

第五章

陣列及在資料結構上的應用



5-4 陣列在「搜尋」上的實務應用



【定義】是指在一群資料中找尋所要的特定資料。當資料量很龐大時，如何快速搜尋到資料是本章所要探討的重點。

【分類】若依資料量大小來區分，搜尋可分兩類：

- 1.內部搜尋：例如「二分搜尋法」。
- 2.外部搜尋：例如「循序搜尋法」。

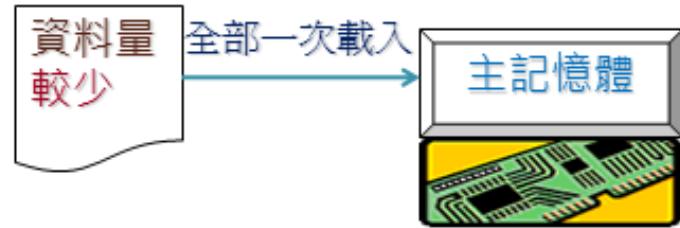
一、內部搜尋

【定義】

是指較少資料量要進行「搜尋」時，可以全部一次載入到「主記憶體」中，進行「搜尋」動作。由於，主記憶體是內嵌到電腦主機版中，因此，此種「搜尋」就稱為「內部搜尋法」。

【適用時機】資料量較少者。

【圖解說明】



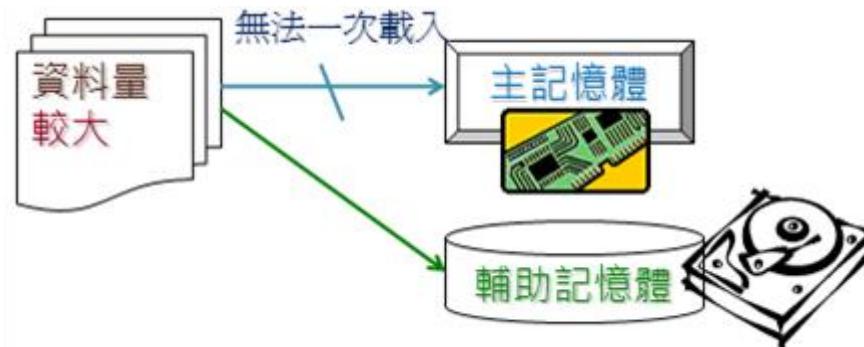
二、外部搜尋

【定義】

是指較大資料量要進行「搜尋」時，由於檔案太大，使得要「搜尋」的資料無法一次全部載入到主記憶體中。因此，我們在進行「搜尋」時，必須要借助輔助記憶體來分批處理。由於，輔助記憶體「不是」內嵌在電腦主機版中，因此，此種搜尋就稱為「外部搜尋」。

【適用時機】資料量較大者。

【圖解說明】



5-4.1 循序搜尋法(Sequential Search)

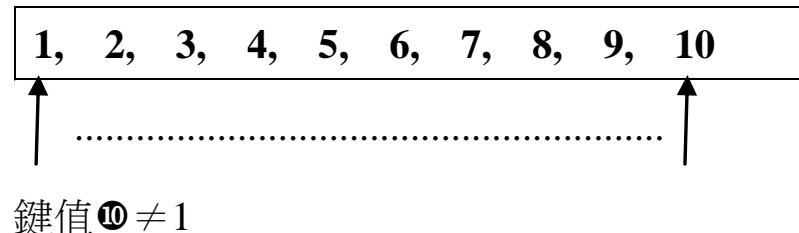


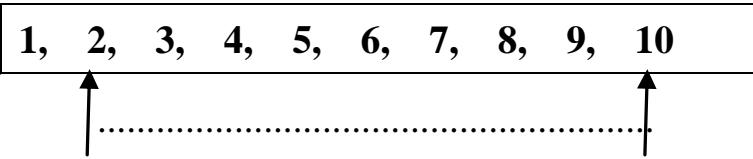
在日常生活中，我們常常往往想從一群資料中找尋所要的特定資料，而在資料結構中最普遍也最簡單的應該就是「循序搜尋法」。

【定義】

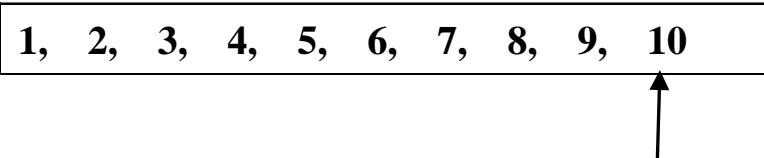
又稱為線性搜尋(Linear Search)，它是指從第一個資料項開始依序取出與「鍵值Key」相互比較，直到找出所要的元素或所有資料均已找完為止。

【圖解說明】





鍵值⑩≠2



鍵值⑩=10(成功找到了！)

【優點】1. 程式容易撰寫。

2. 資料不須事先排序。

【缺點】搜尋效率較差(平均次數= $\frac{N+1}{2}$)，因為不管資料順序為何，每次都必須要從頭到尾拜訪一次。

【演算法】

演算法：循序搜尋法

```
01 Procedure sequential_search(int list[], int n, int key)
02 Begin
03     for (i = 0; i < n; i++)    //從頭到尾拜訪一次
04         if (list[i] == key)    //比對陣列內的資料是否等於欲搜尋的條件
05             return i+1;        //若找到符合條件的資料，就傳回其索引
06         return(-1);          //若找不到符合條件的資料，就傳回 -1
07     End
08 End Procedure
```

【演算法】

list[0]	list[1]	list[2]	list[3]	list[4]	list[n-1]
90	80	40	50	65	77

→

欲找鍵值key=50

A[3]=key ∴在第四個位置找到鍵值key

【實作】請設計一個循序搜尋程式來找尋下列資料列，並顯示「1」資料項在資料列的所在位置。

輸入資料：9,8,4,5,6,7,1,2

輸出結果：1在陣列中的第7筆

行號	程式檔名：ch5-4-1.java
01	<code>public class ch5_4_1 {</code>
02	<code> static int SearchValue=1;</code>
03	<code> static int NumID;</code>
04	<code> public static void main(String[] args)</code>
05	<code> { //宣告及初值設定</code>
06	<code> int NumArray[]={9,8,4,5,6,7,1,2};</code>
07	<code> //循序搜尋程式</code>
08	<code> for(int i=0;i<NumArray.length;i++)</code>
09	<code> {</code>
10	<code> if(NumArray[i]==SearchValue)</code>
11	<code> NumID=i+1;</code>
12	<code> }</code>
13	<code> System.out.println("Your Numbers:");</code>
14	<code> for(int i=0;i<NumArray.length;i++)</code>
15	<code> {</code>
16	<code> System.out.print(" "+ NumArray[i]);</code>
17	<code> }</code>
18	<code> System.out.println();</code>
19	<code> System.out.print("1 in array of "+ NumID);</code>
20	<code> }</code>
21	真理大學 資訊工程學系

