Introduction to Algorithms HW1

109511207 蔡宗儒

Pseudo Code

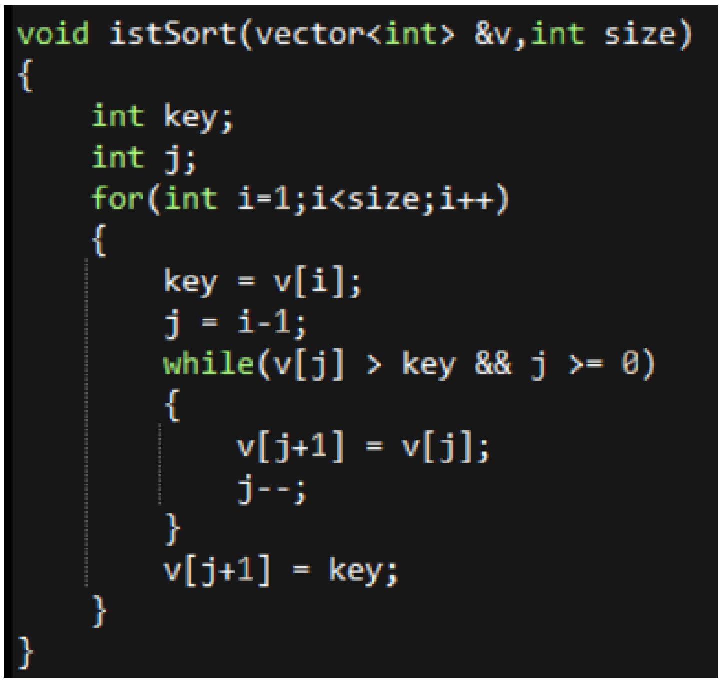
```
INSERTION-SORT(A)
1.  for j = 2 to A.length
2.      key = A[j]
3.      i = j-1
4.      while i > 0 & A[i] > key
5.          A[i+1] = A[i]
6.          i = i – 1
7.      A[i+1] = key
```

```
MERGE(A,p,q,r)
1.  n1 = q-p+1
2.  n2 = r-q
3.  let L[1…n1+1] and R[1…n2+1] be new arrays
4.  for i = 1 to n1
5.      L[i] = A[p+i-1]
6.  for j = 1 to n2
7.      R[j] = A[q+j]
8.  L[n1+1] = MAX
9.  R[n2+1] = MAX
10. i = 1
11. j = 1
12. for k = p to r
13.     if L[i] <= R[j]
14.         A[k] = L[i]
15          i = i+1
16.     else A[k] = R[j]
17.         j = j+1
```

```
MERGE-SORT(A,p,r)
1.  if p<r
2.      q = (p+r)/2
3.      MERGE-SORT(A,p,q)
4.      MERGE-SORT(A,q+1,r)
5.      MERGE(A,p,q,r)
```

Introduction of Function

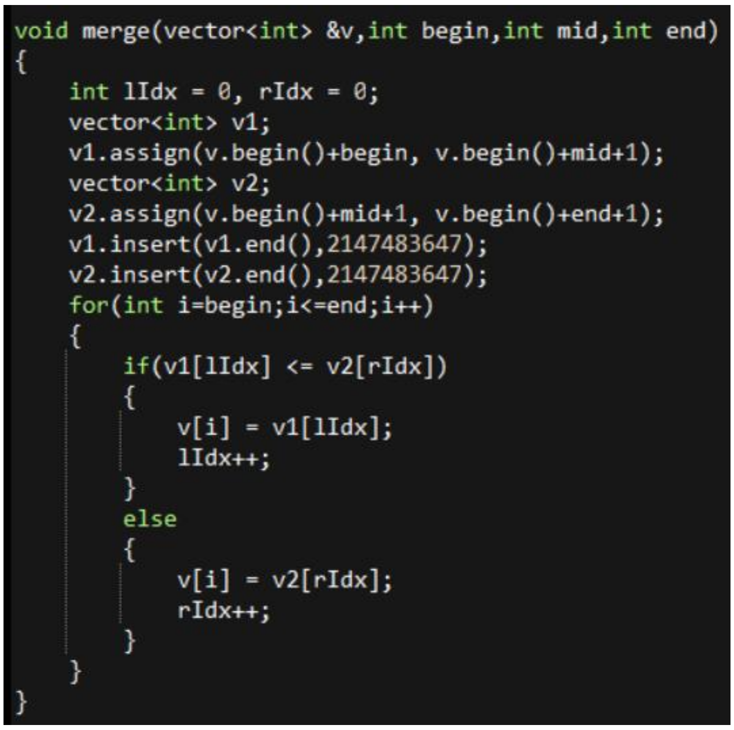
Insertion Sort



```
void istSort(vector<int> &v,int size)
{
    int key;
    int j;
    for(int i=1;i<size;i++)
    {
        key = v[i];
        j = i-1;
        while(v[j] > key && j >= 0)
        {
            v[j+1] = v[j];
            j--;
        }
        v[j+1] = key;
    }
}
```

