《資料結構》

試題評析	本次地方特考題目主要包括：時間複雜度計算、運算式處理、搜尋樹、堆積、圖形等4個部分。 第一題：包括時間複雜度的觀念，以及複雜度等級的判斷，第(一)小題簡單，第(二)小題就要看如何推算與解釋。 第二題：測驗運算式與算式樹相關的題目，此題較基本，應該相對比較容易取分。 第三題：是以二元樹和二元搜尋樹為基礎，考驗考生是否能夠推廣到m-路搜尋樹。 第四題：考的是堆積結構的定義，操作與運算的時間，第(三)小題題意看似不合理，但可以拿Leftist tree的合併處理方法來用。 第五題：圖形處理屬於較基本的問題。
考點命中	1.《資料結構(重點整理)》，高點文化出版，王致强編著，頁5-36～5-37。 2.《資料結構(重點整理)》，高點文化出版，王致强編著，頁6-49～6-53。 3.《資料結構(重點整理)》，高點文化出版，王致强編著，頁11-23～11-24。 4.《資料結構(重點整理)》，高點文化出版，王致强編著，頁7-3～7-6、7-37～7-40。 5.《資料結構(重點整理)》，高點文化出版，王致强編著，頁8-8～8-9、8-30～8-32。

一、請用Big-O符號來表示下列函式的成長速率，並說明之：

(一) $T(n) = 3n^3 + 7n^3\sqrt{n} + n^3 \log n$（5分）

(二) $T(n) = 2T(n/2) + n^2$（10分）

答：

(一) $T(n) = 3n^3 + 7n^3\sqrt{n} + n^3 \log n =$

$= n^3(3 + 7\sqrt{n} + \log n)$

$\because \log n \prec \sqrt{n}$ $\therefore 3 + 7\sqrt{n} + \log n = O(\sqrt{n})$

$T(n) = n^3(3 + 7\sqrt{n} + \log n) = n^3 \times O(\sqrt{n}) = O(n^3\sqrt{n})$

(二)方法1：解遞迴關式

取 $n = 2^k$

則 $T(n) = T(2^k) = 2T(2^{k-1}) + (2^k)^2$

令 $a_k = T(2^k)$，遞迴關係式成 $a_k - a_{k-1} = 4^k$…①式

降一階可得 $a_{k-1} - a_{k-2} = 4^{k-1}$…②式

① − ②×4 得 線性齊次方程式

$a_k - 5a_{k-1} + 4a_{k-2} = 0$

特徵方程式 $r^2 - 5r + 4 = 0$

解得根為 r=4 或 1

得通解 $T(n) = a_k = C_1 \times 4^k + C_2 \times 1^k = O(4^k) = O(n^2)$

方法2：使用Master theorem判斷 $T(n) = a \times T\left(\frac{n}{b}\right) + g(n)$ 的複雜度等級

本題中 a=2, b=2, g(n)=n^2

因為 $g(n) = \Omega(n^{\log_b a + \varepsilon})$

即 $g(n) = n^2 = \Omega(n^{\log_2 2 + \varepsilon})$

g(n)=n^2 的等級遠高於 $n^{\log_2 2}$

因此 $T(n) = \Theta(g(n)) = \Theta(n^2)$

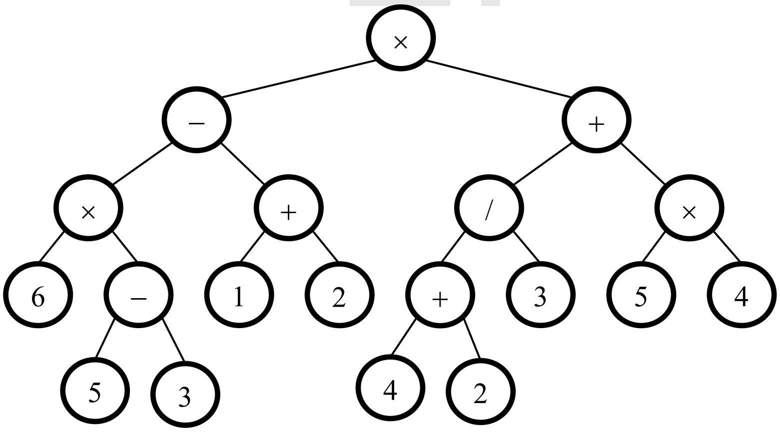
結論 $T(n) = O(n^2)$

二、常用的算術運算式（Arithmetic Expression）有：中序運算式（Infix Expression）、前序運算式（Prefix Expression）、後序運算式（Postfix Expression）三種表示法，考慮下面的算術運算式（Arithmetic Expression）並回答下列問題：

