《資料結構》

一、請設計演算法複製一棵二元樹（copy a binary tree）。（10分）

試題評析	本題屬於二元樹的基本處理，通常樹的處理，在設計上，大部份使用遞迴方法較為便捷。
考點命中	《資料結構》，高點文化出版，王致強編著，頁6-23，精選範例23。

答：

```
1.  treePointer copy(treePointer root) {
2.      treePointer t;
3.      if (root!=NULL) {
4.          t = (treePointer)malloc(sizeof(struct node));
5.          t->data = root->data;
6.          t->left = copy(root->left);
7.          t->right = copy(root->right);
8.          return t;
9.      }
10.     else return NULL;
11. }
```

二、(一)請描述 order 為 m 的 B-tree 之特性。（6分）

(二)請問 order 為 m 高度為 h 的 B-tree：(1)最多有幾個節點？最多有幾個 Key？（6分）

(2)最少有幾個節點？最少有幾個 Key？（8分）

試題評析	本題第一小題考B-tree的特性，第二小題則計算節點個數的上限與下限，以及key的上、下限，屬於基本的問題，考生取分不難，上下限部份未必要依賴記憶，只要觀念清楚可以當場再推導一次即可。
考點命中	《資料結構》，高點文化出版，王致強編著，頁11-24～11-25，要點：B-樹的定義。

答：

(一)

(1) B-tree 是一棵 m-way search tree。

(2) B-tree可以是空的樹，如果不是空的就必須滿足下面3點：

(i)root 至少要有二個 children。

(ii)除了root以外，其它 node 至少有$\left\lceil \frac{m}{2} \right\rceil$個 children。

(iii)所有 failure (terminal)node 高度皆相等(balanced)。

(二)假設 B-樹的 order 為 m，高度為 h(失敗節點的高度為 h+1)。

節點最多有 $1+m+m^2+\cdots+m^{h-1}=\frac{m^h-1}{m-1}$ 個

最多可存關鍵值 $(m-1)+m(m-1)+m^2(m-1)+\ldots+m^{h-1}(m-1)$ Z

$=(m-1)\times\sum_{i=0}^{h-1}m^i=(m-1)\times\frac{m^h-1}{m-1}=m^h-1$ 個。

節點最少有$1+\left\lceil \frac{m}{2} \right\rceil+\left\lceil \frac{m}{2} \right\rceil^2+\cdots+\left\lceil \frac{m}{2} \right\rceil^{h-1}=\frac{\left\lceil \frac{m}{2} \right\rceil^h-1}{\left\lceil \frac{m}{2} \right\rceil-1}$

最少可存關鍵值 $1+2\left(\left\lceil \frac{m}{2} \right\rceil-1\right)+2\left\lceil \frac{m}{2} \right\rceil\left(\left\lceil \frac{m}{2} \right\rceil-1\right)+2\left\lceil \frac{m}{2} \right\rceil^2\left(\left\lceil \frac{m}{2} \right\rceil-1\right)+\cdots+2\left\lceil \frac{m}{2} \right\rceil^{h-2}\left(\left\lceil \frac{m}{2} \right\rceil-1\right)$

$=1+2(\left\lceil \frac{m}{2} \right\rceil-1)\sum_{i=0}^{h-2}\left\lceil \frac{m}{2} \right\rceil^i=1+2(\left\lceil \frac{m}{2} \right\rceil^{h-1}-1)=2\left\lceil \frac{m}{2} \right\rceil^{h-1}-1$ 個。

三、請利用 Double Hashing 將下列 key 值放入 hash table of size 13中（如表1）：（14分）

{24, 53, 17, 46, 14, 32, 37, 92}
$h_1(k)=k \bmod 13$，$h_2(k)=1+(k \bmod 11)$，
$h(k,i)=(h_1(k)+i*h_2(k)) \bmod 13$ $(i=0, 1, \cdots, 12)$

表1

0	1	2	3	4	5	6	7	8	9	10	11	12

試題評析	本題考double hashing 的雜湊表，h1()計算home value，h2()計算collision時的增量，方法不難，按照標準方法，取分應不難。
考點命中	《資料結構》，高點文化出版，王致強編著，頁10-23～10-24，精選範例14。

答：

key	24	53	17	46	14	32	37	92
h1(k)	11	1	4	7	1	6	11	1
h2(k)	3	10	7	3	4	11	5	5
probing	11▲	1▲	4▲	7▲	1 5▲	6▲	11 3▲	1 6 11 3 8▲

註: ▲為最後存入的slot

最終的Hash Table如下:

