序陣列儲存

序號	品名	單價	數量
12	筆記簿	20	100
10	紅鉛筆	10	50
02	黑鉛筆	10	20
11	原子筆	24	118
45	簽字筆	28	123
17	印台	40	12
16	厚紙板	23	231
23	彩色紙	21	156
56	白板	150	321
34	白板筆	12	23
19	紅筆	20	15
21	筆芯	5	23

識別碼沒有依照大小排列

(2) 有序陣列儲存

序號	品名	單價	數量
10	筆記簿	20	100
12	紅鉛筆	10	50
13	黑鉛筆	10	20
17	原子筆	24	118
21	簽字筆	28	123
25	印台	40	12
28	厚紙板	23	231
31	彩色紙	21	156
34	白板	150	321
41	白板筆	12	23
43	紅筆	20	15
51	筆芯	5	23

識別碼有依照大小排列

圖 4-1 無序&有序陣列結構

4-2-2 範例研討：建立無序陣列

(A)系統功能：Ex4_1.java

吾人希望建立一個無序陣列，儲存空間為 50 筆，並具有插入與列印資料的功能，期望操作介面如下：

(1)具有 3 個選單如下：

```
D:\Java2_book\chap4>java Ex4_1
== 歡迎光臨 無序陣列管理系統 ==
(1) 列印無序陣列內容
(2) 插入陣列元素
(3) 離開系統
     84  74   38   27   80
```

(2)選擇列印資料(選擇 1) 如下：

```
            請輸入工作選項 =>1

== 列印無序陣列內容
32   58   40   56   76
```

```
16   10    9   62   94
94   53    1   27   71
25   77   63   85   34
69  59   77   76   98
```

(3)插入元素後其結果 (選擇 2 與 1) 如下：

```
                請輸入工作選項 =>2
請輸入欲插入元素 =>45
== 歡迎光臨 無序陣列管理系統 ==
(1) 列印無序陣列內容
(2) 插入陣列元素
(3) 離開系統
                請輸入工作選項 =>1
== 列印無序陣列內容
32   58   40   56   76
16   10    9   62   94
94   53    1   27   71
25   77   63   85   34
69   59   77   76   98
84  74   38   27   80
45
```

(B)系統分析：

圖 4-2 是我們需建立一個空間為 50 的陣列，空間為：num[0] ~ num[49]，其中利用一個 point 指標來顯示目前資料到甚麼地方，當陣列空間還未儲存資料則 point = -1，插入一筆資料後 point = point + 1，為 0 ，依此類推。如果刪除資料則 point = point -1，如果 point=-1 則表示陣列空間；point=49 則表示陣列已滿不可再存入。

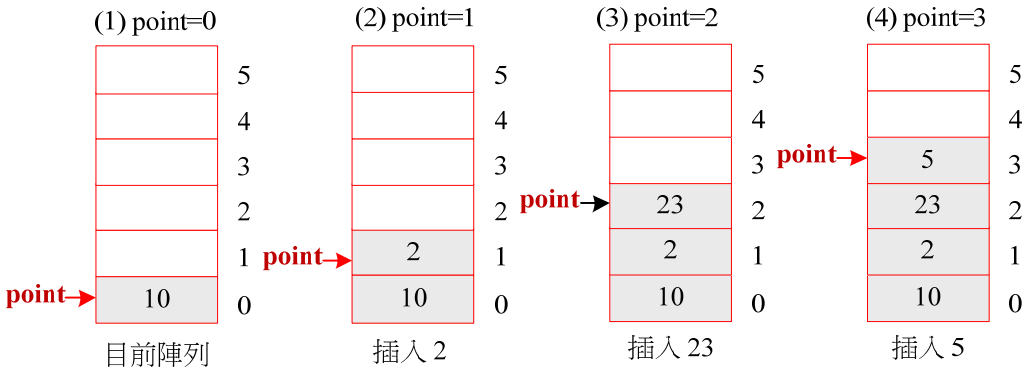

圖 4-2 無序陣列的插入元素(插入 5)

(C)程式範例

圖 4-3 為 Ex4_1.java 的程式架構，包含有 3 個『方法程式』(method) 與 2 個類別變數，其中 main 為主程式、disp_menu 是顯示功能選項程式、disp_array 為列印陣列內容程式。類別變數 num[] 是無序陣列的儲存空間；point 是指示目前陣列存放位置的游標，兩者類別變數允許所有方法程式存取。

其中『類別變數』(class variable)又稱為『整體變數』，它允許類別內所有『方法程式』存取。亦是，num[] 與 point 兩變數允許 main()、disp_menu() 與 disp_array() 等方法共同存取使用。如果這兩個變數在 main() 方法內宣告的話，則僅被 main() 存取，其他兩個方法就無法使用到他們。

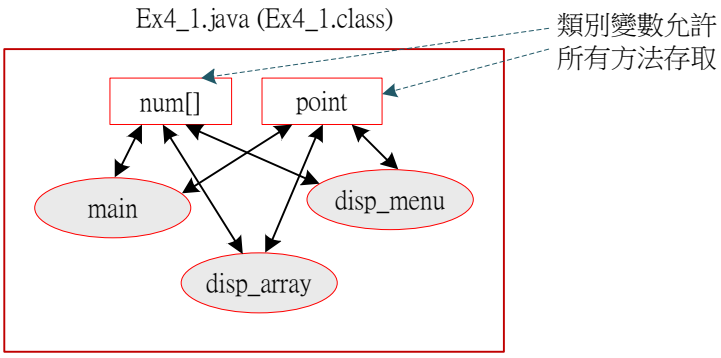

圖 4-3 Ex4_1 程式架構

```
01 //Ex4-1.java
02 /* 製作一個空間為 50 的無序陣列，並具有插入元素與
03
04 * 列印陣列內容的功能 */
05
06 import java.util.*;
07 public class Ex4_1{
08    static int num[] = new int[50];          // 宣告無序陣列空間
09
10    static int point;                        // 宣告游標變數
11    public static void main(String args[]) {
12          Scanner keyin = new Scanner(System.in);
13          Random random = new Random();
14          int value;           // 輸入元素
15
16          int select;          // 功能選擇
17
18          point = -1;          // 游標初值
19
          for(int i=0; i<30; i++){          //給予陣列初值
```

```
            value = random.nextInt(100);
            point = point +1;
            num[point] = value;
        }
        disp_menu();
        select = keyin.nextInt();
        while(select != 3) {
            switch(select) {
                case 1:
                    disp_array();
                    break;
                case 2:
                    if (point >=50) {
                        System.out.printf("陣列已滿無法插入!!\n");
                    }else {
                        System.out.printf("請輸入欲插入元素 =>");
                        value = keyin.nextInt();
                        point = point +1;
                        num[point] = value;
                     }
                     break;
                default:
                    System.out.printf("輸入錯誤 !! 請重新輸入\n");
            }
            disp_menu();
            select = keyin.nextInt();
        }
    }
    static void disp_menu() {
        System.out.printf("== 歡迎光臨 無序陣列管理系統 ==\n");
        System.out.printf("(1) 列印無序陣列內容\n");
        System.out.printf("(2) 插入陣列元素\n");
        System.out.printf("(3) 離開系統\n");
        System.out.printf("\t 請輸入工作選項 =>");
    }
    static void disp_array() {     /* 列印內容陣列 */
        System.out.printf("\n== 列印無序陣列內容\n");
        for(int i=0; i<=point; i++) {
            System.out.printf("%2d    ", num[i]);
            if((i+1) % 5 == 0)
                System.out.printf("\n");       //列印五筆，換行
```

```
            }
            System.out.printf("\n");                    // 列印完畢，換行
        }
    }
```

(D)編譯與執行如下：

圖 4-3-1

(E)重點說明

(1) 第 7~8 行：宣告產生類別變數 num[] 與 point。

(2) 第 16~20 行：給予 num[] 某些初值。

4-2-3 自我挑戰：無序陣列中元素處理

(A)系統功能：PM4_1.java

吾人希望在 Ex4_1 的無序陣列中，可選擇刪除某一個元素的功能，期望操作介面如下：

望操作介面如下：

(1)具有 4 個選單如下：

```
== 歡迎光臨 無序陣列管理系統 ==
(1) 列印無序陣列內容
(2) 插入陣列元素
(3) 刪除陣列元素
(4) 離開系統
        請輸入工作選項 =>1
```

(2)選擇列印資料(選擇 1) 如下：

```
            請輸入工作選項 =>1


== 列印無序陣列內容
83  65  26  31  44  79  62  19  11  26
22  22   1  63  51  48  44  68  49  87
85 23   4  41  93  65  97  64  54  54
```

(3)刪除元素後其結果 (選擇 3 與 1) 如下：

```
            請輸入工作選項 =>3
請輸入欲刪除元素 =>44
== 歡迎光臨 無序陣列管理系統 ==
(1) 列印無序陣列內容
(2) 插入陣列元素
(3) 刪除陣列元素
(4) 離開系統
            請輸入工作選項 =>1


== 列印無序陣列內容
83  65  26  31  79  62  19  11  26  22
22   1  63  51  48  44  68  49  87  86
23  4  41  93  65  97  64  54  54
```

(B)系統分析

圖 4-4 是由無序陣列中刪除某一元素的示意圖。它有以下步驟：

(1) 搜尋欲刪除的元素是否在陣列中，如沒有則結束。

(2) 如找到了，記錄其儲存位置(譬如 num[k])，則由 num[k+1] 以後到游標(num[point])的資料都往前移一位，將 num[k] 內容覆蓋。

