銘傳大學 97 學年度研究所碩士班招生考試

傳播管理研究所碩士班（丙組）、資訊管理學系碩士班

資料結構試題(第二節)

（第 1 頁共 2 頁）（限用答案本作答）　☑可使用計算機　☐不可使用計算機

一、請根據下列演算法 *BS*(…)回答下列問題：

1. 請問以 *BS*(…)演算法在圖一的陣列 *A* 中搜尋鍵值(key) $k = 30$ 需要進行多少次的鍵值比較？ **(10%)**

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
A:	**5**	**6**	**9**	**11**	**15**	**17**	**19**	**21**	**22**	**24**	**25**	**26**	**27**	**28**	**30**	**33**

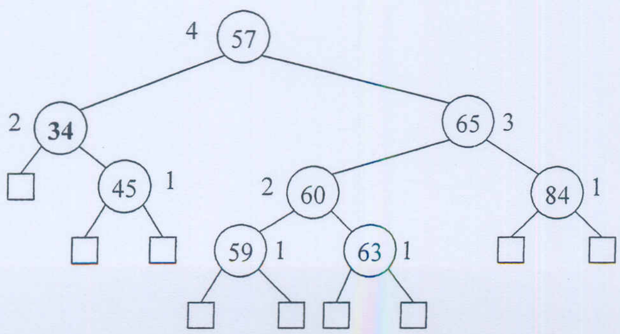
圖一：整數陣列 *A*

2. 假設陣列 *A* 的長度為 $n = 260$，請問以 *BS*(…)演算法搜尋一個鍵值最多需要進行多少次的鍵值比較？ **(15%)**
3. 請證明 *BS*(…)演算法的時間複雜度為 $O(\log n)$。(註：log 代表以 2 為底的對數) **(15%)**

```
Algorithm BS(A, k, low, high):
    Input:  An ordered array A storing n integers, a key k, and integers low and high.
    Output: An element of A equal to k and index between low and high, if such an
            element exists, and otherwise null.
    if low > high then
        return null
    else
        mid ← ⌊(low + high)/2⌋
        e ← A[mid]
        if k = e then
            return e
        else if k < e then
            return BS(A, k, low, mid − 1)
        else
            return BS(A, k, mid + 1, high)
```

二、圖二代表一棵高度平衡樹(AVL Tree) *T*，其中矩形圖示表示外部節點，圓形圖示表示內部節點，節點旁的數字代表一個節點的高度，內部節點中的數字代表鍵值(key)。請回答下列有關高度平衡樹的問題：

1. 請說明高度平衡樹的定義。 **(10%)**
2. 在高度平衡樹上刪除一個節點之後，必須作一些節點重新調整的動作，方能持續維持高度平衡樹的特性，請畫出在圖二上刪除 key = 34 這個節點之後的新的高度平衡樹。 **(10%)**
3. 請證明一棵具有 *n* 個內部節點的高度平衡樹，其樹的高度為 $O(\log n)$。(註：log 代表以 2 為底的對數) **(10%)**



本試題兩面印刷

圖二：高度平衡樹(AVL Tree) *T*

銘傳大學 97 學年度研究所碩士班招生考試

傳播管理研究所碩士班（丙組）、資訊管理學系碩士班

資料結構試題（第二節）

（第二頁共二頁）（限用答案本作答）　　☑可使用計算機　☐不可使用計算機

三、圖三代表一個加權無向圖(weighted undirected graph) G，其中圓形圖示代表頂點(vertex)，連接兩個頂點的直線代表邊(edge)，邊上所標示的數字代表權重(weight)(例如路徑長度、時間、成本等)。請回答下列有關圖的問題：

1. 假設以頂點 **A** 作爲搜尋的起始點，請依序寫出無向圖 G 的深先搜尋(Depth-First Search; DFS)結果。　**(10%)**
2. 同樣以頂點 **A** 作爲搜尋的起始點，請依序寫出無向圖 G 的廣先搜尋(Breadth-First Search; BFS)結果。　**(10%)**
3. 利用 Kruskal's Algorithm 可以用來產生最小擴張樹，其演算法以虛擬碼描述如下：

Algorithm Kruskal(G):

　　Input: A simple connected weighted graph G with n vertices and m edges

　　Output: A minimum spanning tree T

　　for each vertex v in G **do**

　　　　Define an elementary cluster $C(v) \leftarrow \{v\}$.

　　Initialize a priority queue Q to contain all edges in G, using the weights as keys.

　　$T \leftarrow \varnothing$

　　while T has fewer than $n-1$ edges **do**

　　　　$(u, v) \leftarrow Q$.removeMin()

　　　　Let $C(v)$ be the cluster containing v, and let $C(u)$ be the cluster containing u.

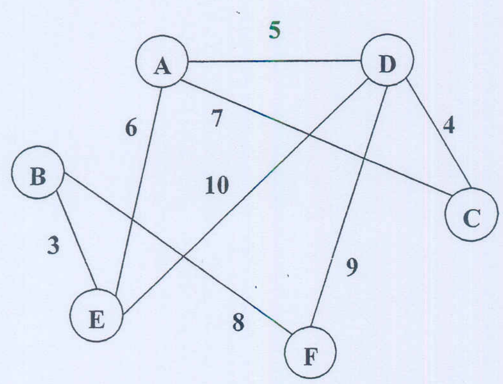
　　　　if $C(v) \neq C(u)$ **then**

　　　　　　Add edge (v, u) to T

　　　　　　Merge $C(v)$ and $C(u)$ into one cluster, that is, union $C(v)$ and $C(u)$.

　　return tree T

請以圖三中的圖形 G 爲例，畫出利用 Kruskal's Algorithm 產生的最小擴張樹。　**(10%)**



圖三：加權無向圖(weighted undirected graph) G

本試題兩面印刷　　試題完