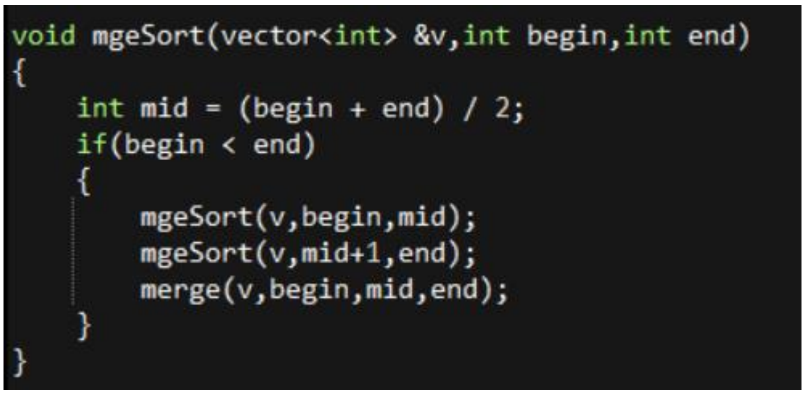
將 key 設為現在要比較的 element，當 key 小於前面已排序的 data 時，將前面較大的 data 往後放一個位置，直到 key 大於前面的 data 時，停止這些步驟，把 key 的值放進此 data 之後。

Merge Sort



```
void merge(vector<int> &v,int begin,int mid,int end)
{
    int lIdx = 0, rIdx = 0;
    vector<int> v1;
    v1.assign(v.begin()+begin, v.begin()+mid+1);
    vector<int> v2;
    v2.assign(v.begin()+mid+1, v.begin()+end+1);
    v1.insert(v1.end(),2147483647);
    v2.insert(v2.end(),2147483647);
    for(int i=begin;i<=end;i++)
    {
        if(v1[lIdx] <= v2[rIdx])
        {
            v[i] = v1[lIdx];
            lIdx++;
        }
        else
        {
            v[i] = v2[rIdx];
            rIdx++;
        }
    }
}
```

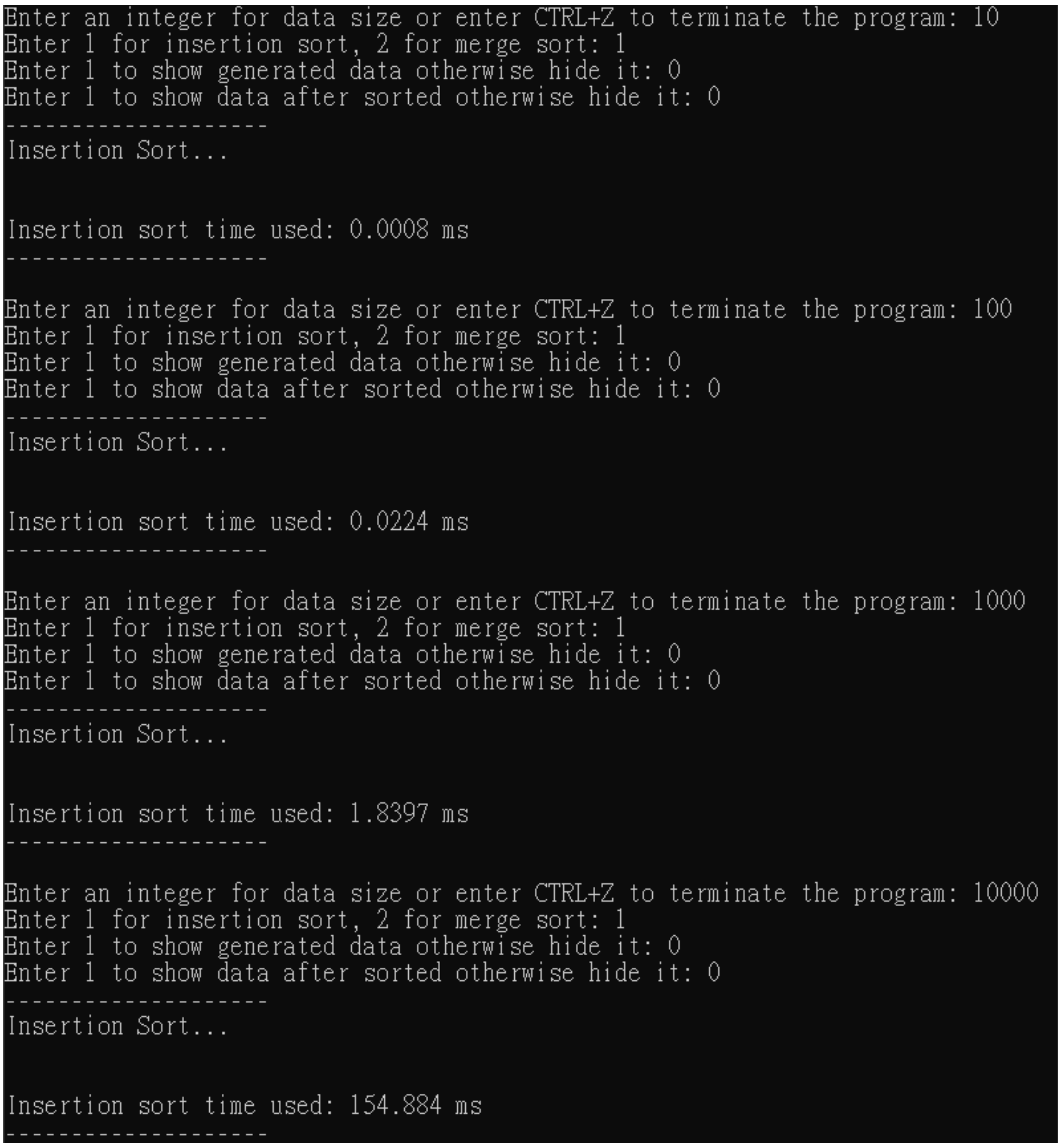
將原本的 array 從中間分成兩半的 subarray，在將兩個 subarray 的較小值依序放回原本的 array。