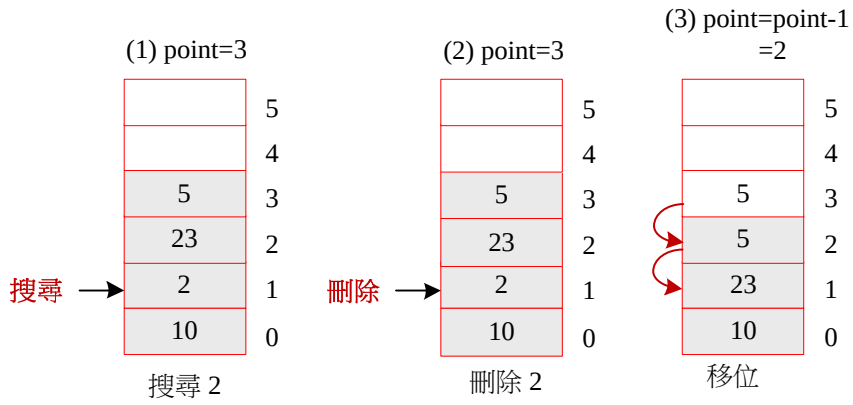

圖 4-4 無序陣列刪除元素(刪除 2)的運作

(C)程式提示

圖 4-5 為程式架構圖，由 Ex4_1.java 中增加了 Linear_search()，它的功能是搜尋所欲刪除元素的位置，如果找到的話，則回傳 location 位置，否則回傳 location=-1。

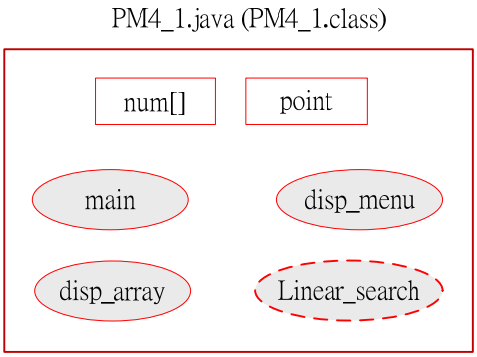

圖 4-5 PM4_1 程式架構

```
01 //PM4_1.java
02 /* 擴充 Ex4_1.java 使具有刪除元素的功能 */
03
04
05 import java.util.*;
06 public class PM4_1{
07    .....
08         disp_menu();
09         select = keyin.nextInt();
10         while(select != 4) {
11               switch(select) {
12                    ....
13                    case 3:
14                          System.out.printf("請輸入欲刪除元素 =>");
15                          value = keyin.nextInt();
16                          int location = Linear_serach(value);
17                          if (location == -1){
```

```
18                                  System.out.printf("陣列內沒有 %d 元素\n",
19 value);
20                          }else {
21                              for(int i = location;i<=point;i++)
22                                  num[i] = num[i+1];
23                          }
24                          point = point - 1;
25                          break;
26                  default:
27                      System.out.printf("輸入錯誤 !! 請重新輸入\n");
28              }
29              disp_menu();
30              select = keyin.nextInt();
31          }
32      }
33      ....
34 ....
35      static int Linear_serach(int value){
36          ....
37          請參考 Ex2_4 線性搜尋法
38          .....
39      }
.....
```

(D)編譯與執行如下：

圖 4-5-1

4-3 有序陣列結構

4-3-1 有序陣列結構簡介

圖 4-6 為有序陣列結構的示意圖，資料在陣列中存放有依照某一個關鍵字(或稱主鍵)的大小排列。這種排列方式的搜尋速度會比無序排列快許多。如果資料沒有依照大小排序，搜尋某一筆資料必須採用『線性搜尋法』，也就是由資料最前頭，一筆接一筆的比對，找出所要資料的位置。如果資料有一萬筆，運氣好第一筆就找到，運氣不好，找了一萬筆都沒有找到(速度為 n)。如果採用有序陣列排序的話，可以採用『二分搜尋法』，平均搜尋的速度是 log2n，n 表示資料筆數，如此速度就比有無序排序快許多。但因資料有依照主鍵大小排序，在作插入的時候，必須找到適當位置才可插入，而且必須經過適當的移位，才可以移動出一個空間，這遠比無序排列困難許多。

序號	品名	單價	數量
10	筆記簿	20	100
12	紅鉛筆	10	50
13	黑鉛筆	10	20
17	原子筆	24	118
21	簽字筆	28	123
25	印台	40	12
28	厚紙板	23	231
31	彩色紙	21	156
34	白板	150	321
41	白板筆	12	23
43	紅筆	20	15
51	筆芯	5	23

識別碼有依照大小排列

圖 4-6 有序陣列的儲存順序

4-3-2 範例研討：建立有序陣列

(A)程式功能：Ex4_2.java

吾人需要一個空間為 50 的有序陣列結構，來驗證陣列的處理方式，期望執行後能顯示下列結果。

```
D:\Java2_book\chap4>javac Ex4_2.java

D:\Java2_book\chap4>java Ex4_2
```

```
== 目前序陣列有 40 筆資料 ==
 2   4   6   8  10  12  14  16  18  20
22  24  26  28  30  32  34  36  38  40
42  44  46  48  50  52  54  56  58  60
62 64  66  68  70  72  74  76  78  80
```

(B)程式範例

圖 4-7 為其程式架構，包含 num[] 與 point 兩個類別變數，與一個 main() 主程式。

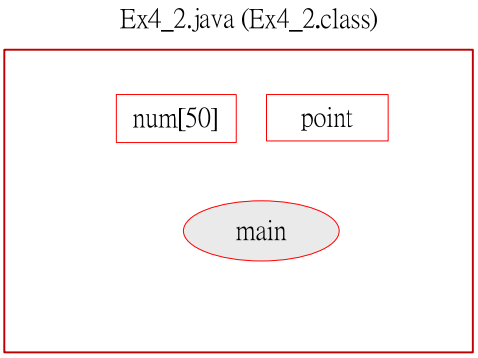

圖 4-7 Ex4_2 程式架構

```
01 //Ex4_2.java
02 /* 製作有序陣列的基本架構 */
03
04
05 import java.util.*;
06 public class Ex4_2{
07    static int num[] = new int[50];        // 宣告陣列空間
08    static int point;                      // 宣告游標變數
09    public static void main(String args[]) {
10          point = -1;          // 游標初值
11          for (int i=0; i< 40; i++){          //給予陣列初值
12                num[i] = (i+1) *2;              //存入有序資料
13                point = point + 1;              //游標指示目前位置
14          }
15 // 列印陣列內容
16          System.out.printf("\n== 目前序陣列有 %d 筆資料 ==\n", point+1);
17          for(int i=0; i<=point; i++) {
18                System.out.printf("%2d   ", num[i]);
19                 if((i+1) % 10 == 0)
20                      System.out.printf("\n");      //列印 10 筆, 換行
21
22
23
24
```

```
25          }
            System.out.printf("\n");                    // 列印完畢，換行
        }
    }
```

4-3-3 範例研討：二分搜尋法

(A)程式功能：Ex4_3.java

數學老師利用一個二維陣列 score[][] 儲存某一班級學生的成績，陣列第一個元素 score[k][0] 存放學生學號，由 411101 ~ 411150；第二個元素 score[k][1] 存放數學成績，由 00 ~ 100 分，如：{411101, 70}, {411102, 80}, {411103, 75}, {411104, 90}, {411105, 85}, {4111106, 65}, {411107, 83}, {411108, 78}, {411109, 67}, (411110, 72)...等等。陣列是依照學號由低到高排列。

請編寫一程式，允許輸入學生學號，則輸出該學生的成績。程式中先寫一小程式填入陣列內每一位同學的學號與成績，成績採 00 ~ 100 之間的亂數，之後印出全班成績，接著再編寫一個二分搜尋程式。期望操作介面如下：

```
D:\Java2_book\chap4>javac Ex4_3.java

D:\Java2_book\chap4>java Ex4_3
=== 411101 ~ 411150 成績列印 ===
  7  50  73  24   8  11  93  68  68   0
 52   5  39  26  48  31  49   3  77  71
 29   3  62  96  17  26  65  65   1  43
 66  68  27  92   1   8  69  62  66  51
 44  31  46  61  14  95   3  12   2  50
請輸入欲尋找的學生學號 => 411120

學號 411120 成績是 71
```

(B)系統分析

『二分搜尋法』是由已排序(由大到小或由小到大排列)的陣列裡，搜尋某一筆資料，圖 4-8 為其運作程序。假設 a 陣列已由小到大排序完成(a[low] ~ a[high])，key 變數儲存欲尋找的資料，首先比對 a 陣列的中間元素(mid)的內容，如果 key = a [mid]，則資料找

到；如果 key < a[mid]，則表示資料位於 a[low] 與 a[mid] 之間，則 mid 取代 high；如果 key > a[mid]，則資料位於 a[mid] ~ a[high] 之間，mid 取代 low 內容。沒有找到資料的話，則將搜尋目標移到前半段或後半段繼續尋找，但還是由 a[low] ~ a[high] 搜尋 (low 或 high 可能會改變)；如此依此類推，一直到找到資料或搜尋完畢為止。由此可見，此演算法每次將所有陣列資料缺切一半，再決定位於前段或後段，因此稱之為『二分搜尋法』。