0	1	2	3	4	5	6	7	8	9	10	11	12
	53		37	17	14	32	46	92			24	

四、(一)在一棵高度為 h(h=0,1,2,…)的 AVL tree 中：(1)高度為6之 AVL tree 最多可能有幾個 nodes？最少可能有幾個 nodes？（假設 root 之 h=0）（6分）(2)假設此樹共有45個 nodes。請問此 AVL tree 可能最高之高度及最矮之高度各為何？（6分）

(二)請將下列數字{17,60,24,5,7}逐步插入圖1的 AVL tree 中，並平衡之。（12分）

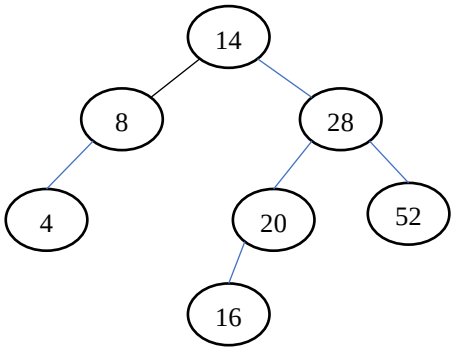


圖1

試題評析	本題為節點個數範圍的計算，但須注意root的高度h=0，後半段測驗AVL tree插入資料後，進行rotation來平衡AVL tree，是屬於AVL tree基本操作的問題。
考點命中	《資料結構》，高點文化出版，王致強編著，頁11-16～11-19，精選範例7、精選範例8。

答：

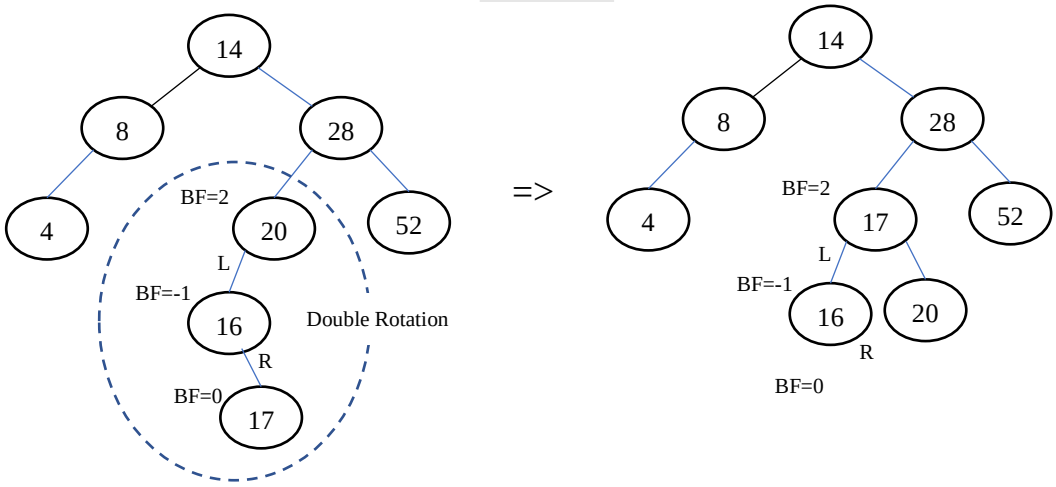
(一)高度為 h 高度(h=0,1,2,…, 假設 root 之h=0)，節點最多是Full binary tree，有$2^{h+1}-1$個節點；

節點最少是Most-Skewed AVL tree有$F_{h+3}-1$個節點，$F_{h+3}-1=\frac{1}{\sqrt{5}}\left(\phi^{h+3}-\hat{\phi}^{h+3}\right)-1$，其中$\phi=\frac{1+\sqrt{5}}{2}$，而$\hat{\phi}=\frac{1-\sqrt{5}}{2}$，費氏數列$F_0$=0,$F_1$=1,$F_2$=1,$F_3$=2,$F_4$=3,$F_5$=5,$F_6$=8,$F_7$=13,$F_8$=21,$F_9$=34,…

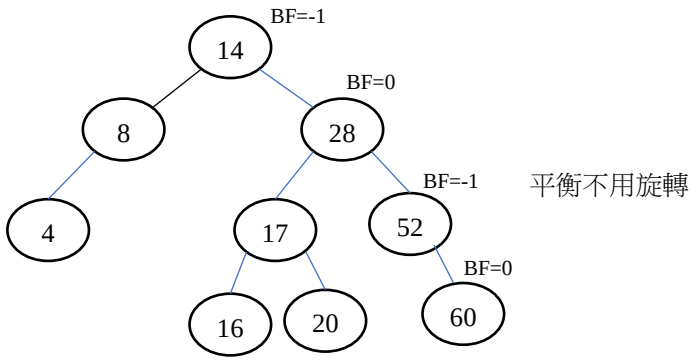
故高度h=6的AVL tree最多會有2^{h+1}-1=2^7-1=127個節點；最少會有F_{h+3}-1=F_9-1=34-1=33個節點。