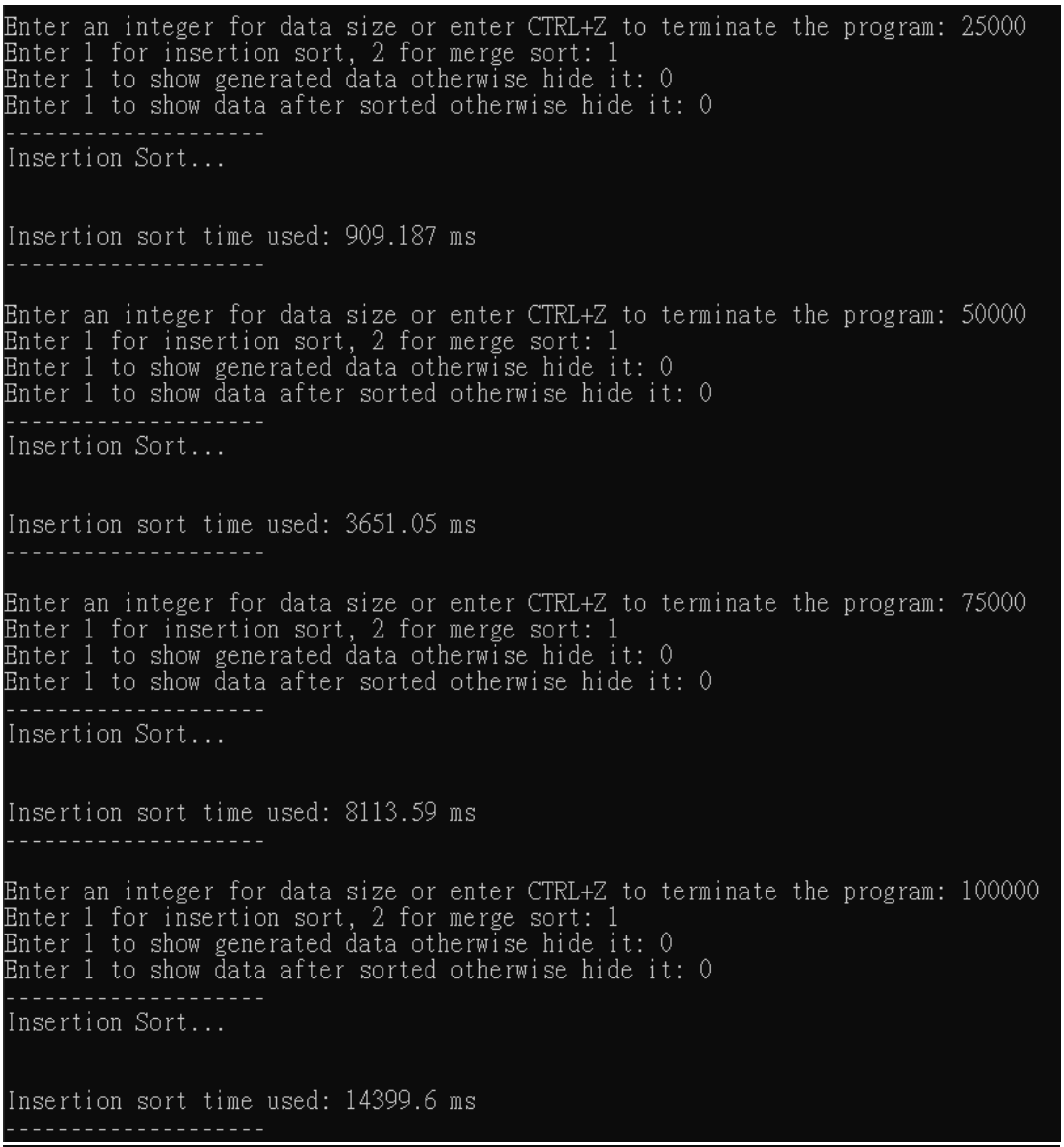
```
void mgeSort(vector<int> &v,int begin,int end)
{
    int mid = (begin + end) / 2;
    if(begin < end)
    {
        mgeSort(v,begin,mid);
        mgeSort(v,mid+1,end);
        merge(v,begin,mid,end);
    }
}
```