此演算法較困難的地方是，如果找不到資料，如何去判斷與結束。其實，我們可以掌握一個重點，當 high 還是大於 low 的時候，表示還有資料未搜尋到；如果 high 小於或等於 low 時，表示已搜尋完所有資料，還未找到資料，則表示沒有該筆資料。

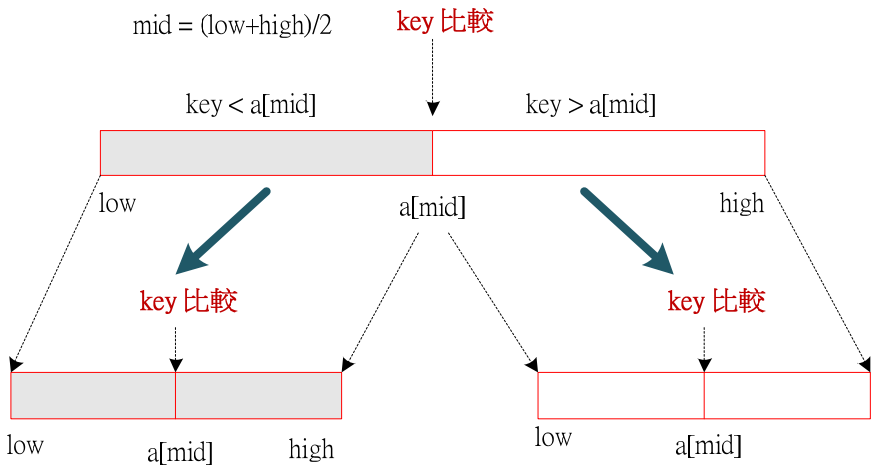

圖 4-8 二分搜尋法

(C)程式範例

```
01  //Ex4_3.java
02  // 二分搜尋法
03
04
05  import java.util.Scanner;
06  import java.lang.Math;
07  public class Ex4_3 {
08          public static void main(String args[]) {
09                  Scanner keyin = new Scanner(System.in);
10                  int a[][], value, flag, low, high, mid;
11                  a = new int[50][2];
12                  int number = 411101;
13                  for(int i=0; i<a.length; i=i+1){
14                          a[i][0] = number + i;
15                          a[i][1] = (int)(Math.random()*100);
16                  }
```

```
17              /* 列印全班成績 */
18
19              System.out.printf("=== 411101 ~ 411150 成績列印 ===\n");
20              for(int i=0; i<a.length; i++) {
21                      System.out.printf("%4d", a[i][1]);
22                      if((i+1) % 10 == 0)
23                              System.out.printf("\n");
24              }
25
26              System.out.printf("請輸入欲尋找的學生學號 => ");
27              value = keyin.nextInt();
28
29              /*    二分搜尋法 */
30              low = 0; high = 49;mid=0;
31              flag=0;                         // 設定是否找到旗號
32              while((low+1) < high) {         // 陣列是否搜尋完
33                      mid = (low + high)/2;
34                      if (value == a[mid][0]) {       // 比對陣列中間元素
35                              flag = 1;               // 找到了, 設定旗號並離開
36                              break;
37                      }
38                      else if (value > a[mid][0])     // 在陣列的後半段
39                              low = mid;
40                      else                            // 在陣列的前半段
41                              high = mid;
42              }
43
44              if (flag == 1)
45                 System.out.printf("學號 %d 成績是 %d\n", a[mid][0], a[mid][1]);
46              else
47                 System.out.printf("沒有 %d 學號學生\n", value);
48      }
49  }
50
51
```

(D)程式重點分析：

(1) 第 28~41 行：**二分搜尋演算法**。

(2) 第 29 行：**『low=0; high=49;』**。設定原陣列搜尋範圍。

(3) 第 30 行：**『flag=0』**。設定搜尋旗號為否（還未找到）。

(4) 第 31~41 行：『**while((low+1) < high) { ….}**』。其中 (low+1) < high 表示元素較高的指標 high 還是高於較低的 low，則陣列還未搜尋完畢。

(5) 第 32 行：『**mid = (high + low)/2**』。陣列元素索引 high 與 low 之間的中間元素的索引 mid。

(6) 第 33 行『**if(value == a[mid] { …}**』。成立的話，表示找到該資料，則設定旗號並中斷離開。

(7) 第 37 行：『**else if (value > a[mid][0]) { …}**』。如未找到，但比中間值 a[mid] 大的話，則設定搜尋後半段（low = mid，但 high 未改變）。

(8) 第 39 行：『**else high = mid**』。都不是的話，則設定搜尋前半段（low 未改變）。

4-3-3 範例研討：有序陣列插入元素

(A)程式功能 Ex4_4.java

吾人希望建立一個可供插入元素的有序陣列，其功能如下：

(1) 具有顯示陣列內容與插入元素的選項，選單如下：

```
D:\Java2_book\chap4>javac Ex4_4.java

D:\Java2_book\chap4>java Ex4_4
== 歡迎光臨 有序陣列管理系統 ==
(1) 列印有序陣列內容
(2) 插入陣列元素
(3) 離開系統
            請輸入工作選項 =>
```

(2) 當選擇顯示陣列內容(選擇 1)，如下：

```
            請輸入工作選項 =>1

== 目前序陣列有 30 筆資料 ==
 2   4   6   8  10  12  14  16  18  20
22  24  26  28  30  32  34  36  38  40
42  44  46  48  50  52  54  56  58  60
```

(3) 插入元素(選擇 2)，如下： (插入元素 15 後，再顯示陣列內容)

```
            請輸入工作選項 =>2
請輸入欲插入元素 =>15
== 歡迎光臨 有序陣列管理系統 ==
(1) 列印有序陣列內容
(2) 插入陣列元素
(3) 離開系統
            請輸入工作選項 =>1

== 目前序陣列有 31 筆資料 ==
 2   4   6   8  10  12  14  15  16  18
20  22  24  26  28  30  32  34  36  38
40 42  44  46  48  50  52  54  56  58
60
```

(B)系統分析

如欲將元素插入有序陣列內，必須搜尋出他適當的位置再插入，才會保持原來有序排列。以圖 4-9 為例，有序陣列 num={2, 5, 10, 23, 34}，並以 point=4 表示目前游標在 4。如欲插入 8 的話，則必須將大於 8 的元素往後移一位，再將 8 插入。

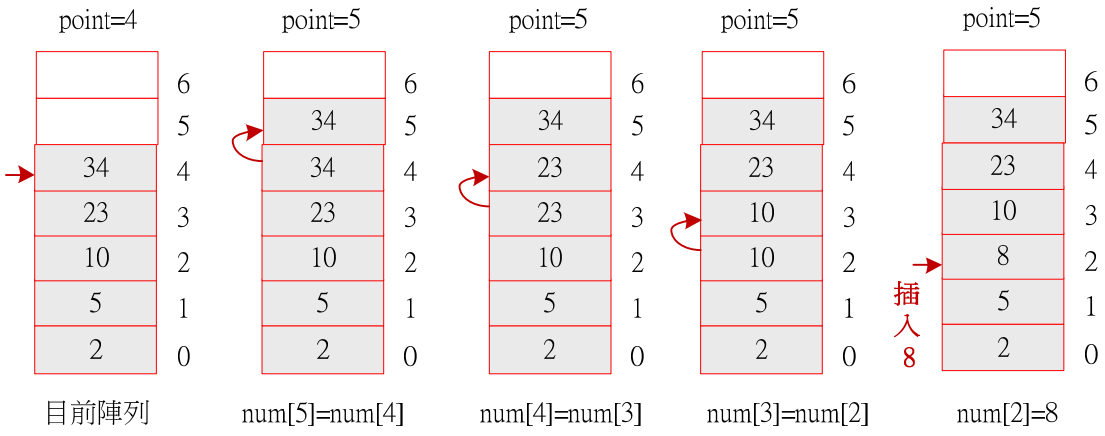

圖 4-9 有序陣列插入元素

插入的運作程序如圖 4-10 所示。首先將游標加一(移一位的意思)，並將目前由標位置存入 k，如果 k 沒有大於 0，表示是存入陣列第一筆資料。如果 k 大於 0，則比較數值是否大於欲插入的元素，如果大於的話，則往上移，否則就存入該空間。