(二)

(1) 插入17

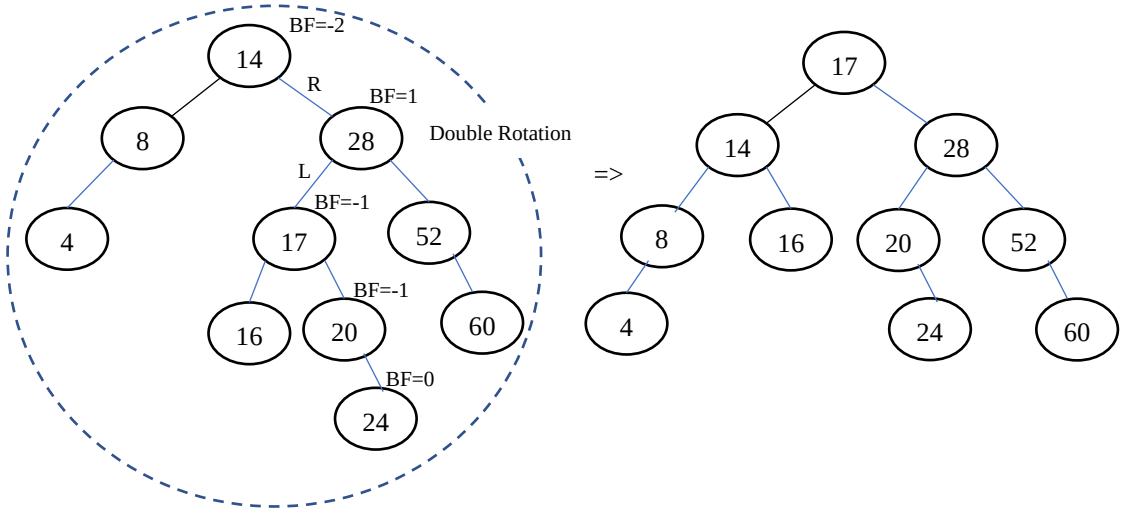


(2) 插入60

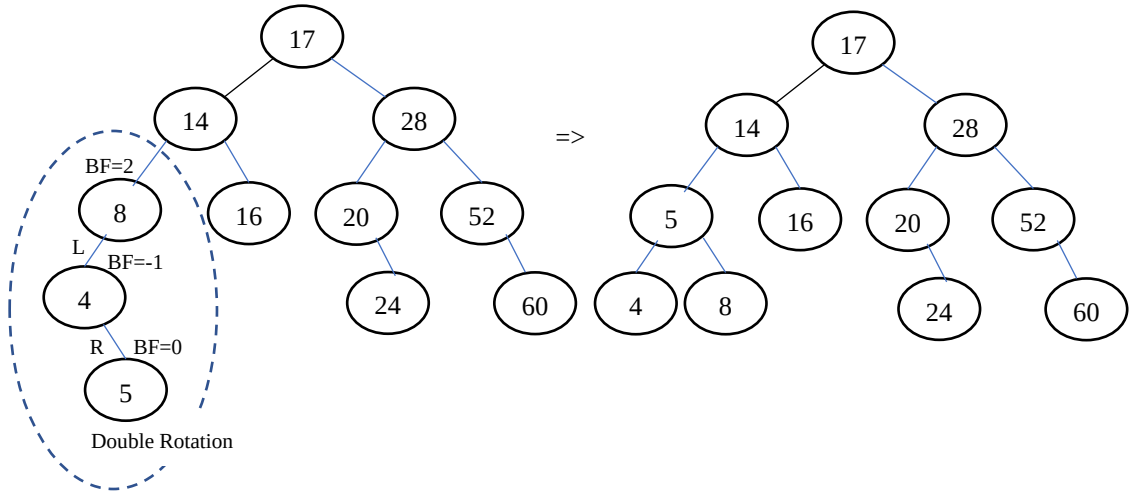


平衡不用旋轉

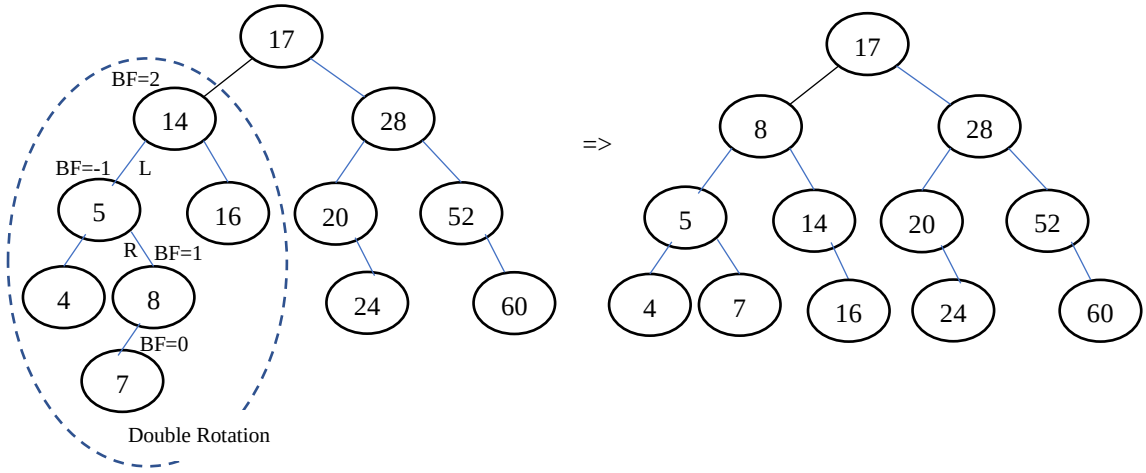
(3) 插入24


BF=-2
14
R
BF=1
8
28
Double Rotation
L
BF=-1
4
17
52
BF=-1
16
20
60
BF=0
24
=>
17
14
28
8
16
20
52
4
24
60

(4) 插入5


17
14
28
BF=2
8
16
20
52
L
BF=-1
4
24
60
R
BF=0
5
Double Rotation
=>
17
14
28
5
16
20
52
4
8
24
60

(5) 插入7


17
BF=2
14