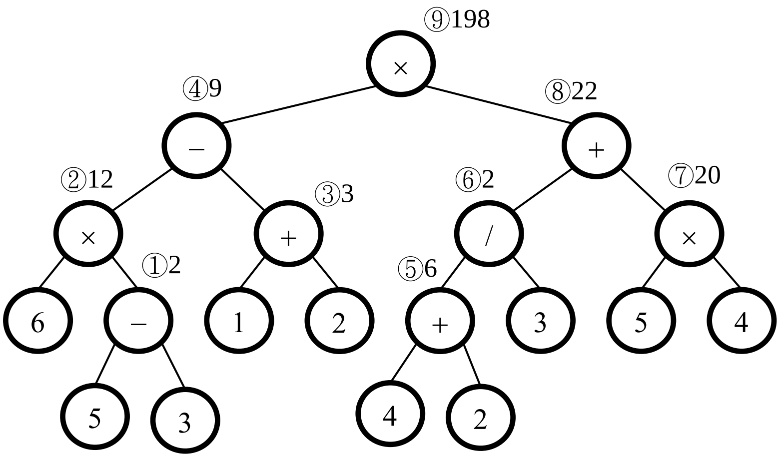
$$((6 \times (5 - 3)) - (1 + 2)) \times (((4 + 2) / 3) + (5 \times 4))$$

(一) 請寫出其前序運算式（Prefix Expression）。（5分）
(二) 請繪出其算術運算樹（Expression Tree）。（5分）
(三) 請說明如何以此算術運算樹計算出算術運算式的值，並一步一步列出運算過程。（10分）

答：
(一)前序運算式為：$- \times 6 - 5\ 3\ + 1\ 2\ + / + 4\ 2\ 3 \times 5\ 4$
(二)算術運算樹



(三)算術運算樹求值計算，以post-order順序bottom-up進行，編號代表順序，旁邊的值是結果。



三、回顧二元樹結構，其為m路樹（m-ary Trees，亦稱多元樹、m元樹）的一個特例，請回答下列相關問題：
(一) 給出m路樹的定義。（5分）

(二) 若用陣列來表示一個m路樹，請說明如何利用陣列的索引值來表示節點間的親子連結關係（意即，假設陣列索引起始值為0，若節點v在陣列的第i個位置，節點v的第c個子節點的位置為何？另一方面，節點v的parent位置為何？）？（10分）
(三) 基於此m路樹結構及二元搜尋樹（Binary Search Tree）的概念，我們可以定義出一個多元搜尋樹。當$m = 4$的時候，可以稱此搜尋樹為四元搜尋樹。請給出(2,4)-樹((2,4)-tree)的定

義並比較與四元搜尋樹的差異。（10分）

答：

(一)m-way tree定義如下：

1.m-路樹可以是一棵empty tree。

2.m-路樹若非empty tree則須符合下面條件：

(1)每個節點的degree≦m (每個節點最多可有m個subtree)，此時節點中存有m-1個資料。

(2)每個subtree也都mayo m-路樹(recursive definition)。

(二)m-way tree的一維陣列表示法：

1. root放在陣列a[0]

2.如果節點v存放在a[i]，則其第c個子節點存放在$a[i\times$m+c]，其中1≤c≤m。

3.如果節點v存放在a[i]，則其parent存放在$a[\left\lfloor\frac{i-1}{m}\right\rfloor]$。

(三)(2,4)-樹是一棵4-way tree，樹根可以是empty；如果不是empty時，可以有1～3個資料(如果是存在的狀況，以k1,k2,k3來代表)，同時會有2～4個subtrees(如果是存在的狀況，以T0,T1,T2,T3來代表)，subtrees都是4-way trees(遞迴定義)。

四元搜尋樹還要比(2,4)-樹多加一個條件，T0中的資料≤k1≤T1中的資料≤k2≤T2中的資料≤k3≤T3中的資料。

四、二元堆積（Binary Heap）是一種優先佇列（Priority Queue），主要用來管理具有優先權順序的資料物件，每個資料物件具有一個可以界定大小或前後順序的鍵值（Key），我們在此假設鍵值越低的資料物件有越高的優先權。

(一) 請完整描述最小堆積（Min_Heap）的定義與相關的操作功能。（5分）

(二) 請說明堆積排序（Heap Sort）的方法並分析其時間複雜度。（5分）

(三) 若有兩個二元樹T_1及T_2，其節點具有堆積特性且高度分別是$O(\log\ n)$與$O(\log\ m)$，請提供一個方法將此兩個二元樹結合成為一個節點具有堆積特性的二元樹T，此方法的時間須為$O(\log n + \log m)$。（10分）