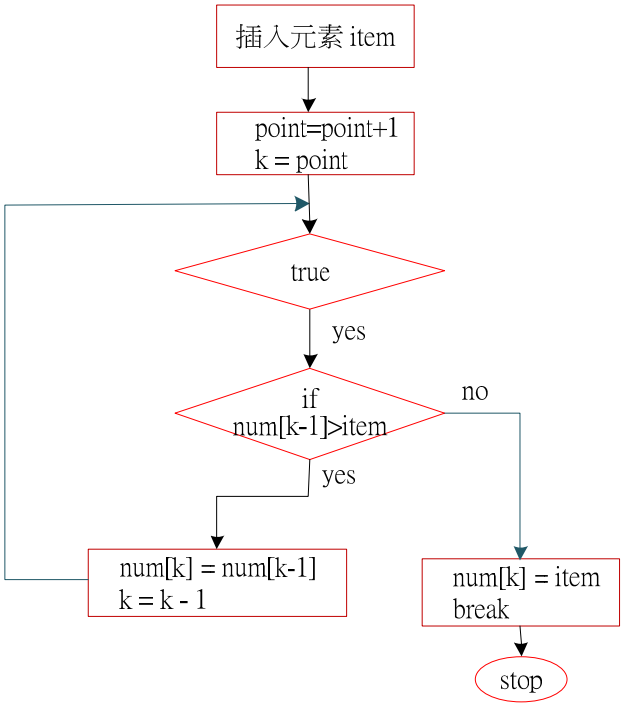

圖 4-10 有序陣列插入元素

(C)程式範例：Ex4_4.java

程式架構如圖 4-11 所示，在 Ex4_4.class 類別內包含 5 個元件，其中兩個類別變數，與 3 個方法程式。

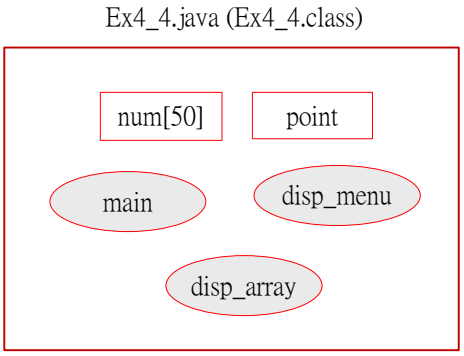

圖 4-11 Ex4_4 程式架構

```
01  //Ex4_4.java
```

```
02 /* 有序陣列的插入元素 */
03
04 import java.util.*;
05 public class Ex4_4{
06    static int num[] = new int[50];        // 宣告陣列空間
07    static int point;                      // 宣告游標變數
08    public static void main(String args[]) {
09        Scanner keyin = new Scanner(System.in);
10        point = -1;          // 游標初值
11        int select, k, item;
12        for (int i=0; i< 30; i++){          //給予陣列初值
13            num[i] = (i+1) *2;              //存入有序資料
14            point = point + 1;              //游標指示目前位置
15        }
16        disp_menu();
17        select = keyin.nextInt();
18        while(select != 3) {
19            switch(select) {
20                case 1:
21                    disp_array();
22                    break;
23                case 2:
24                    if (point >=50) {
25                        System.out.printf("陣列已滿無法插入!!\n");
26                    }else {
27                        System.out.printf("請輸入欲插入元素 =>");
28                        item = keyin.nextInt();
29                        point = point +1;
30                        k = point;
31                        while (true) {
32                            if(num[k-1] > item) {
33                                num[k] = num[k-1];
34                                k = k - 1;
35                            }
36                            else {
37                                break;
38                            }
39                            num[k] = item;
40                        }
41                    }
42                    break;
43                default:
44
45
46
47
```

```
                        System.out.printf("輸入錯誤 !! 請重新輸入\n");
                }
                disp_menu();
                select = keyin.nextInt();
            }

        }
// 列印功能表
    static void disp_menu() {
            System.out.printf("== 歡迎光臨 有序陣列管理系統 ==\n");
            System.out.printf("(1) 列印有序陣列內容\n");
            System.out.printf("(2) 插入陣列元素\n");
            System.out.printf("(3) 離開系統\n");
            System.out.printf("\t 請輸入工作選項 =>");
    }
// 列印陣列內容
   static void disp_array() {
        System.out.printf("\n== 目前序陣列有 %d 筆資料 ==\n", point+1);
        for(int i=0; i<=point; i++) {
                System.out.printf("%2d  ", num[i]);
                 if((i+1) % 10 == 0)
                     System.out.printf("\n");      //列印五筆, 換行
        }
        System.out.printf("\n");                  // 列印完畢, 換行
    }
}
```

4-3-4自我挑戰：有序陣列元素處理

(A)程式功能：PM4_2.java

吾人希望由 Ex4_4 中擴充可以刪除元素的功能，如下：

(1)具有顯示陣列內容、插入與刪除元素的選項，選單如下：

```
D:\Java2_book\chap4>javac PM4_2.java

D:\Java2_book\chap4>java PM4_2
== 歡迎光臨 有序陣列管理系統 ==
```

```
(1) 列印有序陣列內容
(2) 插入陣列元素
(3) 刪除陣列元素
(4) 離開系統
            請輸入工作選項 =>
```

(2)當選擇顯示陣列(選擇 1)，如下：

```
            請輸入工作選項 =>1

== 目前序陣列有 30 筆資料 ==
 2   4   6   8  10  12  14  16  18  20
22  24  26  28  30  32  34  36  38  40
41 44  46  48  50  52  54  56  58  60
```

(3)刪除元素(選擇 3)，如下： (刪除元素 14 後，再顯示陣列內容)

```
            請輸入工作選項 =>3
請輸入欲刪除元素 =>14
== 歡迎光臨 有序陣列管理系統 ==
(1) 列印有序陣列內容
(2) 插入陣列元素
(3) 刪除陣列元素
(4) 離開系統
            請輸入工作選項 =>1

== 目前序陣列有 29 筆資料 ==
 2   4   6   8  10  12  16  18  20  22
24  26  28  30  32  34  36  38  40  42
44 46  48  50  52  54  56  58  60
```

(B)系統分析

有序陣列刪除元素的運作程序幾乎與無序陣列相同，如圖 4-12 所示。首先找到欲刪除的元素，再將該元素後面的元素往前移一位 (num[i] = num[i+1])，再將游標減一(point = point -1) 即可。

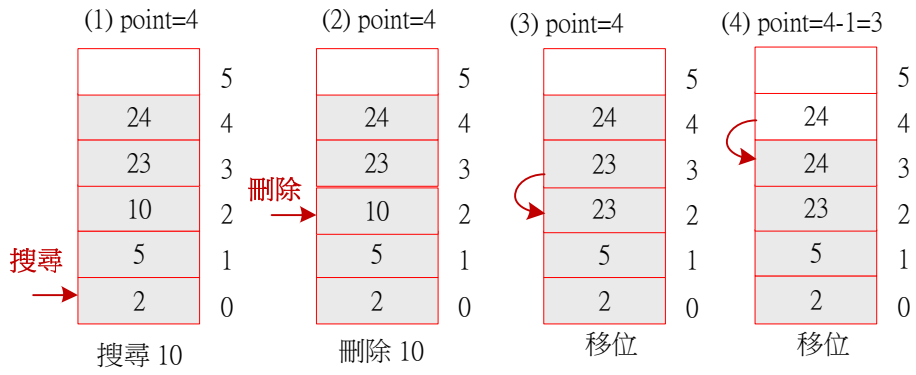

圖 4-12 有序陣列刪除元素的運作

(C)程式提示

圖 4-13 為 PM4_2.java 的程式架構，包含 2 個類別變數與 4 個方法程式，其中 Binary_search() 是讓我們快速找到欲刪除元素的位置，找到後，後面的元素往前移一位即可。

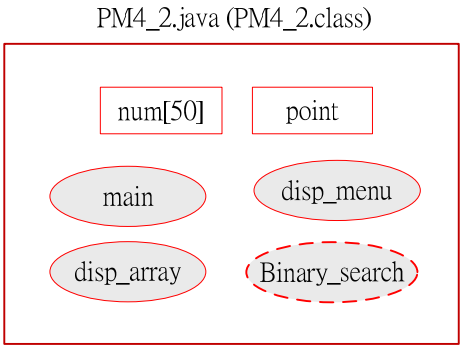

圖 4-13 PM4_2 程式架構

程式片段如下：

```
01  import java.util.*;
02  public class PM4_2{
03  ....
04          while(select != 4) {
05                  switch(select) {
06                      .....
07                      .....
08                       case 3:
09                              System.out.printf("請輸入欲刪除元素 =>");
10
```

```
                        item = keyin.nextInt();
                        int location = Binary_serach(item);
                        if (location == -1){
                                System.out.printf("陣列內沒有 %d 元素\n", item);
                        }else {
                            for(int i = location;i<=point;i++)
                                num[i] = num[i+1];
                        }
                        point = point - 1;
                        break;
                    default:
                            System.out.printf("輸入錯誤 !! 請重新輸入\n");
                }
                 .......
            }
.....
//二分搜尋法
    static int Binary_serach(int value){
        int location=-1;          // 設定找到位置
        ....
        請參考 Ex4_3 二分搜尋法
        ....
       return location;
    }
}
```

4-4 專題製作

4-4-1 範例研討：無序成績管理系統

(A)系統功能：Ex4_5.java

三民國中的張老師希望在電腦上管理班級同學(以學號登錄)的分數，班上課程有：國文、英文、數學、理化與自然，期望有下列功能：

(1) 具有顯示全班成績、插入同學成績、刪除、更新的功能，選單如下：

```
D:\Java2_book\chap4>javac Ex4_5.java

D:\Java2_book\chap4>java Ex4_5
== 三民國中 張老師成績管理系統
```

```
(1) 列印全班成績
(2) 新增學生成績
(3) 刪除學生成績
(4) 查詢學生成績
(5) 更新學生成績
(6) 依平均成績高低列印
(7) 離開系統
            請輸入工作選項 =>
```

(2)當選擇顯示全班成績(選擇 1)，如下：

```
== 列印全班各科成績==
學號    國文    英文    數學    理化    自然    平均
 1      36      70      34      61      93      0
 7      90      51      47      34      87      0
21      88      66      94      66      44      0
 5      37      47      69      45      73      0
28      82      90      60      31      74      0
29      97      41      97      95      39      0
13      63      30      72      79      63      0
10      32      52      51      96      80      0
15      72      64      35      72      46      0
 7       39      41      60      59      59      0
```