28
BF=-1
L
5
16
20
52
R
BF=1
4
8
24
60
BF=0
7
Double Rotation
=>
17
8
28
5
14
20
52
4
7
16
24
60

五、請利用堆積排序法（Heap Sort）將圖 2 逐步建立成 Min Heap，並將數字從小到大逐一列舉。（10 分）

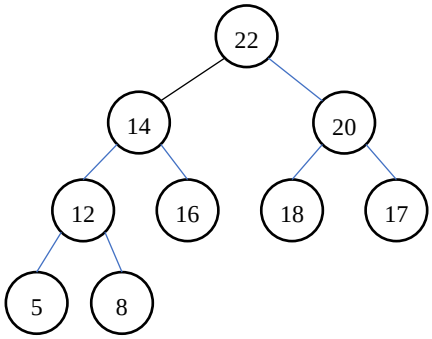


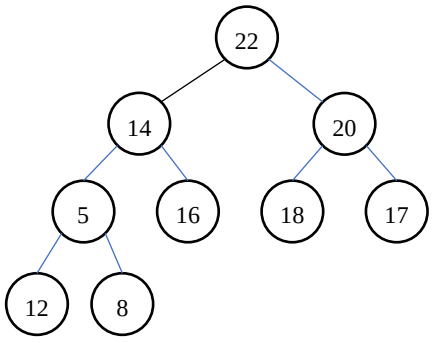
圖2

試題評析	本題為Heap的試題，先運用Heap Sort第一階段，使用 Sift 方法將樹逐漸調整為Min Heap，再將 Min Heap 資料以 Extract-Min，由小而大逐一取出列舉。
考點命中	《資料結構》，高點文化出版，王致強編著，頁9-51～9-55，堆積排序；頁7-5～7-6，要點：刪除最大元素運算。