【執行結果】

```
Your Numbers:  
9 8 4 5 6 7 1 2  
1 in array of 7
```

【說明】

行號02：定義SearchValue常數為1，亦即要搜尋的資料。

行號03：宣告NumID為整數變數，NumID變數是用來記錄要找的資料，在陣列中的索引位置。

行號06：宣告NumArray為陣列變數，用來顯示「原始資料」的陣列，並且設定初值串列。

行號08~12：進行循序搜尋法，從第一個資料項開始依序取出與「鍵值Key」相互比較。

行號11：如果有找到時，則記錄它在陣列索引位置+1。

行號14~17：用來顯示「原始資料」。

行號19：顯示要找的資料，在陣列中的位置。

【實作2】先產生10個亂數值，其範圍為10~100之間，再利用循序搜尋法來找資料項的位置。

A screenshot of a Windows Command Prompt window titled "命令提示字元". The command "C:\Java\ch09>java ch9_2a" is run, followed by the output of 10 random integers between 10 and 100. Below this, the text "=====循序搜尋結果=====" is displayed, and then "28在陣列中的第1筆." indicating the search result.

```
C:\Java\ch09>java ch9_2a
產生10個亂數值：
28 85 70 50 70 17 93 76 38 22
=====循序搜尋結果=====
28在陣列中的第1筆.
```

【解答】

題目：循序搜尋程式 2	程式檔案名稱	ch5_4_1A.java
01 public class ch5_4_1A 02 { 03 public static void main(String[] args) 04 { 05 int list[]= new int [10]; 06 int i,n,c,key; 07 System.out.println("產生10個亂數值："); 08 for (i=0;i<10;i++) 09 { 10 list[i] = (int) (Math.random() * 90) + 10 ; //產生10~100的整數亂數值 11 System.out.print(list[i] + " "); 12 }		

```
13 System.out.println();
14 System.out.println("=====循序搜尋結果=====");
15 System.out.println("28在陣列中的第" + sequential_search(list, 10, 28) + "筆.");
16 }
17
18 public static int sequential_search(int list[], int n, int key)
19 {
20     int i;
21     for (i = 0; i <n; i++)
22         if (list[i] == key) //比對陣列內的資料是否等於欲搜尋的條件
23             return i+1; //若找到符合條件的資料，就傳回其索引
24         return(-1); //若找不到符合條件的資料，就傳回 -1
25     }
26 }
```

【執行結果】

找不到符合條件的資料
產生10個亂數值： 76 63 88 35 84 68 75 99 80 62 =====循序搜尋結果===== 28在陣列中的第-1筆.
找到符合條件的資料
產生10個亂數值： 42 71 38 88 61 53 28 77 63 69 =====循序搜尋結果===== 28在陣列中的第7筆.

5-4.2 二分搜尋法(Binary Search)



【定義】

是指先將資料排序之後，再分割成兩部份，並利用「鍵值」與「中間值」來比較大小，如果「鍵值」大於「中間值」，則可確定要找的資料在後半段的元素，如此分割數次直到找到為止。

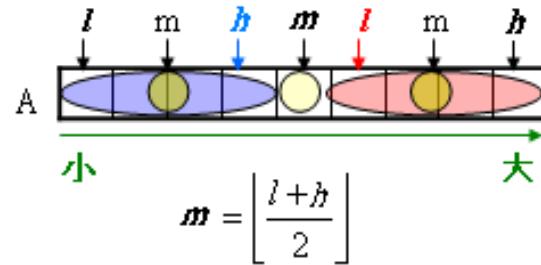
【適用時機】事先已經排序完成。

【優點】搜尋效率較佳(平均次數= $(\lfloor \log_2 N \rfloor + 2)/2$)。

【缺點】1. 資料必須事先排序。

2. 檔案必須是直接存取檔或隨機檔。

【示意圖】



【舉例】

假設現在有6筆紀錄，分別為：4,5,1,2,3,7,6，並且鍵值為6，請追蹤每一回合的比較過程。

【圖解說明】

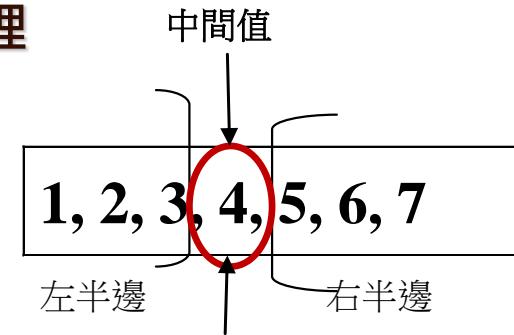
步驟一：排序後

1, 2, 3, 4, 5, 6, 7

步驟二：第一次比較

「鍵值6」與「中間值4」做比較，結果「鍵值6>中間值4」，

則中間值的左半邊資料必須忽略不理

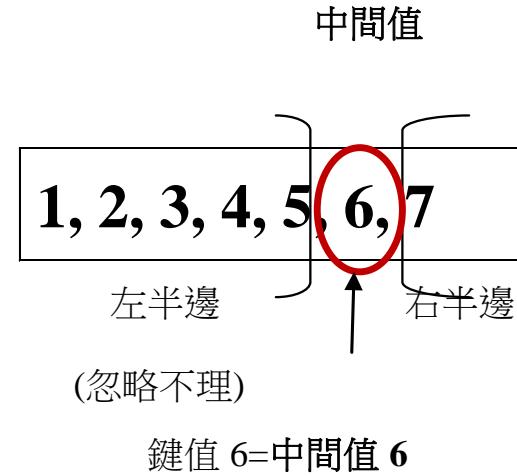


(忽略不理)