以 recursive 的方式，不斷的重複執行 merge 的動作直到目標陣列建立完成。

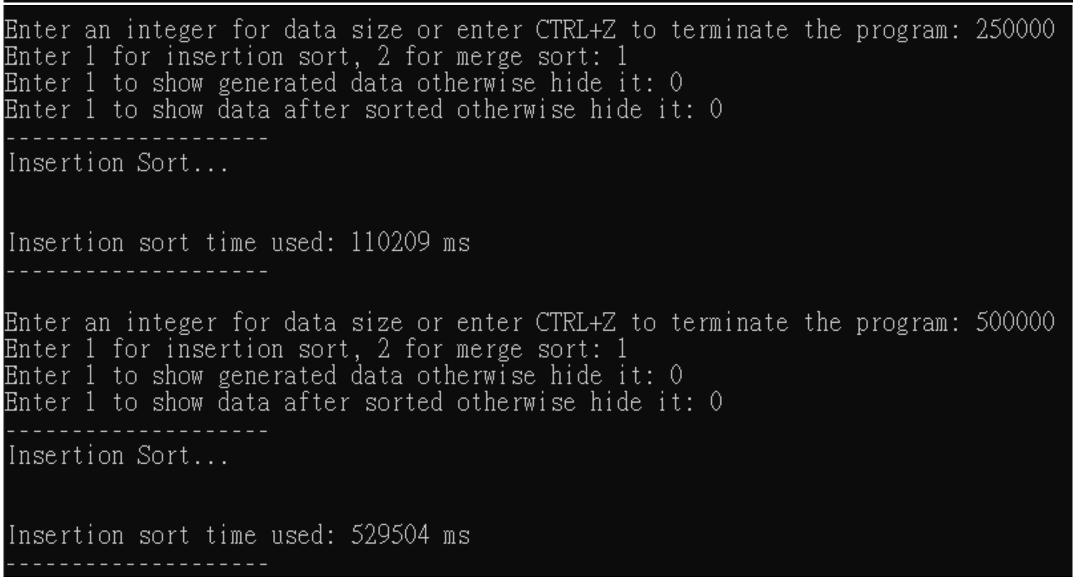
Insertion Sort 執行結果



```
Enter an integer for data size or enter CTRL+Z to terminate the program: 10
Enter 1 for insertion sort, 2 for merge sort: 1
Enter 1 to show generated data otherwise hide it: 0
Enter 1 to show data after sorted otherwise hide it: 0
--------------------
Insertion Sort...


Insertion sort time used: 0.0008 ms
--------------------

Enter an integer for data size or enter CTRL+Z to terminate the program: 100
Enter 1 for insertion sort, 2 for merge sort: 1
Enter 1 to show generated data otherwise hide it: 0
Enter 1 to show data after sorted otherwise hide it: 0
--------------------
Insertion Sort...


Insertion sort time used: 0.0224 ms
--------------------

Enter an integer for data size or enter CTRL+Z to terminate the program: 1000
Enter 1 for insertion sort, 2 for merge sort: 1
Enter 1 to show generated data otherwise hide it: 0
Enter 1 to show data after sorted otherwise hide it: 0
--------------------
Insertion Sort...


Insertion sort time used: 1.8397 ms
--------------------

Enter an integer for data size or enter CTRL+Z to terminate the program: 10000
Enter 1 for insertion sort, 2 for merge sort: 1
Enter 1 to show generated data otherwise hide it: 0
Enter 1 to show data after sorted otherwise hide it: 0
--------------------
Insertion Sort...


Insertion sort time used: 154.884 ms
--------------------
```



