《資料結構》

一、計算正整數a和b的最大公因數gcd(a,b)的演算法，以類似C語言表示如下：

```
1       integer gcd(a,b){
2            x=a;y=b;
3            while (y>0){r=x%y;x=y;y=r;}
4            return x;
5       }
```

其中資料型態integer表示整數，x%y表示x除以y的餘數。請回答下列問題：（每小題10分，共20分）

(一)請證明：輸入任意兩個正整數，此程式執行一定時間後就會停止，不會造成無窮迴圈。

(二)假設a>b，請證明此程式之while迴圈（第3行）至多只會被執行$2\log_2 b+1$次。

試題評析	本題為Euclid's Algorithm的程式，分析其數值計算上的特性，同時計算迴圈的執行次數上限。此題屬於稍微冷門之題材，分析中需要觀察到餘數與被除數、除數之間的關係，才可能推論並證明出完整的結果。
考點命中	《資料結構》，高點文化出版，王致強編著，頁5-9。

答：

(一) while (y>0) { r=x%y; x=y; y=r; }

計算做法：被除數x，除數為y，餘數為r，迴圈每回之後餘數被作為次數的除數。

因為 餘數<除數，因此每一輪回的y值會比前一回更小，最後 出現y==0時，迴圈一定會結束。

(二) 假設，最初的$x_0=a$，$y_0=b$，迴圈1回之後的x值與y值為x_1與y_1，迴圈2回之後的x值與y值為x_2與y_2，……以此類推。可以有下面：

第1回之後 $x_1=y_0$，$y_1= x_0 \bmod y_0$

第2回之後 $x_2=y_1$，$y_2= x_1 \bmod y_1$

……..

第i-2回之後 $y_{i-2}= x_{i-3} \bmod y_{i-3}$，$x_{i-2}=y_{i-3}$

第i-1回之後 $y_{i-1}= x_{i-2} \bmod y_{i-2}$，$x_{i-1}=y_{i-2}$

第 i 回之後 $y_i= x_{i-1} \bmod y_{i-1}$，$x_i=y_{i-1}$

所以 $y_i= x_{i-1} \bmod y_{i-1}=y_{i-2} \bmod y_{i-1}$

根據 (1) 餘數<除數的原理

(2) 當被除數>除數時，餘數<(被除數/2)，分兩種情形說明：

(a) 除數≤(被除數/2)時，餘數<除數≤ (被除數/2)，故餘數<(被除數/2)。

(b) 除數>(被除數/2)時，餘數=被除數-除數，故也是餘數<(被除數/2)。

因此，餘數y_i<被除數$y_{i-2}/2$，若迴圈共執行k次，$y_k=1\le y_{k-2}/2^1\le y_{k-4}/2^2\le\ldots\le y_0/2^{k/2}=b/2^{k/2}$

故$1\le b/2^{k/2}$，$2^{k/2}\le b$，$k/2\le\log_2 b$，$k\le 2\log_2 b$。最後，若a<b，迴圈第1次會先將a與b，故再多加1次，於是最多$2\log_2 b+1$次。

二、給定一個權重圖（weighted graph），G=(V,E,w)，假設V={1,2,...,n}，且每個邊（edge）e的權重w(e)都是正整數。令l(v)為以v為端點的所有邊中權重最小的邊。將這些邊集合起來稱作L，也就是$L=\bigcup_{v\in V} l(v)$。（每小題5分，共20分）

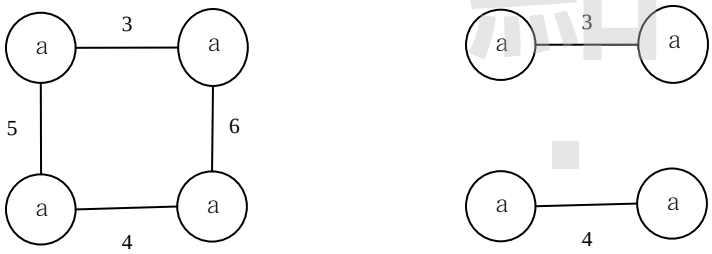
(一)假設每個邊的權重都不相同。請證明由L中這些邊所構成的子圖（edge induced subgraph）G[L]沒有迴圈。

(二)G[L]是否一定是G的擴張樹（spanning tree）？若是請證明之，若不一定是請給一個反例。

(三)用以上之結論，設計一個計算G的最小權重擴張樹（minimum spanning tree）的演算法。
(四)在一般的應用中，邊的權重可能會相同，請修正上述之演算法，使修正後之演算法可以正確找出答案。

試題評析	本題為Solilin’s Algorithm的相關問題，(一)為圖論基本特性證明，可用Contradiction證明、(二)為其與擴張樹的關聯、(三)即為Sollin’s Algorithm、(四)為變化狀況之處理。本題原本圖論證明部份，不熟悉證明的考生，會有一點不適應；若未看出此題是Sollin’s algorithm，則取分會更少。
考點命中	1. 《資料結構》，高點文化出版，王致強編著，頁8-45。 2. 《資料結構》，高點文化出版，王致強編著，第8章精選例題35，頁8-49。

答：
(一)使用反證法(Contradiction)，假設所構成的子圖仍會有迴圈，則迴圈上權重最大的邊(u,v)，(u,v)不可能是端點u最小的邊，亦不可能是端點v最小的邊，故不可能構成迴圈。
(二)一定是沒有迴圈，但未必是連通子圖，故不一定是擴張樹。反例如下：左圖為G，右圖為G[L]。