鍵值 6>中間值 4

步驟三：第二次比較

「鍵值6」與「中間值6」做比較，結果「鍵值6=中間值6」，則表示成功找到了。否則，重複以上的步驟，直到找到為止。



【演算法】

演算法：二分搜尋法

```
01 Procedure binsearch(A[], key , low, high)
02 Begin
03     if(low<=high)
04     {
05         mid= $\lfloor (low + high) / 2 \rfloor$ ;      //計算中間值的位置
06         switch compare(key,A[mid])    //「鍵值」與「中間值」比較大小
07         {
08             Case "=": return mid;    //找到
09             Case "<": return binsearch(A, key, low, mid-1); //找左半部
10             Case ">": return binsearch(A, key, mid+1, high); //找右半部
11         }
12     }
13     Return -1;      //搜尋失敗，傳回-1
14 End
15 End Procedure
```

【實作】

題目：二分法搜尋	程式檔案名稱	ch5_4_2.java
01 import java.util.Scanner; 02 public class ch5_4_2 03 { 04 public static void main(String[] args) 05 { 06 int Temp[]={1,8,15,24,33,45,76,88,99}; 07 int Key; 08 int Low, High, Middle, Searchtime; 09 System.out.println("排序後的數列為：1,8,15,24,33,45,76,88,99"); 10 Scanner indata= new Scanner(System.in); 11 System.out.print("請輸入以上的任一組數字："); 12 String str=indata.next(); 13 Key=Integer.parseInt(str); 14 Low = 0; //設定第一個字母 15 High = 9; //設定最後一個字母 16 Searchtime = 0; //搜尋次數初值設定為 17 Middle = (int)((Low + High)/2); //搜尋中間值 18 do 19 { 20 Searchtime = Searchtime + 1;		



```
21     if (Temp[Middle] == Key)           //找到資料
22     { //顯示資料位置
23         System.out.print("該數字是排在第" + Middle + "個順位");
24         //顯示搜尋次數
25         System.out.print("一共搜尋" + Searchtime + " 次");
26         break;                         //跳出迴圈
27     }
28     else if(Temp[Middle] < Key)
29         Low = Middle + 1;             //改變左半部
30     else
31         High = Middle - 1;          //改變右半部
32         Middle = (int)((Low + High) / 2); //改變中間值
33     }
34     while(Low <= High);
35     System.out.println();
36 }
37 }
```

【執行結果】

排序後的數列為：1,8,15,24,33,45,76,88,99

請輸入以上的任一組數字：88

該數字是排在第7個順位一共搜尋2 次

【作法】

在二分搜尋法中，從數列的「中間」開始搜尋，如果這個數小於我們所搜尋的數，則該數左邊的數一定都小於要搜尋的對象，所以無須浪費時間在左邊的數列；如果數列中間的數大於所搜尋的對象，則右邊的數無須再搜尋，直接搜尋左邊的數列。其步驟如下：

步驟① 將所有資料由小至大排序。

步驟② 設L(Low)表示第一項資料(最小)的索引，H(High)表示最後一項資料(最大)的索引。

步驟③ M(Middle)表示中間項的索引 = $\lfloor (L+H)/2 \rfloor$

步驟④ 將「鍵值」和「中間值」做比較。

當鍵值 > 中間值，則 $L = M + 1$ 。

當鍵值 < 中間值，則 $H = M - 1$ 。

當鍵值 = 中間值，則表示已經找到。

步驟⑤ 重新計算M(中間值之索引) · 重複步驟③和⑤ · 直到找到或所有資料均找過為止。

【實例】

假設我們現在有9筆資料 · 分別是：8,1,99,76,88,45,15,33,24 · 接下來 · 請利用「二分搜尋法」來搜尋出使用者欲找的鍵值88。

【解答】

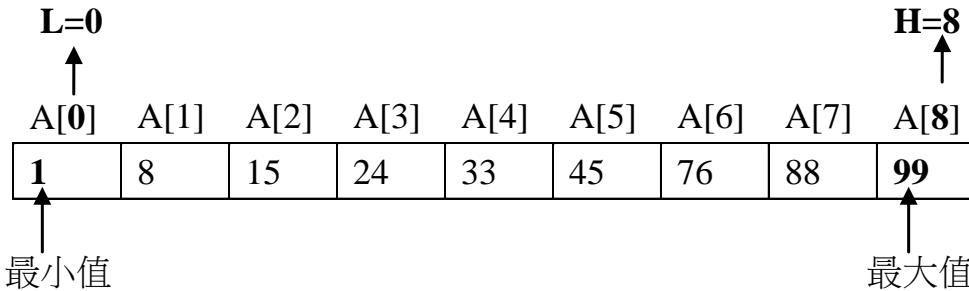
前置作業：依序將9筆資料放到A[0]~A[8]的陣列中

A[0]	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]	A[8]
8	1	99	76	88	45	15	33	24

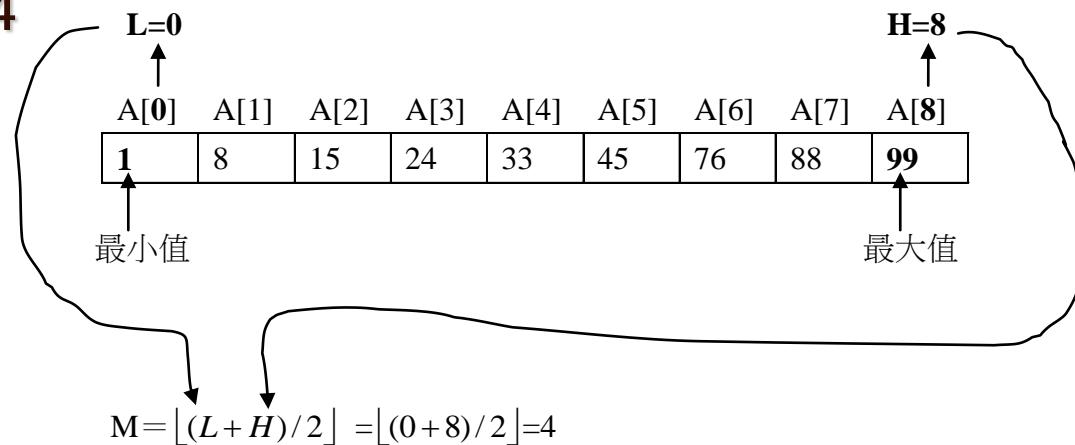
步驟①將所有資料由小至大排序

A[0]	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]	A[8]
1	8	15	24	33	45	76	88	99