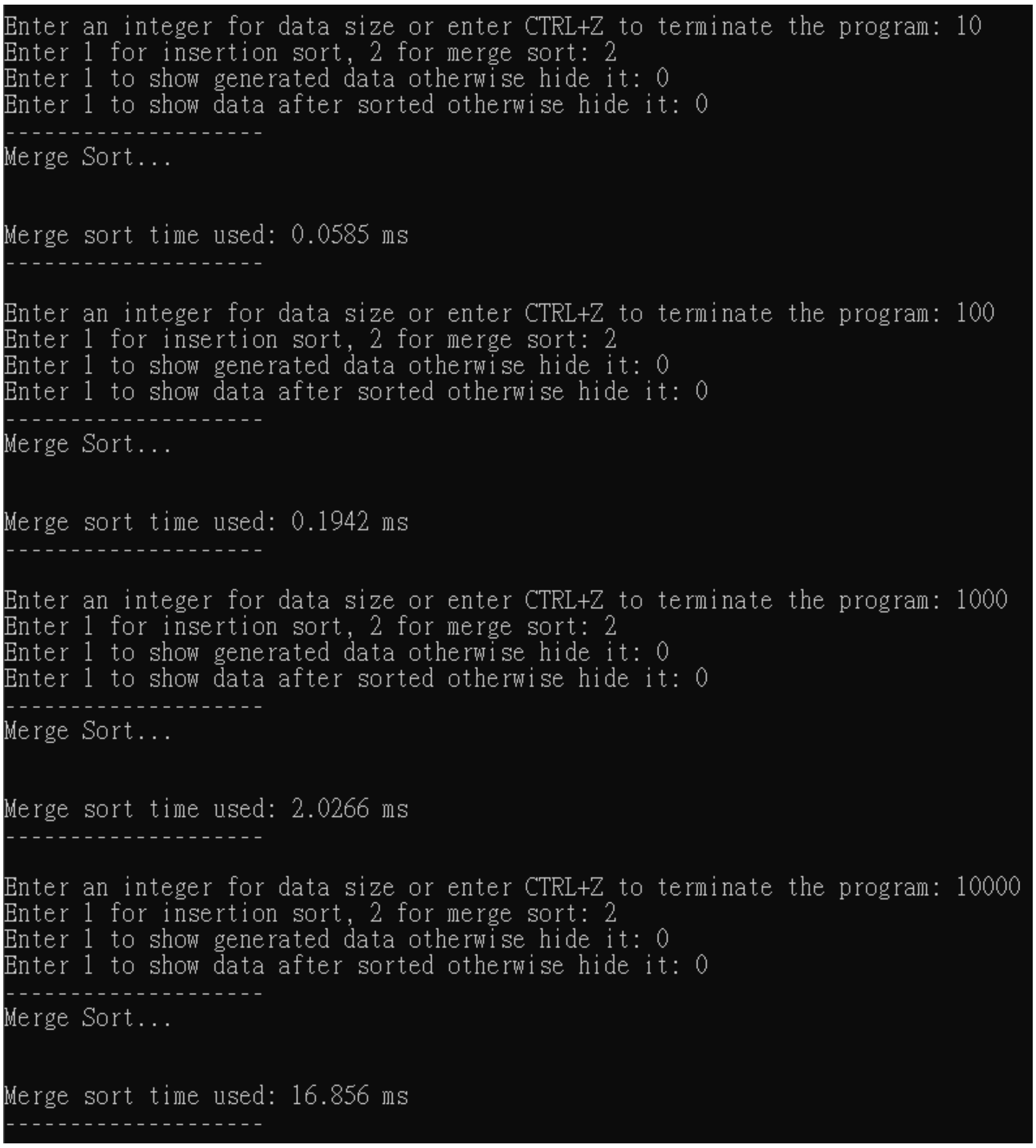
```
Enter an integer for data size or enter CTRL+Z to terminate the program: 25000
Enter 1 for insertion sort, 2 for merge sort: 1
Enter 1 to show generated data otherwise hide it: 0
Enter 1 to show data after sorted otherwise hide it: 0
--------------------
Insertion Sort...


Insertion sort time used: 909.187 ms
--------------------

Enter an integer for data size or enter CTRL+Z to terminate the program: 50000
Enter 1 for insertion sort, 2 for merge sort: 1
Enter 1 to show generated data otherwise hide it: 0
Enter 1 to show data after sorted otherwise hide it: 0
--------------------
Insertion Sort...


Insertion sort time used: 3651.05 ms
--------------------

Enter an integer for data size or enter CTRL+Z to terminate the program: 75000
Enter 1 for insertion sort, 2 for merge sort: 1
Enter 1 to show generated data otherwise hide it: 0
Enter 1 to show data after sorted otherwise hide it: 0
--------------------
Insertion Sort...


Insertion sort time used: 8113.59 ms
--------------------

Enter an integer for data size or enter CTRL+Z to terminate the program: 100000
Enter 1 for insertion sort, 2 for merge sort: 1
Enter 1 to show generated data otherwise hide it: 0
Enter 1 to show data after sorted otherwise hide it: 0
--------------------
Insertion Sort...


Insertion sort time used: 14399.6 ms
--------------------
```