(3)新增學生成績(選擇 2)，如下：

```
            請輸入工作選項 =>2
請輸入學生學號(2 位數) =>45
請輸入國文成績 =>67
請輸入英文成績 =>90
請輸入數學成績 =>87
請輸入理化成績 =>95
請輸入自然成績 =>77
```

(4)刪除學生成績(選擇 3)，如下：

```
            請輸入工作選項 =>3
請輸入欲刪除學生學號 =>37
```

```
37 學生已刪除成功
目前學生人數為 8
```

(5)查詢學生成績(選擇 4)，如下：

```
請輸入工作選項 =>4
請輸入欲查詢學號 =>32
學號=32 國文=99 英文=93 數學=91 理化=61 自然=68 平均=0
```

(6)更新學生成績(選擇 5)如下：

```
請輸入工作選項 =>5
請輸入欲更新成績的學號 =>32
更新 32 學生的成績 =>
請更新國文成績(99) =>98
請更新英文成績(93) =>90
請更新數學成績(91) =>87
請更新理化成績(61) =>77
請更新自然成績(68) =>80
```

(7)列印成績排序(選擇 6)如下：

```
請輸入工作選項 =>6

== 列印全班各科成績==
學號    國文    英文    數學    理化    自然    平均
35      34      34      59      36      31      38
23      90      46      49      51      36      54
46      49      72      99      43      49      62
25      98      46      49      46      76      63
50      94      32      82      94      30      66
37      92      31      80      69      72      68
23      61      68      91      97      39      71
30      83      78      63      93      70      77
32      98      90      87      77      80      86
```

(B)系統分析

我們需要一個 50*7 的二維陣列來儲存，假設全班人數最高為 50 位，每位學生需 7 個欄位來存放學號、國文、英文、數學、理化、自然與平均分數。為了讓使用者方便測試系統的正確性，程式內先產生 10 位學生的資料，都是由亂數產生。另外，學號並沒有按照大小順序排列。

(C)程式範例

其程式架構如圖 4-14 所示，main() 方法內除了產生學生資料的初始值外，也提供各項功能的輸入選單。Disp_menu() 與 disp_course() 是顯示功能選單與學生成績資料顯示；Linear_search() 功能是搜尋到所欲處理資料的位置；Buffer_sort() 功能是依照平均分數排列。

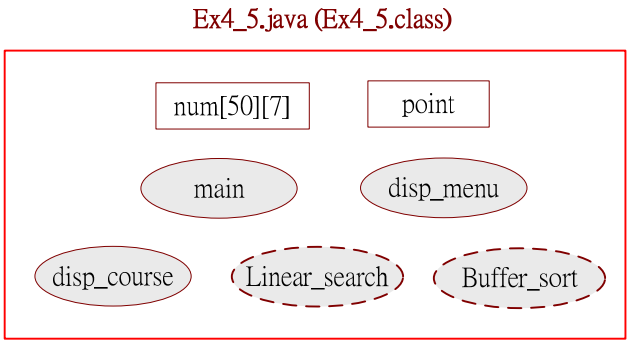

圖 4-14 Ex4_5 程式架構

```
01 //Ex4_5.java
02 /* 三民國中張老師成績管理系統 */
03
04
05 import java.util.*;
06 public class PM4_2{
07    static int course[][] = new int[50][7];        // 宣告無序陣列空間
08    //陣列欄位名稱
09
10    static String name[]={"學號","國文","英文","數學","理化","自然","平均"};
11    static int point;                              // 宣告游標變數
12
13    public static void main(String args[]) {
14          Scanner keyin = new Scanner(System.in);
15          Random ran = new Random();
16          int value;          // 輸入元素
17          int select;         // 功能選擇
18
19          point = -1;         // 游標初值(初值已存入 5 筆資料)
20          int location;
21
```

```
22          for (int i=0; i<10; i++){              //給予陣列初值
23                course[i][0] = 1 + ran.nextInt(50);
24                for(int j=1; j<=5; j++)
25                      course[i][j] = 30 + ran.nextInt(70);
26                point = point + 1;
27          }
28
29          disp_menu();
30          select = keyin.nextInt();
31          while(select != 7) {
32                switch(select) {
33                    case 1:
34                          disp_course();
35                          break;
36                    case 2:
37                          if (point >=50) {
38                              System.out.printf("陣列已滿無法插入!!\n");
39                          }else {
40                                point = point +1;
41                                System.out.printf("請輸入學生學號(2 位數)
42 =>");
43                                course[point][0]= keyin.nextInt();
44                                System.out.printf("請輸入國文成績  =>");
45                                course[point][1]= keyin.nextInt();
46                                System.out.printf("請輸入英文成績  =>");
47                                course[point][2]= keyin.nextInt();
48                                System.out.printf("請輸入數學成績  =>");
49                                course[point][3]= keyin.nextInt();
50                                System.out.printf("請輸入理化成績  =>");
51                                course[point][4]= keyin.nextInt();
52                                System.out.printf("請輸入自然成績  =>");
53                                course[point][5]= keyin.nextInt();
54                           }
55                           System.out.printf("目前學生人數為  %d\n", point);
56                           break;
57                    case 3:
58                          System.out.printf("請輸入欲刪除學生學號  =>");
59                          value = keyin.nextInt();
60                          location = Linear_serach(value);
61                          if (location == -1){
62                                System.out.printf("沒有學號= %d  學生\n", value);
63
64
65
66
67
```

```
                    }else {
                        for(int i = location;i<=point;i++)
                            course[i] = course[i+1];
                    }
                    point = point - 1;
                    System.out.printf("%d 學生已刪除成功\n", value);

                    System.out.printf("目前學生人數為 %d\n", point);

                    break;
                case 4:
                    System.out.printf("請輸入欲查詢學號 =>");

                    value = keyin.nextInt();
                    location = Line_serach(value);
                    if (location == -1){
                        System.out.printf("沒有學號= %d 學生\n", value);
                    }else {
                        for(int i=0;i<=6;i++){
                        System.out.printf("%s=%d  ", name[i], course[location][i]);
                        }
                        System.out.printf("\n");
                    }
                    break;
                case 5:
                    System.out.printf("請輸入欲更新成績的學號 =>");

                    value = keyin.nextInt();
                    location = Line_serach(value);
                    if (location == -1){
                        System.out.printf("沒有學號= %d 學生\n",
value);

                    }
                    else {
                    System.out.printf("更新 %d 學生的成績 =>\n", course[location][0]);

                    System.out.printf("請更新國文成績(%d) =>", course[location][1]);
                    course[location][1]= keyin.nextInt();
                    System.out.printf("請更新英文成績(%d) =>", course[location][2]);
                    course[location][2]= keyin.nextInt();
                    System.out.printf("請更新數學成績(%d) =>", course[location][3]);
                    course[location][3]= keyin.nextInt();
                    System.out.printf("請更新理化成績(%d) =>", course[location][4]);
                    course[location][4]= keyin.nextInt();
                    System.out.printf("請更新自然成績(%d) =>", course[location][5]);
                    course[location][5]= keyin.nextInt();
                    }
                    break;
```

```
114                 case 6:
115                     int sum;
116                     for(int i=0; i<=point; i++){
117                         sum = 0;
118                         for(int j=1; j<=5; j++)
119                             sum = sum + course[i][j];
120                         course[i][6] = sum/5;
121                     }
122                     buffer_sort();
123                     disp_course();
124                     break;
125                 default:
126                     System.out.printf("輸入錯誤 !! 請重新輸入\n");
127             }
128             disp_menu();
129             select = keyin.nextInt();
130         }
131     }
132     static void disp_menu() {
133         System.out.printf("== 三民國中 張老師成績管理系統 ==\n");
134         System.out.printf("(1) 列印全班成績\n");
135         System.out.printf("(2) 新增學生成績\n");
136         System.out.printf("(3) 刪除學生成績\n");
137         System.out.printf("(4) 查詢學生成績\n");
138         System.out.printf("(5) 更新學生成績\n");
139         System.out.printf("(6) 依平均成績高低列印\n");
140         System.out.printf("(7) 離開系統\n");
141         System.out.printf("\t 請輸入工作選項 =>");
142     }
143     static void disp_course() {     /* 列印全班各科成績 */
144         System.out.printf("\n== 列印全班各科成績==\n");
145         for(int i=0; i<name.length; i++)
146             System.out.printf("%s       ", name[i]);
147         System.out.printf("\n");
148         for(int i=0; i<=point; i++) {
149             for(int j=0; j<course[i].length; j++)
150                 System.out.printf("%2d          ", course[i][j]);
151             System.out.printf("\n");
152         }
153         System.out.printf("\n");                    // 列印完畢，換行
154
155
156
157
158
159
```

```
160     }
161     static int Linear_serach(int value){
162         int flag=0, i=0;
163         int location=-1;
164         while (i <= point) {
165             if(value == course[i][0]) {
166                 flag = 1;
167                 location = i;
168                 break;
169             }
170             i = i+1;
171         }
        if (flag == 1)
                return location;
        else
                return -1;
    }
    static void buffer_sort(){
        int temp[] = new int[7];
        for(int i=0; i<=point; i++){
            for(int j=0; j<=point; j++) {
                if(course[i][6] < course[j][6]){
                    temp = course[i];
                    course[i] = course[j];
                    course[j] = temp;
                }
            }
          }
      }
}
```