步驟② 假設L(Low)表示第一項資料(亦即最小值的索引)，而H(High)表示最後一項資料(亦即最大值的索引)。因此，我們就可以順利求得最小值1的索引就是0，而最大值99的索引就是8。



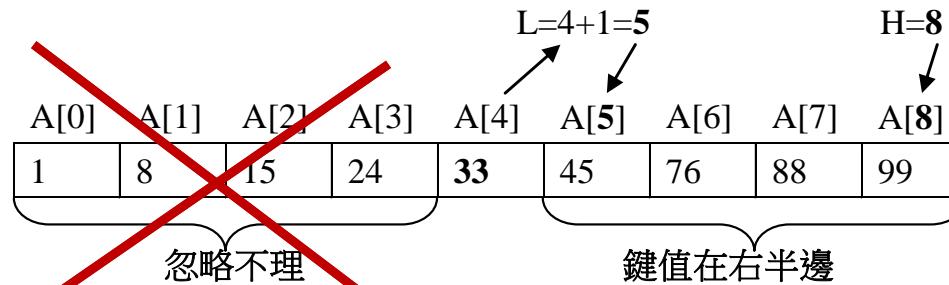
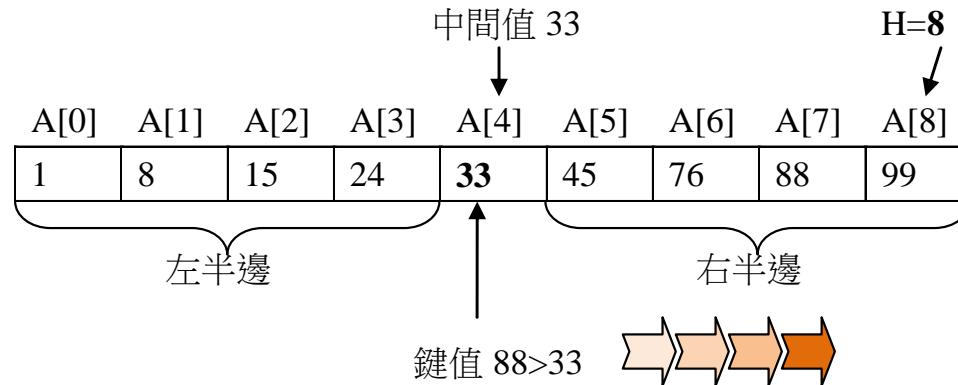
步驟③ 假設M表示中間值的索引，其計算公式就是L加H括號除以2再取下限，因此，我們就可以將L=0,H=8的值代入到公式中，就可以順利求出M=4



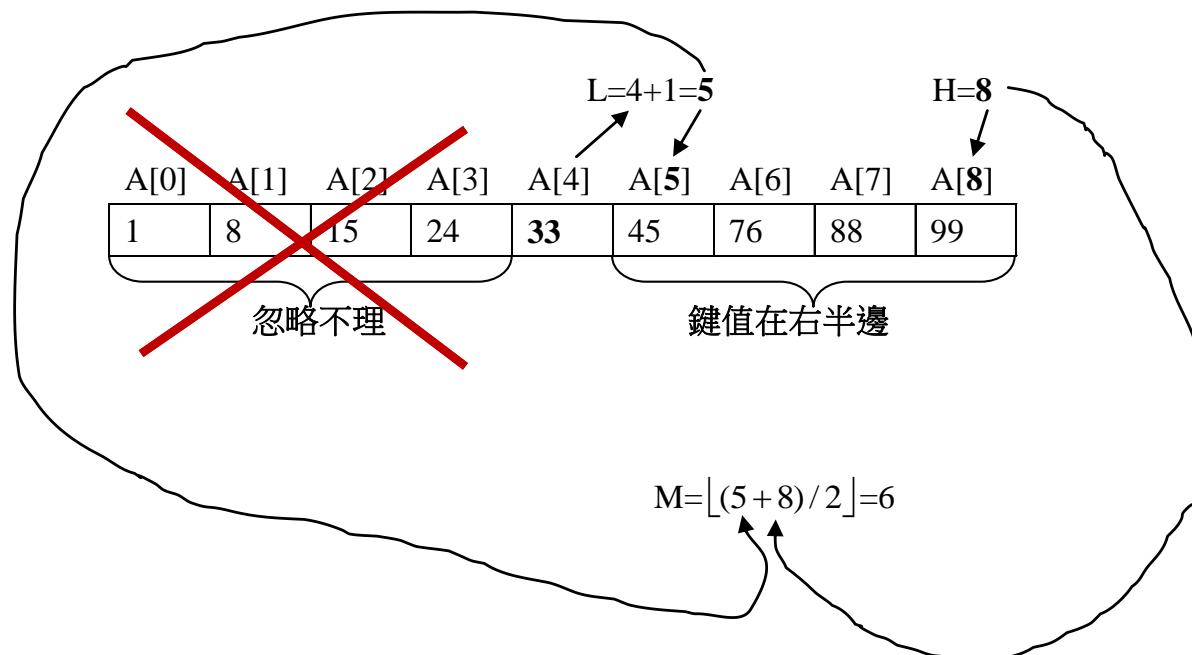
步驟④ 將「鍵值」和「中間值」做比較。

假設鍵值Key=88

因為，鍵值 $88 >$ 中間值33，則可確定要找的資料在右半邊的元素中，所以，我們要再計算「右半邊」資料的最小值索引L，其計算公式就是「中間值索引加1」。因此，就可以求得 $L=5$