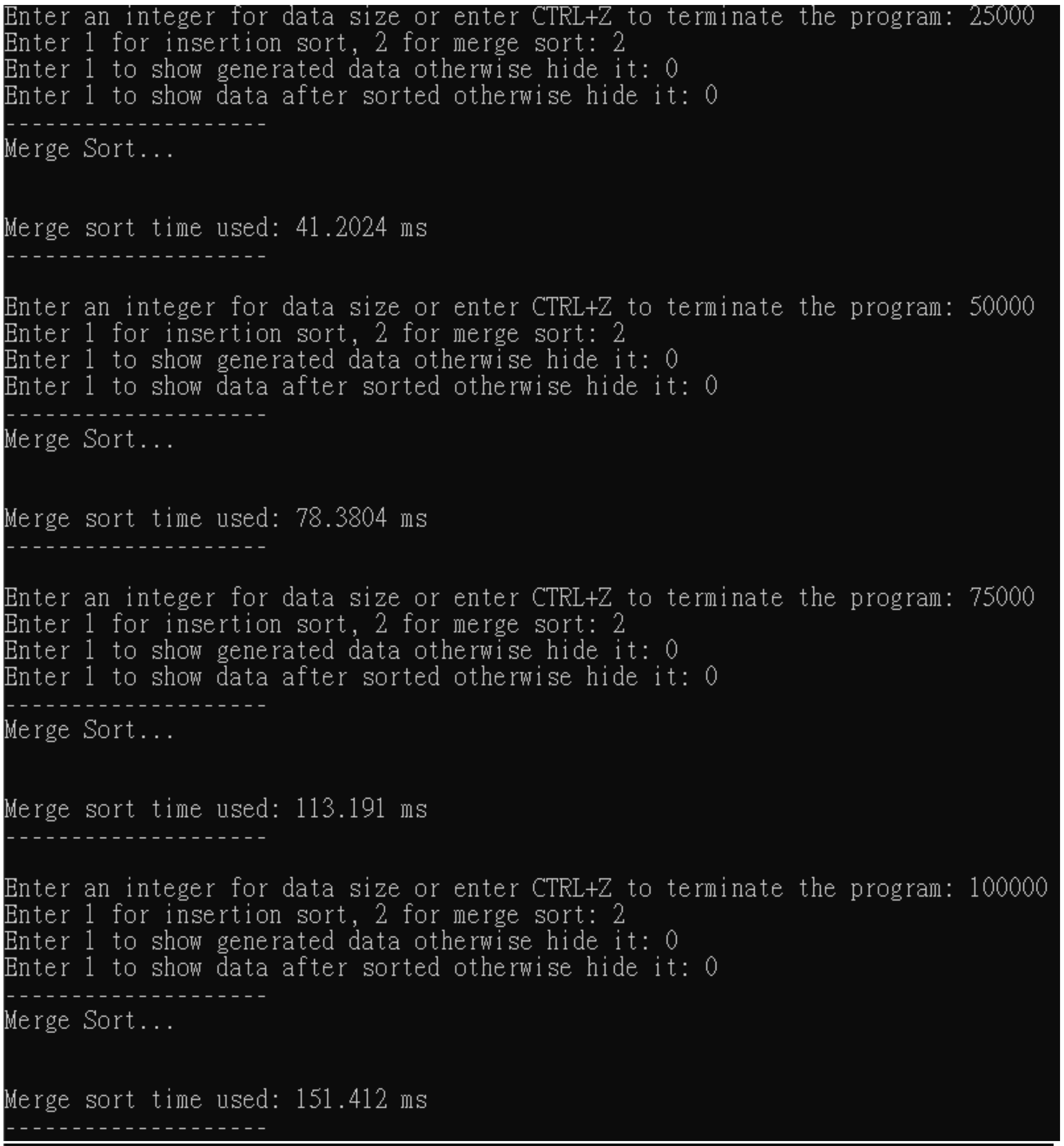
```
Enter an integer for data size or enter CTRL+Z to terminate the program: 250000
Enter 1 for insertion sort, 2 for merge sort: 1
Enter 1 to show generated data otherwise hide it: 0
Enter 1 to show data after sorted otherwise hide it: 0
--------------------
Insertion Sort...


Insertion sort time used: 110209 ms
--------------------

Enter an integer for data size or enter CTRL+Z to terminate the program: 500000
Enter 1 for insertion sort, 2 for merge sort: 1
Enter 1 to show generated data otherwise hide it: 0
Enter 1 to show data after sorted otherwise hide it: 0
--------------------
Insertion Sort...


Insertion sort time used: 529504 ms
--------------------
```

上三圖為 Insertion Sort 在 input size 為 10, 100, 1000, 10000, 25000, 50000, 75000, 100000, 250000, 500000 所花的時間。