答：

(一)Min Heap的定義：

(1) Min Heap結構上是一棵Complete Binary Tree。

(2) Min Heap是一棵min tree，也就是說parent≤child。

Min Heap主要提供2個operations：

(1) Insertion：可以插入一項新資料。

(2) Extract-Min：可以取出最小的一項資料。

(二)堆積排序是利用堆積結構來進行排序，分為2個階段。

(1)先將所有資料建立成Heap，時間為O(n)。

(2)再持續執行Extract-Min，直到Heap變成empty為止，就可以得到遞增的排序順序，時間為O(nlogn)。

總時間：O(n)+O(nlogn)=O(nlogn)。

(三)題目並未要求合併完成，必須回復Complete Tree結構，只要求是二元樹即可。

這樣可以採用合併的方法即可，方法如下：

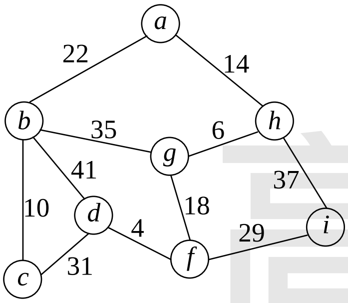
(1)比較兩個Min Heap的roots，較小的以a來代表，另一個以b代表。

(2)以a為整體的root，a的left sub-tree維持不變。

(3)將原來a的right sub-tree與b遞迴進行合併(recursively)，直到全部合併成一個Min Heap為止。

合併的時間=2個Heap的高度總和=O(logn)+O(logm)=O(logn+logm)。

五、下圖是一個加權圖$G=(V,E)$，其中V是點集合而E是邊集合。



(一) 請使用相鄰矩陣（Adjacency Matrix）表示法來表示加權圖G。（5分）
(二) 不考慮權重，從節點g開始並按照字母順序對G進行廣度優先尋訪（Breadth-First Search, BFS），請繪出尋訪完後所產生的BFS樹（BFS Tree）。（5分）
(三) 請利用Prim's演算法，從節點d起始，找出一個最小擴張樹（Minimum Spanning tree），請以圖示方式一步步畫出過程與結果，並說明Prim's演算法的時間複雜度。（10分）

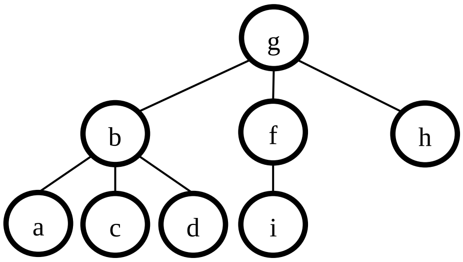
答：

(一)相鄰矩陣

	a	b	c	d	f	g	h	i
a	0	22	∞	∞	∞	∞	14	∞
b	22	0	10	41	∞	35	∞	∞
c	∞	10	0	31	∞	∞	∞	∞
d	∞	41	31	0	4	∞	∞	∞
f	∞	∞	∞	4	0	18	∞	29
g	∞	35	∞	∞	18	0	6	∞
h	14	∞	∞	∞	∞	6	0	37
i	∞	∞	∞	∞	29	∞	37	0

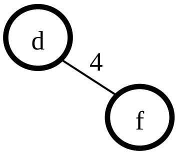
(二)從節點 g 開始進行BFS搜尋：g, b, f, h, a, c, d, i

產生的BFS Tree如下：

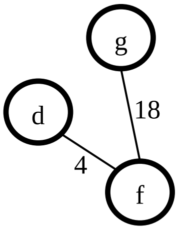


(三) Prim's演算法，從節點d起始，找出一個最小擴張樹。

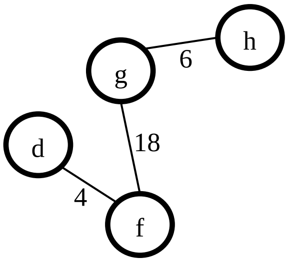
(1)



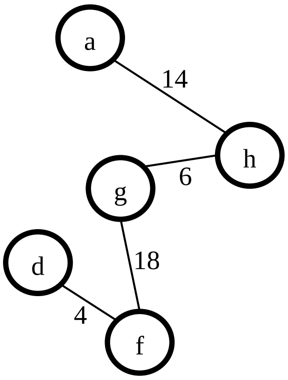
(2)



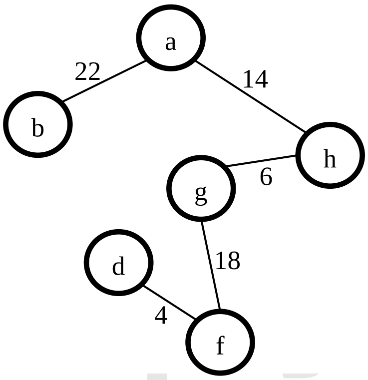
(3)



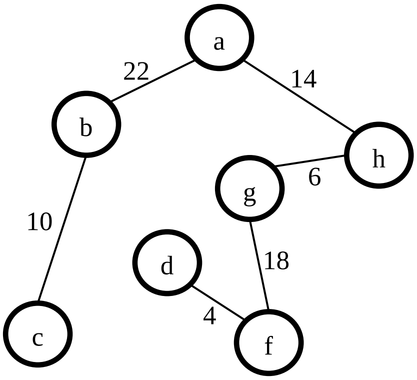
(4)



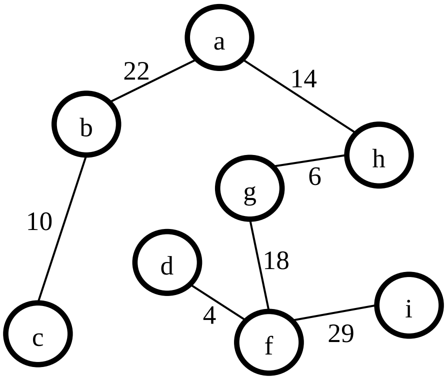
(5)



(6)



(7)



Prim’s演算法的時間，因為圖形G採用Adjacency Matrix，時間複雜度為$O(|V|^2)$。