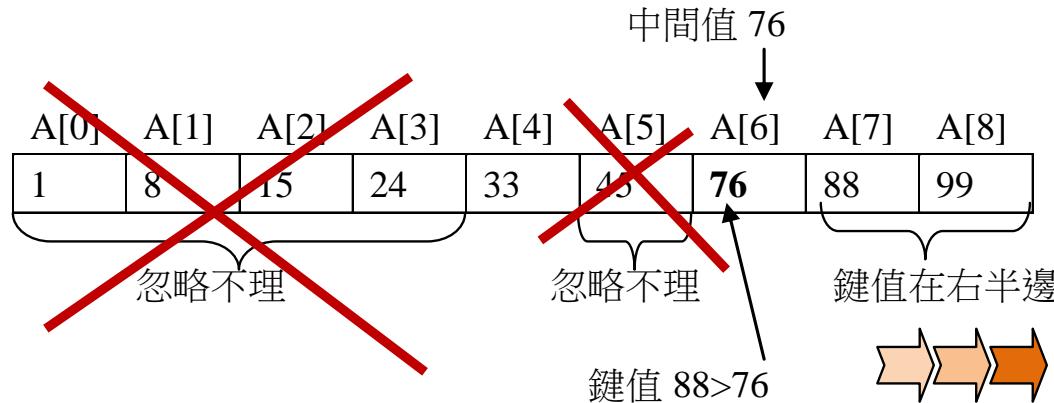


步驟⑤ 針對右半邊的資料，再重新計算中間值之索引M，其計算公式與步驟三相同，也就是L加H括號除以2再取下限，因此，我們就將可以 $L=5, H=8$ 的值代入到公式中，就可以順利求出 $M=6$

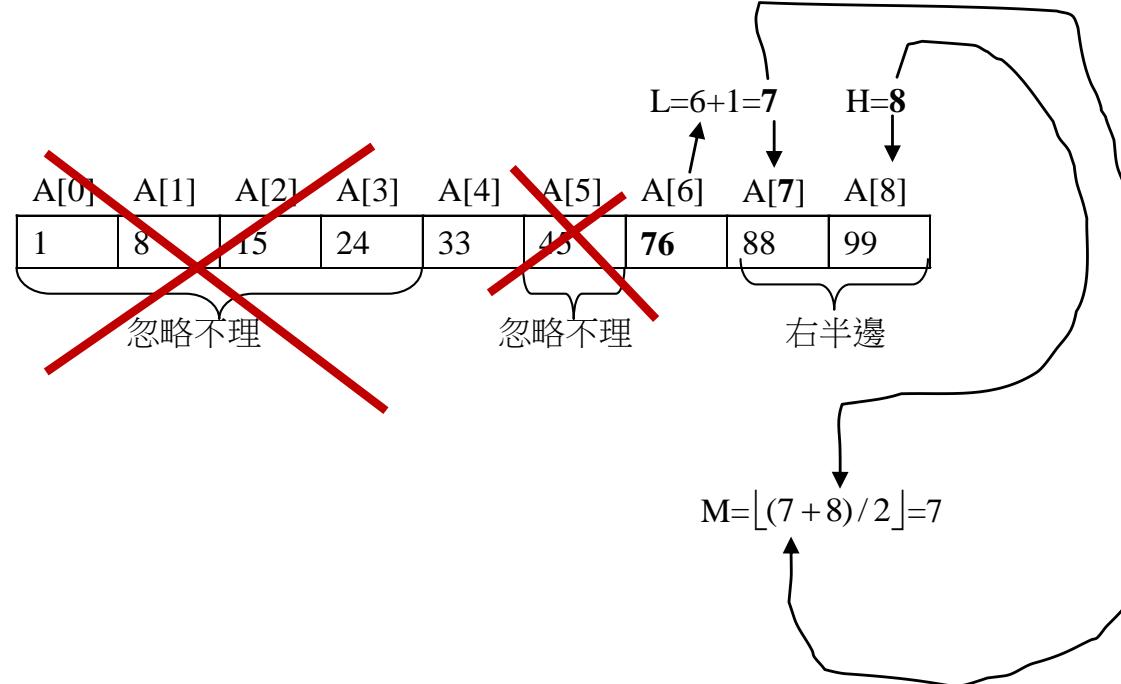


步驟⑥ 將「鍵值」和「中間值」做比較。

由於，鍵值88>中間值76，則可確定要找的資料在右半邊的元素中，所以，我們要再計算「右半邊」資料的最小值索引L，其計算公式就是「中間值索引加1」。因此，就可以求得 $L=7$

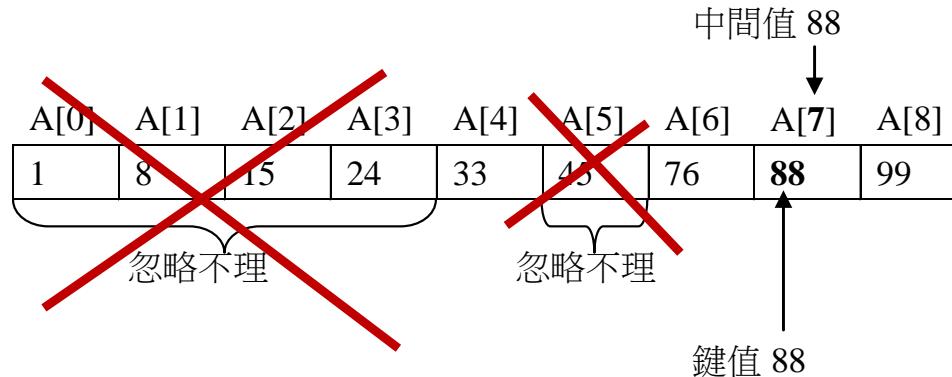


最後，針對右半邊的資料，再重新計算中間值之索引M，因此，我們就將可以L=7,H=8的值代入到公式中，就可以順利求出M=7



步驟⑦ 將「鍵值」和「中間值」做比較。

最後，因為鍵值88等於中間值88，所以找尋成功。



5-5 二維陣列

【引言】

在前面所介紹一維陣列，可以視為直線方式來存取資料，這對於一般的問題都可以順利的處理，但是對於比較複雜的問題時，那就必須要使用二維陣列來處理。否則會增加程式的複雜度。例如：計算4位同學的5科成績之總分與平均的問題。

【定義】宣告陣列時，其括弧內的「註標」個數，有兩個時稱為「二維陣列」。

【語法】~~資料型態[,] 陣列名稱=new 資料型態[M,N];~~

【說明】M代表列數，N代表行數

【例如】`int[][] Score = new int[4][5];`

//列註標表示範圍：0~3 共有4列

//行註標表示範圍：0~4 共有5行

在宣告之後，主記憶的邏輯配置如下所示：

行 列	第 0 行	第 1 行	第 2 行	第 3 行	第 4 行
第 0 列	Score [0][0]	Score [0][1]	Score [0][2]	Score [0][3]	Score [0][4]
第 1 列	Score [1][0]	Score [1][1]	Score [1][2]	Score [1][3]	Score [1][4]
第 2 列	Score [2][0]	Score [2][1]	Score [2][2]	Score [2][3]	Score [2][4]
第 3 列	Score [3][0]	Score [3][1]	Score [3][2]	Score [3][3]	Score [3][4]