4-4-2 自我挑戰：有序成績管理系統

(A) 系統功能：PM4_3.java

請依照 Ex4_5.java 程式功能的資料結構改成有序陣列儲存，同樣的具有下列功能能：

```
D:\Java2_book\chap4>javac PM4_3.java

D:\Java2_book\chap4>java PM4_3
== 三民國中 張老師成績管理系統
(1) 列印全班成績
(2) 新增學生成績
(3) 刪除學生成績
```

```
(4) 查詢學生成績
(5) 更新學生成績
(6) 依平均成績高低列印
(7) 離開系統
        請輸入工作選項 =>
```

(B) 系統分析

將資料結構改成有序陣列，對於資料插入、刪除、搜尋、更新處理方面有稍微不同，這裡僅提示程式架構，如圖 4-15 所示。至於如何實現就留給讀者自挑戰看看。

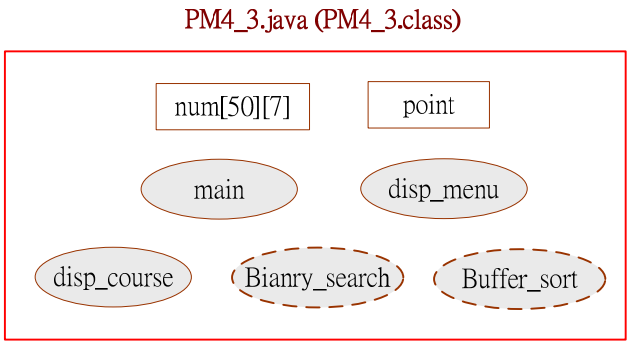

圖 4-15 PM4_3 程式架構

4-5 佇列資料結構

4-5-1 陣列佇列結構

佇列(Queue)在資訊系統裡時常用到，是屬於排列順序裝置，順序是『先進先出』(First-In-First-Out, FIFO)，表示先來的先出去的意思。圖 4-16 是 Queue 的儲存結構，它有 Front(前端)與 Rear(後端) 兩個口，Front 是資料的出口；Rear 是資料的入口，也就是資料由 Rear 進入，由 Front 出去。

圖 4-16 Queue 的運作程序

各種資料結構(如鏈路、樹狀、、等等)都可以用來實現佇列功能，但我們還是用最簡單的陣列來實現它。圖 4-17 是陣列佇列結構圖，我們取用一個陣列(假設空間為 50)，佇列出口(Front)固定在陣列 0 位置(queue[0])，佇列入口(Rear)採用游標方式，指示目前佇列可儲存的空閒位置。當資料插入佇列時，則存入 Rear 所指位置，並且將 Rear 內容加一，並判斷是否超載(大於 50)。如果由 Front 取出資料時，則所有資料往前移一位，到 Rear 所指位置為止，並將其內容減一。

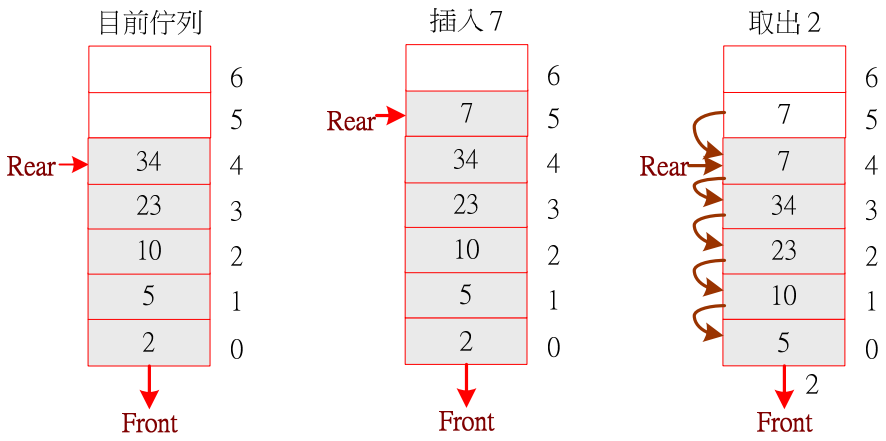

圖 4-17 陣列實現 Queue 結構

4-5-2 範例研討：醫院掛號系統

(A) 系統功能：Ex4_6.java

甄美麗美容中心需要一套掛號系統，功能是記錄客戶掛號的先後順序，功能如下：

(1) 具有顯示目前掛號客戶順序、登錄客戶掛號之功能，選單如下：

```
D:\Java2_book\chap4>javac Ex4_6.java

D:\Java2_book\chap4>java Ex4_6
== 歡迎光臨 甄美麗掛號系統 ==
(1) 列印目前掛號客戶名單
(2) 客戶掛號
(3) 離開系統
            請輸入工作選項 =>
```

(2) 當選擇登錄客戶掛號(選擇 2)，如下：

```
            請輸入工作選項 =>2
請輸入客戶姓名 =>張大有
張大有 已掛號成功!!
== 歡迎光臨 甄美麗掛號系統 ==
(1) 列印目前掛號客戶名單
(2) 客戶掛號
(3) 離開系統
            請輸入工作選項 =>2
請輸入客戶姓名 =>林旺
林旺 已掛號成功!!
```

(3) 選擇顯示目前掛號客戶名單(選擇 1)，如下：

```
            請輸入工作選項 =>1

== 目前有 2 位客戶掛號 ==
(1)張大有
(2)林旺
== 歡迎光臨 甄美麗掛號系統 ==
(1) 列印目前掛號客戶名單
```

(B) 系統分析

我們需建立一個 Queue 陣列來存放客戶掛號順序，Queue 是要儲存客戶名單，因此需要宣告程字串(String) 資料型態。

(C) 程式範例

圖 4-18 為其程式架構，其中 Queue[]、Front 與 Rear 等 3 個類別變數是佇列的主要裝置，再利用 4 個方法來實現系統功能。

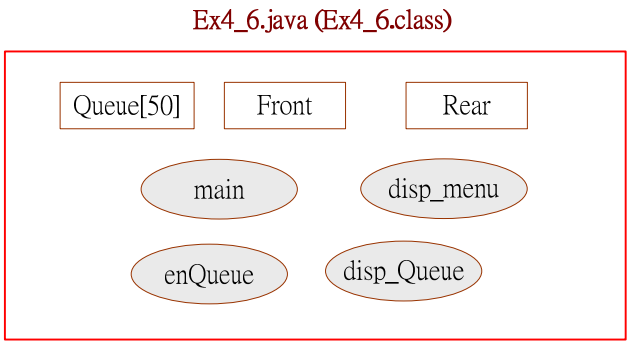

圖 4-18 Ex4_6 程式架構

```
01  //Ex4_6.java
02  /* 醫院掛號系統(僅插入 Queue 功能) */
03
04
05  import java.util.*;
06  public class Ex4_6{
07     static String Queue[] = new String[50];        // 宣告佇列空間
08     static int Rear;                               // 宣告佇列後端
09
10     static int Front=0;                            // 宣告佇列前端
11     public static void main(String args[]) {
12           Scanner keyin = new Scanner(System.in);
13           Rear = -1;                                // 佇列初值，表示佇列空間
14
15           String customer;
16           int select;
17           disp_menu();
18           select = keyin.nextInt();
19           while(select != 3) {
20                 switch(select) {
21                      case 1:
22                            disp_Queue();
23                            break;
24                      case 2:
25                            System.out.printf("請輸入客戶姓名 =>");
26                            customer = keyin.next();
27                            if (enQueue(customer))
28                                  System.out.printf("%s 已掛號成功!!\n", customer);
29                            else
30                                  System.out.printf("目前掛號已滿請稍候再掛!!\n");
31                             break;
32                      default:
33                           System.out.printf("輸入錯誤 !! 請重新輸入\n");
34                }
35
```

```
36                  disp_menu();
37                  select = keyin.nextInt();
38              }
39
40      }
41  // 列印功能表
42      static void disp_menu() {
43              System.out.printf("== 歡迎光臨 甄美麗掛號系統 ==\n");
44              System.out.printf("(1) 列印目前掛號客戶名單\n");
45              System.out.printf("(2) 客戶掛號\n");
46              System.out.printf("(3) 離開系統\n");
47              System.out.printf("\t 請輸入工作選項 =>");
48      }
49  // 列印 Queue 內容
50     static void disp_Queue() {
51          System.out.printf("\n== 目前有 %d 位客戶掛號 ==\n", Rear+1);
52          for(int i=0; i<= Rear; i++)
53                  System.out.printf("(%d)%s \n", i+1, Queue[i]);
54      }
55  // 加入 Queue 元素
56      static boolean enQueue(String customer){
57          if (Rear >= 50) {
58                  return false;
59          }else {
60                  Rear = Rear +1;
61                  Queue[Rear] = customer;
62                  return true;
63          }
64      }
    }
```

4-5-3 自我挑戰：醫生看診系統

(A) 系統功能：PM4_4.java

甄美麗美容中心除了客戶掛號功能外，還需要醫生看診系統，請您擴充 Ex4_6.java 系統的功能，使其具有醫生看診順序的功能。基本上，醫生看診是依照客人先來先看(Queue 功能)。當醫生選擇一位客戶時，則由掛號系統刪除該客戶排隊，其他客戶往前移一位，功能如下：

(1) 具有顯示目前掛號客戶順序、登錄客戶掛號之功能，選單如下：

```
D:\Java2_book\chap4>javac PM4_4.java

D:\Java2_book\chap4>java PM4_4
== 歡迎光臨 甄美麗掛號系統 ==
(1) 列印目前掛號客戶名單
(2) 客戶掛號
(3) 醫生看診客戶
(4) 離開系統
            請輸入工作選項 =>
```