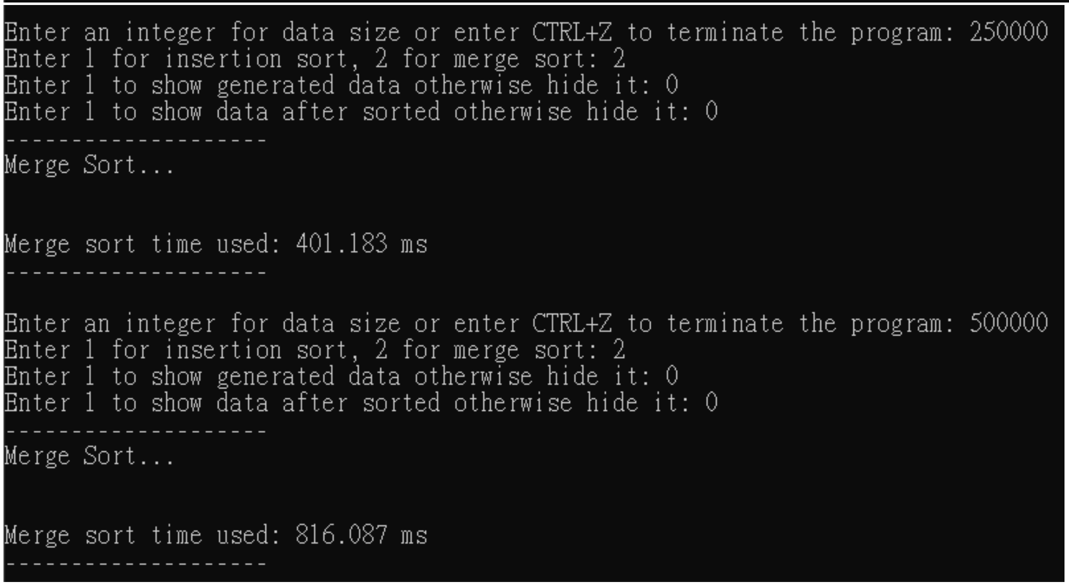
Merge Sort 執行結果



```
Enter an integer for data size or enter CTRL+Z to terminate the program: 10
Enter 1 for insertion sort, 2 for merge sort: 2
Enter 1 to show generated data otherwise hide it: 0
Enter 1 to show data after sorted otherwise hide it: 0
--------------------
Merge Sort...


Merge sort time used: 0.0585 ms
--------------------

Enter an integer for data size or enter CTRL+Z to terminate the program: 100
Enter 1 for insertion sort, 2 for merge sort: 2
Enter 1 to show generated data otherwise hide it: 0
Enter 1 to show data after sorted otherwise hide it: 0
--------------------
Merge Sort...


Merge sort time used: 0.1942 ms
--------------------

Enter an integer for data size or enter CTRL+Z to terminate the program: 1000
Enter 1 for insertion sort, 2 for merge sort: 2
Enter 1 to show generated data otherwise hide it: 0
Enter 1 to show data after sorted otherwise hide it: 0
--------------------
Merge Sort...


Merge sort time used: 2.0266 ms
--------------------

Enter an integer for data size or enter CTRL+Z to terminate the program: 10000
Enter 1 for insertion sort, 2 for merge sort: 2
Enter 1 to show generated data otherwise hide it: 0
Enter 1 to show data after sorted otherwise hide it: 0
--------------------
Merge Sort...


Merge sort time used: 16.856 ms
--------------------
```



```
Enter an integer for data size or enter CTRL+Z to terminate the program: 25000
Enter 1 for insertion sort, 2 for merge sort: 2
Enter 1 to show generated data otherwise hide it: 0
Enter 1 to show data after sorted otherwise hide it: 0
--------------------
Merge Sort...


Merge sort time used: 41.2024 ms
--------------------

Enter an integer for data size or enter CTRL+Z to terminate the program: 50000
Enter 1 for insertion sort, 2 for merge sort: 2
Enter 1 to show generated data otherwise hide it: 0
Enter 1 to show data after sorted otherwise hide it: 0
--------------------
Merge Sort...


Merge sort time used: 78.3804 ms
--------------------

Enter an integer for data size or enter CTRL+Z to terminate the program: 75000
Enter 1 for insertion sort, 2 for merge sort: 2
Enter 1 to show generated data otherwise hide it: 0
Enter 1 to show data after sorted otherwise hide it: 0
--------------------
Merge Sort...


Merge sort time used: 113.191 ms
--------------------

Enter an integer for data size or enter CTRL+Z to terminate the program: 100000
Enter 1 for insertion sort, 2 for merge sort: 2
Enter 1 to show generated data otherwise hide it: 0
Enter 1 to show data after sorted otherwise hide it: 0
--------------------
Merge Sort...


Merge sort time used: 151.412 ms
--------------------
```



```
Enter an integer for data size or enter CTRL+Z to terminate the program: 250000
Enter 1 for insertion sort, 2 for merge sort: 2
Enter 1 to show generated data otherwise hide it: 0
Enter 1 to show data after sorted otherwise hide it: 0
--------------------
Merge Sort...


Merge sort time used: 401.183 ms
--------------------

Enter an integer for data size or enter CTRL+Z to terminate the program: 500000
Enter 1 for insertion sort, 2 for merge sort: 2
Enter 1 to show generated data otherwise hide it: 0
Enter 1 to show data after sorted otherwise hide it: 0
--------------------
Merge Sort...


Merge sort time used: 816.087 ms
--------------------
```

上三圖為 Merge Sort 在 input size 為 10, 100, 1000, 10000, 25000, 50000, 75000, 100000, 250000, 500000 所花的時間。

Run Time Comparison

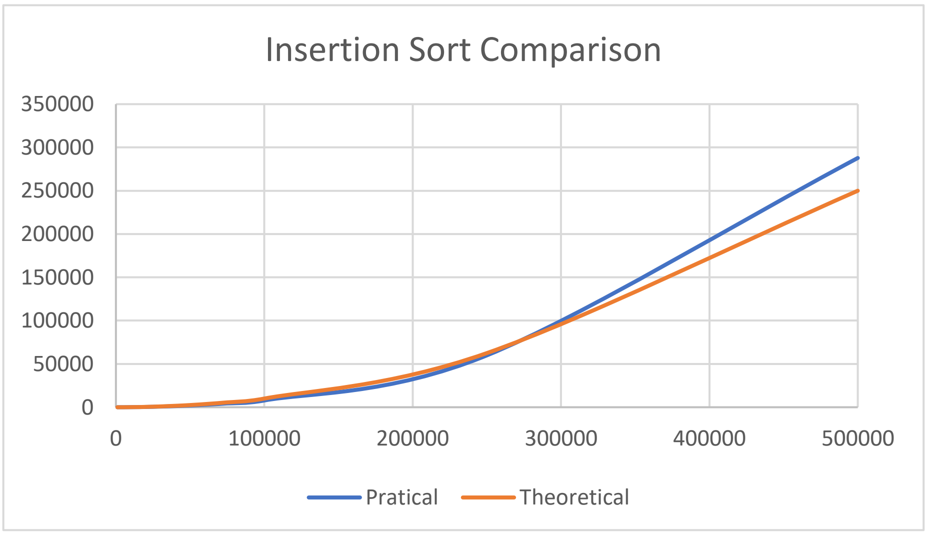
Input Size	Insertion Sort	Merge Sort
10	0.0008ms	0.0585ms
100	0.0224ms	0.1942ms
1000	1.8397ms	2.0226ms
10000	154.844ms	16.856ms
25000	909.187ms	41.2024ms
50000	3651.05ms	78.3804ms
75000	8113.59ms	113.191ms
100000	14399.6ms	151.412ms
250000	110209ms	401.183ms
500000	529504ms	816.087ms