【存取方法】利用二維陣列中的兩個註標來表示。

【示意圖】

教室座位的排列方式	點名(第二排第一名)
	



5-5.1 二維陣列的儲存方式

【定義】陣列名稱之後加上“註標”即可存取陣列元素。

【語法】陣列名稱[註標1][註標2]=資料來源;

【說明】(1)「陣列名稱」的命名規則和一般變數相同。

(2)「註標」必須是一數字型態。

【實作】假設我們現在2位同學，其成績如下：

第1位同學成績：60,65,70

第2位同學成績：80,85,90

請計算以上兩位同學的平均成績。

【第一種寫法】「指定陣列大小」

行號	程式檔名：ch5-5-1A.java
01	<code>public class ch5_5_1A {</code>
02	<code> public static void main(String[] args) {</code>
03	<code> int [] Avg=new int[2];</code>
04	<code> int[][] Score = new int[2][3];</code>
05	<code> Score[0][0]=60; Score[0][1]=65;Score[0][2]=70;</code>
06	<code> Score[1][0]=80; Score[1][1]=85;Score[1][2]=90;</code>
07	<code> for(int i=0;i<2;i++)</code>
08	<code> {</code>
09	<code> for(int j=0;j<3;j++)</code>
10	<code> {</code>
11	<code> Avg[i]=Avg[i]+Score[i][j];</code>
12	<code> }</code>
13	<code> System.out.println("Student"+ (i+1) + "=" + Avg[i]/3);</code>
14	<code> }</code>
15	<code>}</code>
16	<code>}</code>

【第二種寫法】「未指定陣列大小」必須要搭配「初值串列設定」

如果你想使用第二種寫法，就可以將上面程式碼中的行號04~06，匯集成一行如下行號05所示。

行號	程式檔名：ch5-5-1B.java
01	<code>public class ch5_5_1B {</code>
02	<code> public static void main(String[] args) {</code>
03	<code> -----</code>
04	<code> int Score[][]={{60,65,70},{80,85,90}};</code>
05	<code> -----</code>
06	<code> <略></code>
07	<code> }</code>
08	<code>}</code>

【執行結果】二維陣列的初值設定方法，請參考下一單元。

```
Student1=65  
Student2=85
```



5-5.2 二維陣列的初值設定

【定義】是指宣告陣列的同時並指定初值。

【目的】可以縮短程式的長度。

【語法】資料型別 陣列名稱[][] ={{初值串列1},{初值串列2},...,{初值串列n}};

【說明】宣告A是一個含有5個整數的陣列，其中初值為：

```
int[][] A = new int[3,3];
A[0][0] = 1; A[0][1] = 2; A[0][2] = 3;
A[1][0] = 4; A[1][1] = 5; A[1][2] = 6;
A[2][0] = 7; A[2][1] = 8; A[2][2] = 9;
```

其寫法如下：`int[][] A=new int[][]{{1,2,3},{4,5,6},{7,8,9}};`

也可以簡寫為：`int[] A ={{1,2,3},{4,5,6},{7,8,9}};`

【實作】

行號	程式檔名：ch5-5-2.java
01	<code>public class ch5_5_2 {</code>
02	<code> public static void main(String[] args) {</code>
03	<code> int [] Avg=new int[2];</code>
04	<code> int Score[][]={{60,65,70},{80,85,90}};</code>
05	<code> for(int i=0;i<2;i++)</code>
06	<code> {</code>
07	<code> for(int j=0;j<3;j++)</code>
08	<code> {</code>
09	<code> Avg[i]=Avg[i]+Score[i][j];</code>
10	<code> }</code>
11	<code> System.out.println("Student"+ (i+1) + "=" + Avg[i]/3);</code>
12	<code> }</code>
13	<code> }</code>
14	<code>}</code>