(2) 當選擇登錄客戶掛號，並顯示已掛號客戶(選擇 2 與 1)，如下：

```
            請輸入工作選項 =>2
請輸入客戶姓名 =>劉真立
劉真立 已掛號成功!!
== 歡迎光臨 甄美麗掛號系統 ==
(1) 列印目前掛號客戶名單
(2) 客戶掛號
(3) 醫生看診客戶
(4) 離開系統
            請輸入工作選項 =>2
請輸入客戶姓名 =>張真真
張真真 已掛號成功!!
== 歡迎光臨 甄美麗掛號系統 ==
(1) 列印目前掛號客戶名單
(2) 客戶掛號
(3) 醫生看診客戶
(4) 離開系統
            請輸入工作選項 =>1

== 目前有 2 位客戶掛號 ==
(1)劉真立
```

```
(2)張真真
```

(3) 選擇醫生看診客戶姓名並觀察客戶掛號名單(選擇 3 與 1)，如下：

```
            請輸入工作選項 =>3
請 劉真立 先生/小姐進入看診室
== 歡迎光臨 甄美麗掛號系統 ==
(1) 列印目前掛號客戶名單
(2) 客戶掛號
(3) 醫生看診客戶
(4) 離開系統
            請輸入工作選項 =>1

== 目前有 1 位客戶掛號 ==
(1)張真真
```

(B) 系統分析

醫生叫診是依照先進先出的規則，每叫一個客戶則表示由 Queue 內刪除一個元素，因此，僅依照 Ex4_6 擴充刪除佇列元素的功能即可。

(C) 程式提示

圖 4-19 為其程式架構，增加了 emptyQueue() 與 deQueue() 兩個方法，前者是測試 Queue 是否已空閒，表示有沒有掛號中的客戶；後者是醫生叫號後，刪除前面的客戶。

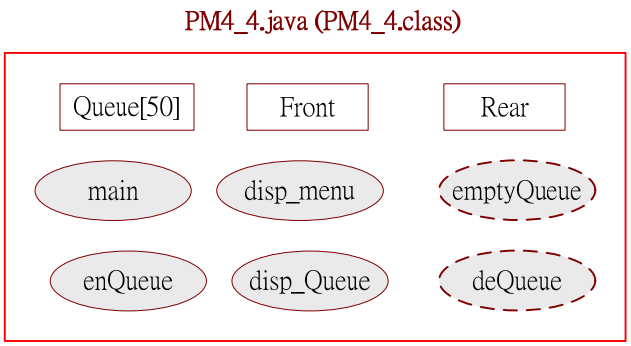

圖 4-19 PM4_4 程式架構

程式提示如下：

```
01 //PM4_4.java
02 /* 醫院掛號系統(含客戶掛號與醫生看診順序的功能) */
03
04
05 import java.util.*;
06 public class PM4_4{
07    .......
08         while(select != 4) {
09              switch(select) {
10
11            ..................
12                   case 3:
13                        if (emptyQueue())
14                             System.out.printf("目前沒有客戶掛號\n");
15                        else{
16                             customer = deQueue();
17                             System.out.printf("請 %s 先生/小姐進入看診室\n",
18 customer);
19                        }
20                        break;
21                   ........
22         }
23
24     }
25 // 列印功能表
26     static void disp_menu() {
27     .......
28     }
29 // 列印 Queue 內容
30    static void disp_Queue() {
31         ........
32     }
33 // 加入 Queue 元素
34     static boolean enQueue(String customer){
35         ........
36     }
37 // 探索 Queue 是否空間
38     static boolean emptyQueue(){
39         if (Rear < 0)
40              return true;
41         else
42              return false;
43     }
44 // 刪除 Queue 元素
45
46
```

```
47      static String deQueue() {
48              String customer = Queue[Front];
49              for (int i=Front; i< Rear; i++)
50                      Queue[i] = Queue[i+1];
51              Rear = Rear - 1;
                return customer;
        }
}
```

4-6 堆疊資料結構

4-6-1 陣列堆疊結構

堆疊(Stack)在資訊系統裡時常用到，與 Queue 一樣都是屬於排列順序裝置，但他的順序是『先進後出』(First-In-Last-Out, FILO)，表示先來的後出去的意思。圖 4-20 是 Stack 的儲存結構，他只有一個出入口，資料進出都由這個口，地多利用 Top 變數來指示目前出入口位置。另外利用 push() 方法來操作將資料推入堆疊內(Top = Top + 1)，與 pop() 方法將資料由堆疊內擠出(Top = Top – 1)。

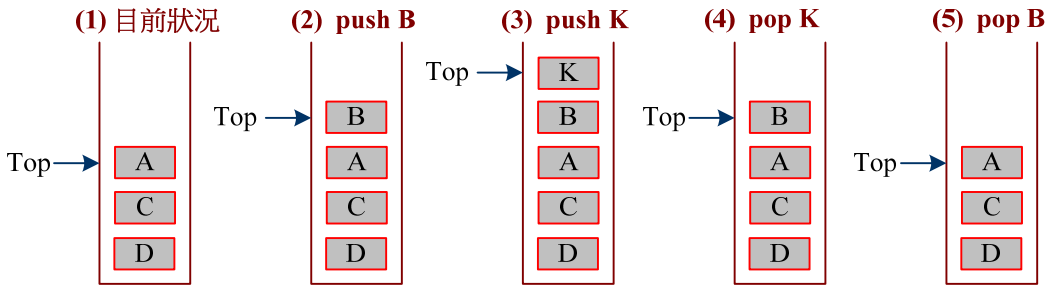

圖 4-20 Stack 的運作程序

4-6-2 範例研討：走迷宮演練

(A) 程式功能：Ex4_7.java

談到堆疊(Stack)大多人都會想到走迷宮的問題，但其實它應用非常廣泛，譬如導航系統一定都會用到。吾人利用堆疊記錄過去走的路徑，當須退回原處時，再依照堆疊內記錄退回原處，就不會迷路了。

圖 4-21 是迷宮地圖，吾人將地圖中每一個節點都用座標表示，橫座標由 A ~ Q，縱座標由 0 ~9，位置由 A0 ~ Q9，並所有路徑(或位置)都可以通過。假設，我們由 D9 開始，經過圖中節點到達 Q0，回程由 Q0 是否可以回到 D9。

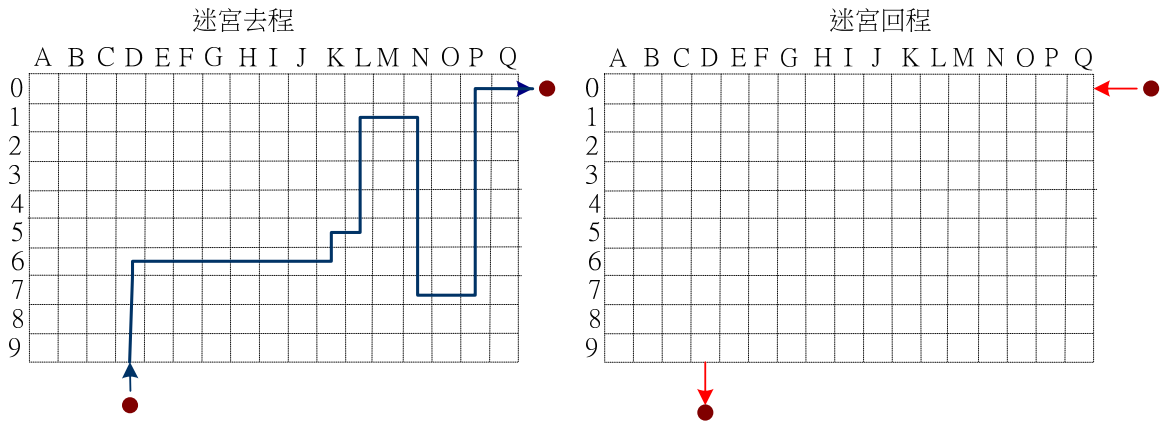

圖 4-21 迷宮走路演練

請編寫一只程式依照圖中路徑行走，完畢後再顯示所經過的節點如何，接著再依照走過的記錄是否可以回到原點，驗證走迷宮的初步構想。期望功能如下：

(1) 期望具有以下 4 種功能，選單如下：

```
D:\Java2_book\chap4>javac Ex4_7.java

D:\Java2_book\chap4>java Ex4_7
== 歡迎光臨 走迷宮演練 ==
(1) 列印以走過的路線
(2) 迷宮去程開始
(3) 迷宮回程開始
(4) 離開系統
            請輸入工作選項 =>
```

(2) 當選擇迷宮去程(選擇 2 與 1)，則出現走過路線如下：

```
            請輸入工作選項 =>2
D0 ==> D9 ==> D8 ==> D7 ==> E7 ==>
F7 ==> G7 ==> H7 ==> I7 ==> J7 ==>
K7 ==> K6 ==> L6 ==> L5 ==> L4 ==>
L3 ==> L2 ==> M2 ==> N2 ==> N3 ==>
N4 ==> N5 ==> N6 ==> N7 ==> N8 ==>
```

```
O8 ==> P8 ==> P7 ==> P6 ==> P5 ==>
P4 ==> P3 ==> P2 ==> P1 ==> Q1 ==>
總共走了 35 路徑
```

