# 快速排序練習程式介紹

## A3261576 林郁翔

```
★ 練習 - Microsoft Visual Studio
檔案(F) 編輯(E) 檢視(V) 專案(P) 建置(B) 傾錯(D) 小組(M) 工具(T) 測試(S) 分析(N) 視窗(W) 說明(H)
 ③ ▼ ⑤ | 🏗 ▼ 🚰 💾 🤚 🤊 ▼ 🤍 ▼ | ▶ 本機 Windows 偵錯工具 ▼ 💍 ▼ | Debug ▼ | Win32
                                                                          - | 🗓 | 🎜 🍦 造 🏗 | 🖫 🤨
  Source.cpp* → ×
  型 練習
                                         (全域節團)
                                                                            → Ø main()
       void quickSort(int*myarray, int l
      int Partition(int*myarray, int le
                                   [0]=2
      //快速排序
      //Date:20150321
                                   [2]=6
       //Author:林郁翔
                                   [3]=8
                                   [4]=10
     ⊟int main()
                                   [6]=37
      -
          int myarray[10]{37, 2, 6, 4,
                                   [8]=68
          quickSort(myarray, 0, 9);
for (int i = 0; i < 10; i++)
                                   [9]=89
請按任意鍵繼續 - - -
             Cout << "[" << i << "]="
          system("pause");
          return 0:
  100 % 🕶 🖣
   顯示輸出來源(S): 負錯
                                   微軟注音 半:
```

#### (Screen shot)

解介:快速排序,使用遞迴、指標達成。

本程式預設陣列為課本 8-15 上提供的陣列。

#### Main

```
// Date: 20150321
// Author: 林郁翔
//==
Dint main()
{
    int myarray[10]{37, 2, 6, 4, 89, 8, 10, 12, 68, 45};
    quickSort(myarray, 0, 9);
    for (int i = 0; i < 10; i++)
    {
        cout << "[" << i << "]=" << myarray[i] << endl;
    }

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

- 1.在 main 中宣告一個陣列為 10 大小的陣列
- 2.呼叫 quickSort 進行排序
- 3.印出每個元素的 Index 以及 value

# quickSort

```
Dvoid quickSort(int*myarray, int left, int right)

{
    if (left < right){
        int pivotLocation = Partition(myarray, left, right);
        quickSort(myarray, left, pivotLocation - 1);
        quickSort(myarray, pivotLocation + 1, right);
    }
}</pre>
```

1. 主要用來執行遞迴步驟的函式,分別對左右執行切割動作。

#### swap

1.使指標傳遞使元素作交換動作

### **Partition**

```
1 練習
  □int Partition(int*myarray, int left, int right)
        int pvalue = myarray[left]; //分割點的值
        int pIndex = left; //分割點的index
        int changeR = right, changeL = left;
        int count = 0;
        while (1)
            for (int i = changeR; i >= left; i--)
                if (pvalue > myarray[i])
                    swap(myarray[pIndex], myarray[i]);
                    changeL = pIndex;
                    changeL++; //下次從r的下個位置
                    pIndex = i; //更新點位置
                    break:
                if (i == left)
                    return pIndex;
            for (int i = changeL; i <= pIndex; i++)</pre>
                if (pvalue < myarray[i])</pre>
                    swap(myarray[pIndex], myarray[i]);
                    changeR = pIndex;
                    changeR--; //下次從L的下個位置
                    pIndex = i; //更新點位置
                    break;
                if (i == pIndex)
                    return pIndex;
            }
        }
```

我把程式分成兩區段,(1)是從右邊找到左邊;(2)是從左邊找到右邊

- 1. 我宣告一個 Pvalue 當作主要分割點的值
- 2. 宣告一個 pindex 當作分割點的索引值,並且當作回傳值使用
- 3. ChangeL 是當作下次從哪邊開始的點;ChangeR 亦同

- 4. 進入 if 區段後先做交換,並把兩個元素的 index 分別指定到 ChangeL 或 R,並且更新 plndex 即當前交換完後的分割點索引。
- 5. 並把它做 -1 或 +1 的動作,使下次會從下個值開始查找 如果成功進到 if 區段那麼便會 break 跳離 for 迴圈,繼續執行 while 迴圈的 動作。但假若 for 跑完後都沒有進入 if 區段,代表此時值已經不用再被替換了,因此我在這寫成 「若剛好等於條件式的界線就跳出」,這步想很久,不知道有沒有發現上面有個 int count=0; 原本想用計數來跳離迴圈,最後發現用上面的方法比較快。※註程式裡面已經把 count 變數刪除了
- 6. 假如 執行的 if(i==pIndex)或 if(i==left)則 return 當前的 pIndex,即最後交換完的最終位置。
- 7. 交給遞迴來跑,最後就達成排序了。