

7. Which of the following approach results in optimal time complexity for sorting a list?

- a) merge sort
- b) insertion sort
- c) bubble sort
- d) selection sort

【考題命中】計算機概論第四回，P.73~P.81，觀念完全命中

(二)插入排序法 (Insertion Sort)

1. 方法

將第*i*筆資料插入到前面*n-1*筆已經排好的資料記錄中，使之成為*i*筆已排好的資料序列，依此直到將所有資料排序完成。

Initial	13, 4, 54, 23, 1
1	<u>4</u> , 13, 54, 23, 1
2	4, <u>13</u> , 54, 23, 1
3	4, 13, <u>23</u> , 54, 1
4	<u>1</u> , 4, 13, 23, 54

3. 分析

(1) 時間複雜度

Best case：當資料都是由小排到大，*n*筆資料需要比*n-1*次，故時間複雜度為 $O(n)$ 。

Worst case：當資料是由大排到小，*n*筆資料需要比 $1+2+\dots+n-1$ 次 $\rightarrow n(n-1)/2$ 次，故時間複雜度為 $O(n^2)$ 。

(三)選擇排序法 (Selection Sort)

1. 方法

從第*i*到*n*筆中挑出最小值，在與第*i*筆資料交換，直到所有資料都排序完。

Initial	26, 8, 29, 80, 16, 18
1	<u>8</u> , <u>26</u> , 29, 80, 16, 18
2	8, <u>16</u> , 29, 80, <u>26</u> , 18
3	8, 16, <u>18</u> , 80, 26, <u>29</u>
4	8, 16, 18, <u>26</u> , <u>80</u> , 29
5	8, 16, 18, 26, <u>29</u> , <u>80</u>

2. 演算法

```
void selectionSort(int * list)
{
    for (int i = 0; i < n-1; i++) {
        int x = i;
        for (int j = i + 1; j <= n; j++) {
            if (list[j] < list[i]) x = j;
        }
        if (i != x) {
            swap(list[i], list[x]);
        }
    }
}
```

3. 分析

(1) 時間複雜度

Best case、Worst case和Average case皆是 $O(n^2)$ 。

(四)氣泡排序法 (Bubble Sort)

1. 方法

對元素兩兩做比較，在每回合中將最大元素上升到最高的格子。即在第*i*回合會將1到*n-i+1*中最大元素升到*n-i+1*格。

Initial	26, 18, 29, 80, 16, 8
1	26, 18, 29, 16, 8, 80
2	26, 18, 16, 8, 29, 80
3	18, 16, 8, 26, 29, 80
4	16, 8, 18, 26, 29, 80
5	8, 16, 18, 26, 29, 80

3. 分析

(1) 時間複雜度

- A. Best case：資料已經由小排到大 (1, 2, 3, 4, 5)，此時只需要比較*n-1*次，故時間複雜度為 $O(n)$ 。
- B. Worst case：資料由大排到小 (5, 4, 3, 2, 1)，需要比較 $(n-1)+\dots+2+1=n(n-1)/2$ 次，故時間複雜度為 $O(n^2)$ 。
- C. Average case： $O(n^2)$

(2) 為Stable sort

...8, 8⁺... → 因為8沒有大於8⁺，故不會產生交換。

(六)合併排序法 (Merge Sort)

1. 作法

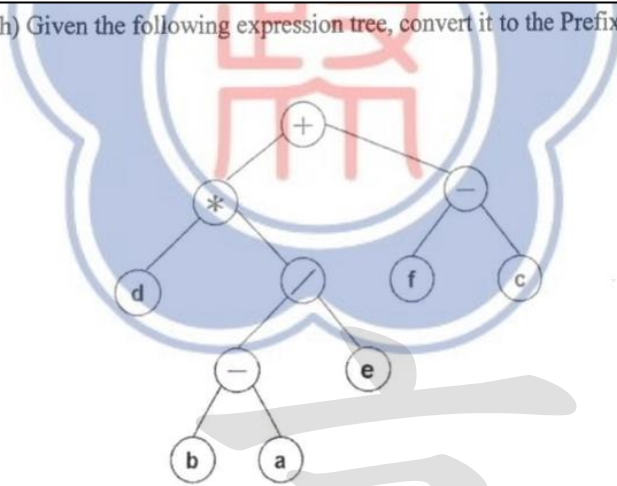
- (1) 將陣列分割直到只有一個元素。
- (2) 開始兩兩合併，每次合併同時進行排序，合併出排序過的陣列。
- (3) 重複2的動作直接全部合併完成。

Initial	26, 5, 77, 1, 61, 11, 59, 15, 48, 19
1	[5, 26], [1, 77], [11, 61], [15, 59], [19, 48]
2	[1, 5, 26, 77], [11, 15, 59, 61], [19, 48]
3	[1, 5, 11, 15, 26, 59, 61, 77], [19, 48]
4	[1, 5, 11, 15, 19, 26, 48, 59, 61, 77]

3. 分析

- 1. 時間複雜度：Merge為 $O(m+n)$ ，整個mergeSort為 $O(n\log n)$ 。
- 2. 為stable sort。

5. (10%, 5 points for each) Given the following expression tree, convert it to the Prefix and Postfix notations



【考題命中】 計算機概論第四回，P.50~P.51，觀念命中

2. 前序走訪 (Preorder Traversal)：先走訪樹根，再走訪左子樹，之後走訪右子樹。

```
void preorder(TreePtr tree) {
    if (tree != NULL) {
        printf("%d", t->data);
        preorder (t->lChild);
        preorder(t->rChild);
    }
}
```

3. 後序走訪 (Postorder Traversal)：先走訪左子樹，再走訪右子樹，最後走訪樹根。

```
void postorder(TreePtr tree) {
    if (tree != NULL) {
        postorder (t->lChild);
        postorder (t->rChild);
        printf("%d", t->data);
    }
}
```