【執行結果】

```
Student1=65  
Student2=85
```

5-6 多維陣列的觀念



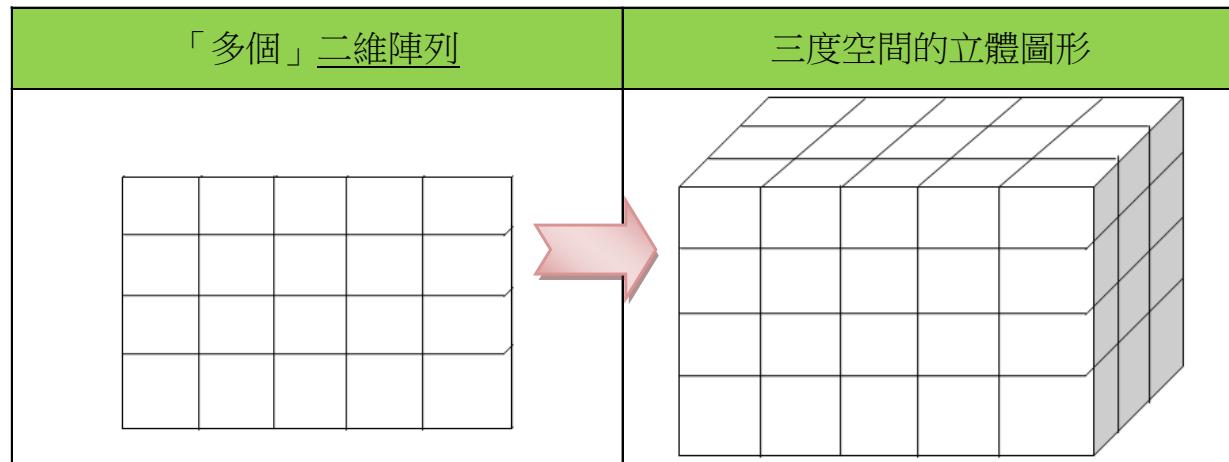
【定義】

宣告陣列時，其括弧內的「註標」個數，是二個以上時，就稱為「多維陣列」。

【三維陣列的思維】

我們可以將「三維陣列」視為「多個」二維陣列的組合。其圖形為三度空間的立體圖形。

【示意圖】



【語法】**資料型態[][][] 陣列名稱=new資料型態[L][M][N];**

【說明】L代表二維陣列個數

M代表列數

N代表行數

【例如】**int[][][] Score = new int[3][4][5];**

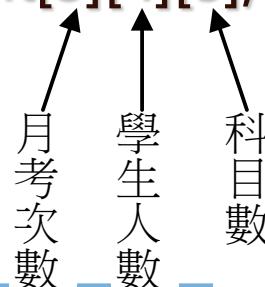
//二維陣列的個數：0~2 共有3個二維陣列

//列註標表示範圍：0~3 共有4列

//行註標表示範圍：0~4 共有5行

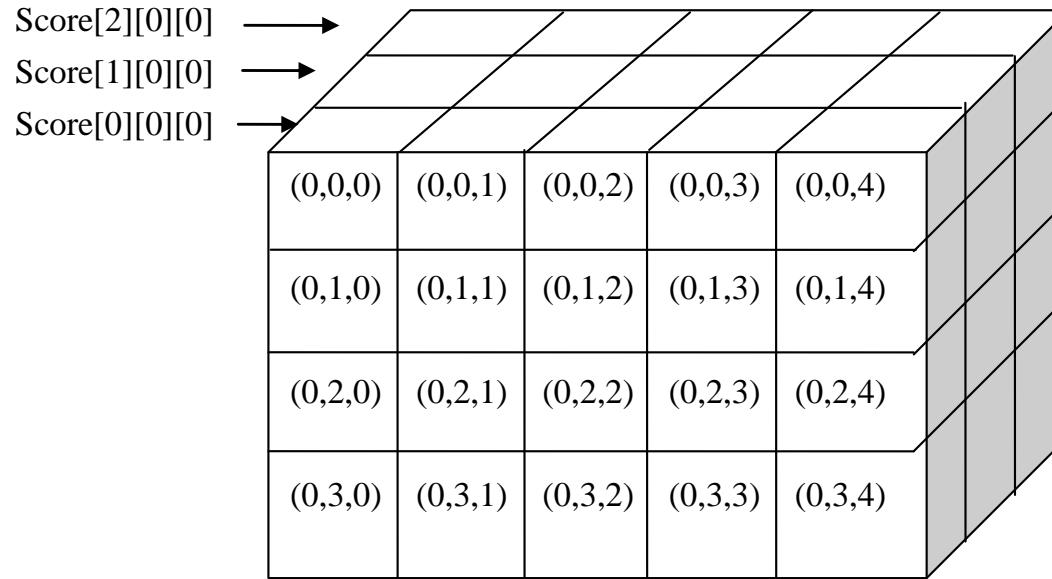
【舉例】設計一個某高中，3次月考，全班4位同學的5科目成績時。利用三維陣列來存取每人學生的成績。

int[][][] Score=new int[3][4][5];



【說明】此例子中Score陣列共有三個註標，故Score陣列是一個三維陣列。

宣告Score是由3個(0~2)二維陣列，每個二維陣列包含4列(0~3)，5行(0~4)組合而成的整數三維陣列。並且共計有 $3 \times 4 \times 5 = 60$ 元素。如下圖所示：



【說明】

此例子中Score陣列共有三個註標，故Score陣列是一個三維陣列。

//其中，第一個註標為：二維陣列的個數：0~2 共有3個二維陣列

第二個註標為：列註標表示範圍：0~3 共有4列

第三個註標為：行註標表示範圍：0~4 共有5行

【註】基本上，三維陣列在實作時，必須要使用到三重迴圈來讀取資料，因此，會增加程式的複雜度，除非必要，否則建議使用二維陣列來完成即可。

【實例】假設老師利用二維表格，來記錄學生的三次月考成績，如下所示：

第一次月考成績單

姓名	國文	英文	數學	資料庫	程式設計
張三	65	85	78	75	69
李四	66	55	52	92	47
王五	75	99	63	73	86
雄雄	77	88	99	91	99

第二次月考成績單



姓名	國文	英文	數學	資料庫	程式設計
張三	77	88	66	77	66
李四	65	66	88	55	77
王五	70	88	56	88	88
雄雄	80	90	95	99	99

第三次月考成績單

姓名	國文	英文	數學	資料庫	程式設計
張三	55	67	56	98	67
李四	66	69	76	66	78
王五	77	89	88	77	77
雄雄	88	89	99	97	88

【實作】

程式檔案名稱		ch5-6.java
01	public class ch5_6	
02	{	
03	public static void main(String[] args)	
04	//宣告及初值設定	
05	int i, j,k;	
06	int[][] Stu_Sum = new int[3][4]; //學生總成績	
07	int[][] Stu_Average = new int[3][4]; //學生平均成績	
08	int[][] Subject_Sum = new int[3][5]; //科目總成績	
09	int[][] Subject_Average = new int[3][5]; //科目平均成績	
10	String[] Stu_Name = { "張三", "李四", "王五", "雄雄" };	
11	int[][][] Score = { { { 65, 85, 78, 75, 69 }, { 66, 55, 52, 92, 47 }, { 75, 99, 63, 73,	
12	86 }, { 77, 88, 99, 91, 99 } }, { { 77, 88, 66, 77, 66 }, { 65, 66, 88, 55, 77 }, { 70, 88, 56,	
13	88, 88 }, { 80, 90, 95, 99, 99 } }, { { 55, 67, 56, 98, 67 }, { 66, 69, 76, 66, 78 }, { 77, 89, 88,	
14	77, 77 }, { 88, 89, 99, 97, 88 } } };	
15	//處理	
16	// ===== 讀取資料並計算各科總分=====	
17	for (k = 0; k <= 2; k++) //3次月考	
18	for (i = 0; i <= 3; i++) //控制列數	
19	for (j = 0; j <= 4; j++) //控制行數	
20	{ //計算出每一位「同學」的總成績與每一「科目」的總成績	
21	Stu_Sum[k][i] = Stu_Sum[k][i] + Score[k][i][j];	
22	Subject_Sum[k][j] = Subject_Sum[k][j] + Score[k][i][j];	
23	}	
24	//===== 印出開頭=====	
25	for (k = 0; k <= 2; k++)	
26	{	
27	System.out.println("=====第" + (k + 1) + "次月考成績如下：=====");	
28	System.out.println("姓名 國文 英文 數學 計概 程設");	
29	System.out.println("=====");	
30	// ===== 印出成績單=====	
31	for (i = 0; i <= 3; i++) //控制列數	
32	{	
33	System.out.print(Stu_Name[i]);	
34	for (j = 0; j <= 4; j++) //控制行數	
35	System.out.print(" " + Score[k][i][j]);	
36	System.out.println();	
37	}	
38	System.out.println();	
39	}	
40	}	
41	}	