(3) 選擇查閱過去走的路線(選擇 1)，如下：

```
            請輸入工作選項 =>1


== 到目前經過 35 個路徑 ==
(1)D0   (2)D9   (3)D8   (4)D7   (5)E7
(6)F7   (7)G7   (8)H7   (9)I7   (10)J7
(11)K7   (12)K6   (13)L6   (14)L5   (15)L4
(16)L3   (17)L2   (18)M2   (19)N2   (20)N3
(21)N4   (22)N5   (23)N6   (24)N7   (25)N8
(26)O8   (27)P8   (28)P7   (29)P6   (30)P5
(31)P4   (32)P3   (33)P2   (34)P1   (35)Q1
```

(4) 當選擇迷宮回程(選擇 3)，則會依照之前走過路徑回到原地，如下：

```
            請輸入工作選項 =>3
Q1 ==> P1 ==> P2 ==> P3 ==> P4 ==>
P5 ==> P6 ==> P7 ==> P8 ==> O8 ==>
N8 ==> N7 ==> N6 ==> N5 ==> N4 ==>
N3 ==> N2 ==> M2 ==> L2 ==> L3 ==>
L4 ==> L5 ==> L6 ==> K6 ==> K7 ==>
J7 ==> I7 ==> H7 ==> G7 ==> F7 ==>
E7 ==> D7 ==> D8 ==> D9 ==> D0 ==>
回程路徑已結束
```

(B) 系統分析

首先我們宣告陣列 path[]，並依照圖 4-21(a) 路徑位置填入該陣列。前進時，由 path[] 中讀取下一個路徑位置，並將該位置推入(Push)堆疊內，一直到讀取完畢，表示已走完所有路徑，堆疊內也記錄所有走過的節點。回程時，再一個接一個位置由堆疊內擠出(Pop)，依照擠出位置行走，就可以回到原點。

(C) 程式範例

圖 4-22 為其程式架構，我們將 Push 與 Pop 製作成獨立的方法，可利用他們對 Stack 做推入與擠出的操作。

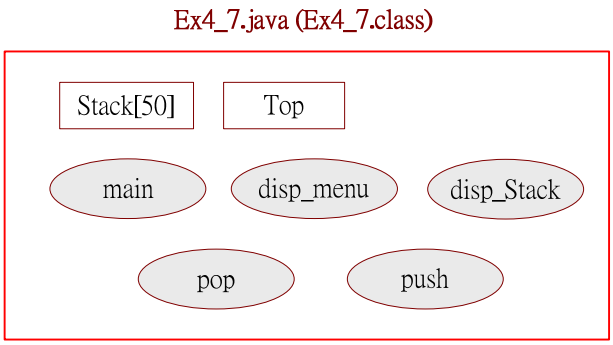

圖 4-22 Ex4_7 程式架構

```
01 //Ex4_7.java
02 /* 走迷宮演練(驗證 Stack 功能) */
03
04
05 import java.util.*;
06 public class Ex4_7{
07    static String Stack[] = new String[50];        // 宣告佇列空間
08    static int Top;                                // 宣告佇列後端
09
10    public static void main(String args[]) {
11          Scanner keyin = new Scanner(System.in);
12          Top = -1;                                 // 佇列初值，表示佇列空間
13          String step;
14          String path[] = {"D0", "D9", "D8", "D7", "E7",
15                           "F7", "G7", "H7", "I7", "J7",
16                           "K7", "K6", "L6", "L5", "L4",
17                           "L3", "L2", "M2", "N2", "N3",
18                           "N4", "N5", "N6", "N7", "N8",
19                           "O8", "P8", "P7", "P6", "P5",
20                           "P4", "P3", "P2", "P1", "Q1"};
21          int select;
22          disp_menu();
23          select = keyin.nextInt();
24          while(select != 4) {
25                switch(select) {
26                    case 1:
27                          disp_Stack();
28                          break;
29                    case 2:
30                          for (int i=0; i<path.length; i++){
31                              step = path[i];
32                              if (push(step))
33                                    System.out.printf("%s ==> ", step);
34                              else {
35                                    System.out.printf("目前路徑已滿請回頭!!\n");
```

```
36                                  break;
37                              }
38                              if ((i+1) %5 == 0)
39                                      System.out.printf("\n");
40                          }
41                          System.out.printf("總共走了 %d 路徑\n", Top+1);
42                          break;
43                  case 3:
44                      int k = 0;
45                      while (Top >= 0) {
46                          step = pop();
47                          System.out.printf("%s ==> ", step);
48                          k = k + 1;
49                          if(k%5 == 0)
50                                  System.out.printf("\n");
51                      }
52                      System.out.printf("回程路徑已結束\n");
53                      break;
54                  default:
55                      System.out.printf("輸入錯誤 !! 請重新輸入\n");
56              }
57              disp_menu();
58              select = keyin.nextInt();
59          }
60
61      }
62  // 列印功能表
63      static void disp_menu() {
64          System.out.printf("== 歡迎光臨 走迷宮演練 ==\n");
65          System.out.printf("(1) 列印以走過的路線\n");
66          System.out.printf("(2) 迷宮去程開始\n");
67          System.out.printf("(3) 迷宮回程開始\n");
68          System.out.printf("(4) 離開系統\n");
69          System.out.printf("\t 請輸入工作選項 =>");
70      }
71  // 列印 Stack 內容
72      static void disp_Stack() {
73          System.out.printf("\n== 到目前經過 %d 個路徑 ==\n", Top+1);
74          for(int i=0; i<= Top; i++){
75                  System.out.printf("(%d)%s    ", i+1, Stack[i]);
```

```
82                  if ((i+1) %5 == 0)
83                          System.out.printf("\n");
84          }
85      }
86  // 加入 Push 元素
87      static boolean push(String step){
88          if (Top >= 50) {
89                  return false;
90          }else {
91                  Top = Top +1;
92                  Stack[Top] = step;
93                  return true;
94          }
95      }
        static String pop(){
            String step = Stack[Top];
            Top = Top - 1;
            return step;
        }
    }
```

4-6-3 自我挑戰：迷宮闖關遊戲

(A) 程式功能：PM4_5.java

圖 4-22 是一張迷宮地圖，請編寫一只程式，使其能由 D9 進入後，由 Q3 出去，表示闖關成功，再顯示所有經過的路徑為何。

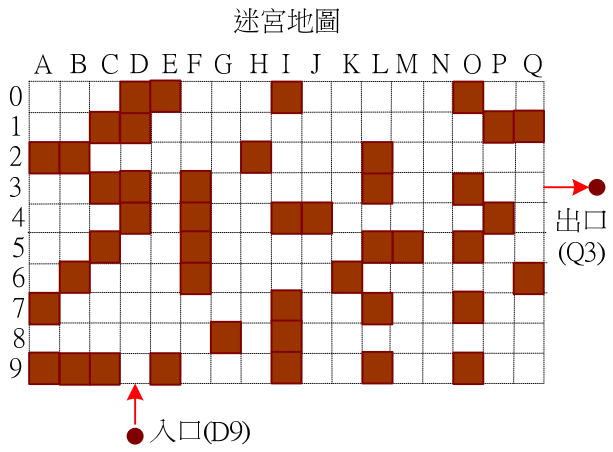

圖 4-23 迷宮地圖

(B) 系統分析

走迷宮最基本必須知道目前在哪一節點，可以往哪一個下一個節點走，以 G6 節點為例，下一個節點可以是 {G5, H6, G4, F6}，亦是“(“G”± 1)(6 ± 1)。

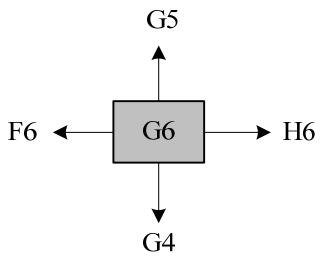

圖 4-23 節點的下一個節點

接著，我們必須知道哪些節點是可以通過的，我們用 path[] 來記錄可以經過的節點如下：(將迷宮地圖數位化)

```
Path[] = {A0, A1, A3, A4, A5, A6, A8, B0, B1, B3, B4, B5, B7, B8, C0,
          C2, C4, C6, C7, C8, D2, D5, D6, D7, D8, D9, E1, E2, E3, E4,
          E5, E6, E7, E8, F0, F1, F2, F7, F8, F9, G0, G1, G2, G3, G4,
          G5, G6, G7, G9, H0, H1, H3, H4, H5, H6, H7, H8, H9, I1, I2,
          I3, I5, I6, J0, J1, J2, J3, J5, J6, J7, J8, J9, K0, K1, K2, K3, K4,
          K5, K7, K8, K9, L0, L1, L4, L6, L8, M0, M1, M2, M3, M4,
          M6, M7, M8, M9, N0, N1, N2, N3, N4, N5, N6, N7, N8, N9
          O1, O2, O4, O6, O8, P0, P2, P3, P5, P6, P7, P8, P9, Q0, Q2,
          Q3, Q4, Q5, Q7, Q8, Q9}
```

當我到達某一節點後，計算出下一個路徑節點為何，再搜尋是否在 path[] 陣列內(扣除剛經過的節點)，如果有則往下一個節點走，如果都沒有的話，則必須再退回上一個節點，再搜尋可以通過的節點。

僅提示到此，接著讓同學搜尋(Google 一下甚麼都有)其他方法實現它。