(三)使用Sollin’s algorithm
Minimum spanning tree T←ϕ;
每個頂點可視為一棵樹，整個圖形視為一個forest;
while forest不只有一棵樹 do begin
由每一棵樹選出權重最小的邊，加入T;
forest包含的樹變少;
end;
最後T中的邊即為Minimum Spanning Tree
(四)由每一棵樹選出權重最小的邊，加入T時，若發現有迴圈，則將迴圈中最大的邊剔除。

三、假設陣列A[1..n]儲存n個正整數x1, x2, . . . , x_n。（每小題10分，共20分）
(一)已知所有的正整數 $x_i \le M$。請設計一個O(n+M)時間的演算法將這些整數由小到大排列。
(二)已知所有的正整數 $x_i \le n^2$。請設計一個O(n)時間的演算法將這些整數由小到大排列，或證明這是不可行的。

試題評析	本題為Sorting問題，採用時間複雜度做為線索，找出符合要求的排序法，兩個小題考的都是分佈式排序法。
考點命中	《資料結構》，高點文化出版，王致強編著，頁5-9。

答：
(一)使用分佈式計數排序法
for i←1 to M do count[i]←0; // O(M)
for i←1 to n do count[A[i]]←count[A[i]]+1; // O(n)
start[1]←1;

```
for i←2 to M do start[i]←start[i-1]+count[i-1];   // O(M)
for i←1 to n do begin                              // O(n)
    B[start[A[i]]]←A[i];
    start[A[i]]<start[A[i]]+1;
end
```

總時間：O(n+M)

(二) 使用RSD Bucket Sort

取radix=n，準備n個buckets(n個FIFO queues, $Q_0,Q_1,Q_2,\cdots,Q_{n-1}$)

每個key的格式為$key_i(d_2d_1d_0)$radix n

```
for k←0 to 2 do begin
    for i←1 to n do
        j←keyi的dk
        enqueue(Qj, keyi);
    end;
    for j←0 to n-1 do
        將Qj資料依序取出;
end;
```

四、假設有個陣列A[1..n]儲存著n個整數。可將A[1..n]看成二元樹，其中A[1]是樹根。A[i]的左右子節點分別為A[2i]和A[2i+1], i=1,2,...,n/2。若2i>n或2i+1>n，則這些子節點是不存在的。若A滿足A[i]≥max{A[2i],A[2i+1]},1≤i≤n/2，則稱陣列A[1..n]是一個堆疊（heap）。假設有個副程式sift(A,r,n)其輸入參數A是一個陣列，n是A的大小，r≤n是一個指標，指向此子樹的樹根。副程式sift(A,r,n)的功能是將A[r]為樹根的子樹變成heap。在呼叫sift(A,r,n)之前，它的左右子樹都已經是heap。副程式sift(A,r,n)所需的計算時間是O(h(r))，其中h(r)是以A[r]為樹根的子樹的高度，也就是從樹根到任一樹葉的最長距離。（每小題10分，共20分）

(一)用sift(A,r,n)設計一個線性時間的演算法，將陣列A[1..n]變成heap。

(二)分析以上所設計演算法的計算複雜度為O(n)。

試題評析	本題為建立Max-Heap的方法，可以採用Heap sort中的第一階段來完成，所需時間正好為O(n)。
考點命中	《資料結構》，高點文化出版，王致強編著，頁9-45。

答：

(一)
```
for r←n/2 downto 1 do
    sift(A,r,n);
```

(二) 在worst case下，sift的總時間為

$T(n)=(k-1)*1+(k-2)*2^1+(k-3)*2^2+\cdots+1*2^{k-2}$ ……(1)

(1)*2得

$T(n)*2=(k-1)*2+(k-2)*2^2+(k-3)*2^3+\cdots+1*2^{k-1}$ ……(2)

(2)-(1)

$T(n)=-k+1+2+2^2+\cdots+2^{k-1}=n-k=O(n)$

五、斐波納契數（Fibonacci number）F_n的定義是$F_0=0, F_1=1, F_n=F_{n-1}+F_{n-2}, n>1$。

計算Fibonacci number F_n的演算法，以類似C語言表示如下：

```
1     integer f[N];    //  array of N integers
2     integer F(n){
3          if (f[n]<0)
4              f[n]=F(n-1)+F(n-2);
```

```
5          return f[n];
6   }
7      integer Fib(n){
8          f[0]=0;f[1]=1;
9          for (i=2; i≤n; i=i+1)
10             f[i]=-1;
11         return F(n);
12  }
```

其中資料型態integer表示整數。假設輸入的整數n>1。主程式執行Fib(n)，則副程式F(n)第4行之指令：

f[n]=F(n-1)+F(n-2)會被執行幾次？請說明理由。（20分）

試題評析	本題關於為Fibonacci的遞迴程式呼叫次數計算，程式的寫法表面似乎是遞迴，但其中利用到動態程式規劃(dynamic programming)技巧，利用已經計算過的項目，不需要再計算一次，來減少呼叫次數。
考點命中	《資料結構》，高點文化出版，王致強編著，頁5-29，精選例題8。

答：採用遞迴樹來表示其呼叫關係：先往左側一路遞迴下去，而同一個項目只會計算1次，故計算次數=n-1次。