【執行結果】

=====第1次月考成績如下：=====

姓名	國文	英文	數學	計概	程設
----	----	----	----	----	----

=====

張三	65	85	78	75	69
----	----	----	----	----	----

李四	66	55	52	92	47
----	----	----	----	----	----

王五	75	99	63	73	86
----	----	----	----	----	----

雄雄	77	88	99	91	99
----	----	----	----	----	----

=====第2次月考成績如下：=====

姓名	國文	英文	數學	計概	程設
----	----	----	----	----	----

=====

張三	77	88	66	77	66
----	----	----	----	----	----

李四	65	66	88	55	77
----	----	----	----	----	----

王五	70	88	56	88	88
----	----	----	----	----	----

雄雄	80	90	95	99	99
----	----	----	----	----	----

=====第3次月考成績如下：=====

姓名	國文	英文	數學	計概	程設
----	----	----	----	----	----

=====

張三	55	67	56	98	67
----	----	----	----	----	----

李四	66	69	76	66	78
----	----	----	----	----	----

王五	77	89	88	77	77
----	----	----	----	----	----

雄雄	88	89	99	97	88
----	----	----	----	----	----



課後評量



1. 任意輸入五個數值資料，求出此五個數字中，有幾個數是小於此五個數值資料的平均值。

【執行畫面】

請輸入五組數字(利用「逗號，」分離)

11,22,33,44,55

共有2個數小於平均值33

2. 輸入下列10個數值23,22,51,93,5,100,34,66,77,80，請設計一個程式求出偶數的個數及奇數的個數。

【執行畫面】

請輸入十組數字(利用「逗號，」分離)

11,22,33,44,55,66,77,88,99,100

偶數個數5

奇數個數5

3. 請輸入8位同學成績，並依照成績的高低排名次(同分時也可以使用)

【執行畫面】

ALETHEIA
UNIVERSITY

請輸入8位同學成績(利用「逗號，」分離)

77,66,66,99,85,78,89,99

學號 程式設計 名次

1	77	6
2	66	7
3	66	7
4	99	1
5	85	4
6	78	5
7	89	3
8	99	1



4. 印出方形數字方塊(數字列印)

- (1) 說明：A. 請以迴圈指令印出如下圖的數字方塊。
B. 數字方塊外層數字可由程式中設定。

(2) 執行畫面：

```
555555555  
544444445  
543333345  
543222345  
543212345  
543222345  
543333345  
544444445  
555555555
```

5. 利用統計圖來呈現學生在網路學習系統中，每天的線上時間。

星期	一	二	三	四	五	六	日
時間(小時)	8	5	2	7	14	10	15

【執行畫面】

```
1      ****
2      ***
3      **
4      ****
5      *****
6      *****
7      *****
```

本週總閱讀時間：61

6. 設計一個 4×4 的二維陣列，再輸入16個數，請將元素 $A(i, j)$ 與 $A(j, i)$ 互換，並輸出結果，情況如下：

=====轉換前=====

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

=====轉換後=====

1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

7. 任意輸入2~10位學生成績，找最低分、最高分與總平均。

請輸入8位同學成績(利用「逗號，」分離)

11,22,33,44,55,66,77,88

最低分=11

最高分=88

總平均=49.5

8. 利用亂數任意產生10個整數，其範圍是在1~100之間，先進行「排序」之後，再以程式來計算介於70~90之間共有幾個。

產生10個亂數值(範圍：1~100)：

58 63 75 35 77 81 59 18 92 46

=====氣泡排序法=====

18 35 46 58 59 63 75 77 81 92

=====氣泡排序法=====

介於70~90之間的個數有：3個

9. 試寫一個程式，計算兩個矩陣之相加。

分析：

假設有兩個矩陣之初值設定如下：

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \quad B = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$

先宣告一個C陣列，來存放A+B矩陣的值

$$C = \begin{bmatrix} 1+5 & 2+6 \\ 3+7 & 4+8 \end{bmatrix} = \begin{bmatrix} 6 & 8 \\ 10 & 12 \end{bmatrix}$$

==輸出A矩陣==

[1 2]

[3 4]

==輸出B矩陣==

[5 6]

[7 8]

==兩個矩陣相加==

[6 8]

[10 12]

10. 試寫一個程式，計算兩個矩陣之相乘。

分析：

假設有兩個矩陣之初值設定如下：

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \quad B = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$

宣告一個C陣列，來存放B*A矩陣的值

在數學上矩陣相乘的公式：

$$C[i,j] = \sum_{k=0}^1 A[j,k] * B[k,j]$$

$$E = \begin{pmatrix} 1*5+2*7 & 1*6+2*8 \\ 3*5+4*7 & 3*6+4*8 \end{pmatrix} = \begin{pmatrix} 19 & 22 \\ 43 & 50 \end{pmatrix}$$

==輸出A矩陣==

```
[1 2]  
[3 4]
```

==輸出B矩陣==

```
[5 6]  
[7 8]
```

==兩個矩陣相乘==

```
[19 22]  
[43 50]
```



11. 10進位轉換成2、8與16進位

請輸入十進位數字

15

2進位為：1111

8進位為：17

16進位為：F



12. 【題目】迴文判斷

【說明】

請從表單中讀取一個欲判斷的數字，若此數字為迴文
(Palindrome,左右讀起均同，例如12321)、則印出此數字及“是
迴文數字”，若不是則印出此數字及“不是迴文數字”

【輸入範例】12321

【輸出報表】12321是迴文數字，如下所示：

是迴文數字	不是迴文數字
請輸入一組數字 12321	請輸入一組數字 12345
12321 是迴文數字	12345不是迴文數字