My Observation & Conclusion

$$\begin{aligned} T(n) &= c_1 n + c_2(n-1) + c_4(n-1) + c_5\left(\frac{n(n+1)}{2} - 1\right) \\ &\quad + c_6\left(\frac{n(n-1)}{2}\right) + c_7\left(\frac{n(n-1)}{2}\right) + c_8(n-1) \\ &= \left(\frac{c_5}{2} + \frac{c_6}{2} + \frac{c_7}{2}\right)n^2 + \left(c_1 + c_2 + c_4 + \frac{c_5}{2} - \frac{c_6}{2} - \frac{c_7}{2} + c_8\right)n \\ &\quad - (c_2 + c_4 + c_5 + c_8)\,. \end{aligned}$$

Insertion sort 的 average case 和 worst case 時間複雜度都是 O(n^2) (如上課本的推導)。而我根據 code 執行後的結果，做出以下圖表。

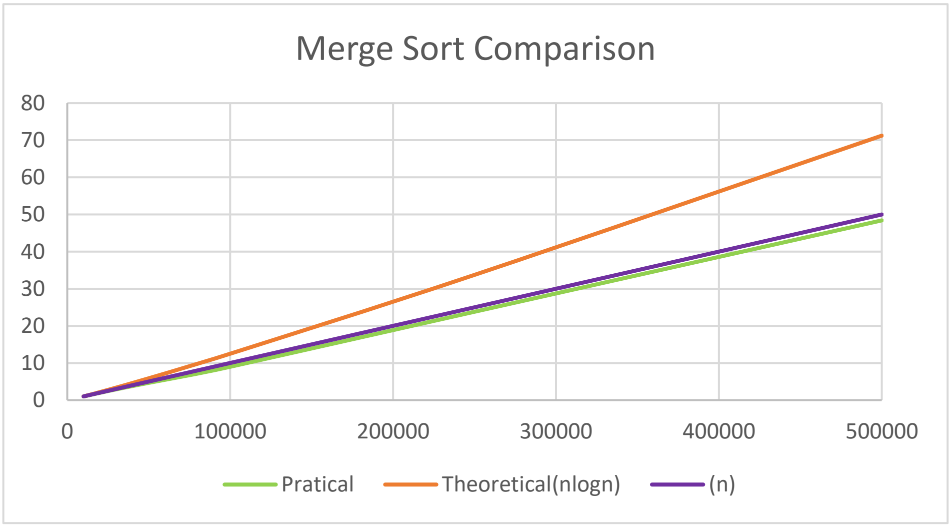
Size	1000	10000	25000	50000	75000	100000	250000	500000
Practical Run Time	1.8397	154.844	909.187	3651.05	8113.59	14399.6	110209	529504
Pratical (t(n)/t(1000))	1	84.1681	494.204	1984.59	4410.28	7827.15	59906	287821
Theoretical (n^2/1000^2)	1	100	625	2500	5625	10000	62500	250000



因為在 size 較小時，執行時間較短，可能會有較大的計算誤差，所以我只選擇了 size 大於 1000 的數據做分析，並以 1000 做為 unit size 計算 run time 是否符合理論值，呈現 n^2 的成長關係。其中 Practiacl 為不同 input size 與 input size = 1000 的實際 run time 比值，Theoretical 則是不同 input size 與 size = 1000 理論上的 run time 比值。可以發現在 input size 變成 n 倍時，practical run time 差不多變成 n^2 倍，與理論相符。

Merge sort 則是由 recursive 來實踐的，會將所有資料分成 $\log_2(n)$層（∵每次都從中間對半切)，每層有 n 筆 data，所以時間複雜度理論上會是 nlog(n)。而我也根據 code 執行後的結果，做出以下圖表。

Size	10000	25000	50000	75000	100000	250000	500000
Practical Run Time	16.856	41.2024	78.3804	113.191	151.412	401.183	816.087
Pratical	1	2.44438	4.65	6.71518	8.98268	23.8006	48.4152
Theoretical(nlogn)	1	2.74871	5.87371	9.14074	12.5	33.7371	71.2371
(n)	1	2.5	5	7.5	10	25	50



一樣因為在 size 較小時，執行時間較短，可能會有較大的計算誤差，所以我只選擇了 size 大於 10000 的數據做分析，並以 10000 做為 unit size 計算 run time 是否跟理論值一樣，呈現 nlog(n)的成長關係。其中 Practiacl 為不同 input size 與 input size = 10000 的實際 run time 比值，Theoretical 則是不同 input size 與 size = 10000 理論上的 run time 比值(應為 O(nlog(n))，但是數據更接近於 O(n)，所以兩個比值都做出來比較了)。然而我卻發現在 input size 變成 n 倍時，practical run time 卻比較接近變為 n 倍，與理論不符，我想了很久始終沒有得到一個合理的解釋，只能說實際的執行時間還是會被其他因素影響，與理論會有所出入。

而比較兩種 sort 在不同的 input size 下的 run time 可以發現，insertion sort 在 input size 較小時(size < 1000 左右)，有較快的 run time；反之 merge sort 在 input size 較大時(size > 1000 左右)，有較佳的 rum time。這與 insertion sort 這種 quadratic sorting algorithms 在 small data set 有較好的 efficiency，在 big data set 有較差的 efficiency 相符。