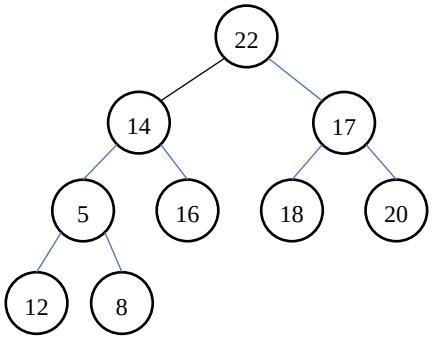
答：

(1)由最後一個內部節點逐一進行 sift(Heapify) 處理到樹根為止，就可以將二元樹調整 Min-Heap。

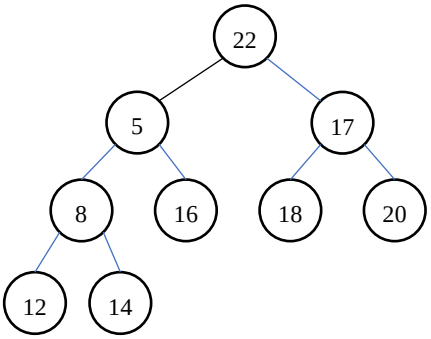
①由 12 做 Heapify



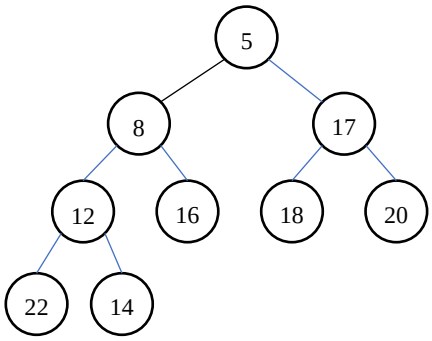
②由 20 做 Heapify



③由 14 做 Heapify

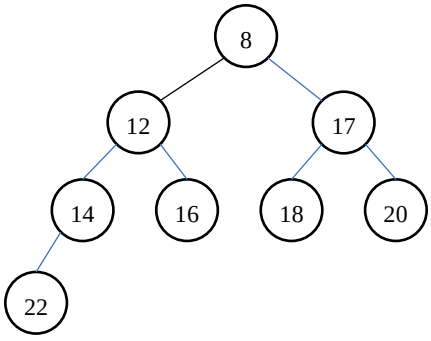


④由 22(root) 做 Heapify

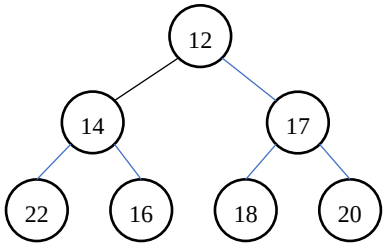


(2)要由小而大逐一列舉，可以持續做Extract-Min，直到Min-Heap完全變empty為止。

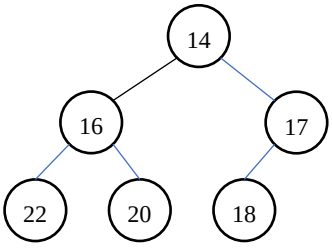
① Extract-Min 取出 5，Min-Heap 如下



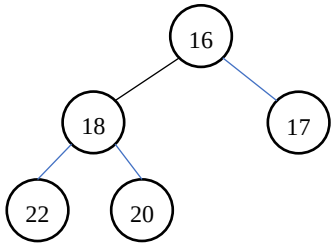
② Extract-Min 取出 8，Min-Heap 如下



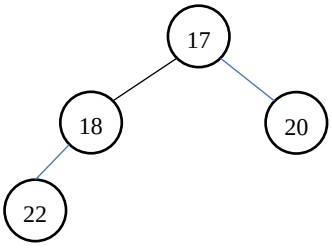
③ Extract-Min 取出 12，Min-Heap 如下



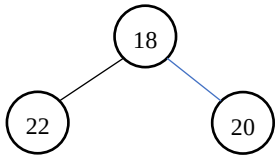
④ Extract-Min 取出 14，Min-Heap 如下



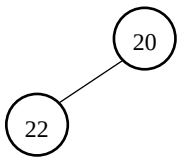
⑤Extract-Min取出16，Min-Heap如下



⑥Extract-Min取出17，Min-Heap如下



⑦Extract-Min取出18，Min-Heap如下



⑧Extract-Min取出20，Min-Heap如下

⑨Extract-Min取出22，Min-Heap如下

empty Min-Heap

六、(一)請利用KMP（Knuth, Morris, Pratt）演算法寫出失敗函數（failure function)之定義。（4分）

(二)找出 pattern “abcdabcabcdabcdabc” 之失敗函數（failure function）值（請填入表2 failure value 中）。（14分）

(三)假設(二)之 pattern 嘗試在 string “abcdabcabcdabcabcda.....” 找出 pattern。當 pattern 從 index 0開始比對到 index 13都一樣，而在 index 14時發現字母不一樣，請問 pattern 如何利用 failure function 所得之結果很快找到下一個要對應之位置？也就是 pattern 的那一位置的值要位移到 string 的那一對應位置。（4分）

表2

index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
string	a	b	c	d	a	b	c	a	b	c	d	a	b	c	a	b	c	d	a
pattern	a	b	c	d	a	b	c	a	b	c	d	a	b	c	d	a	b	c	
failure value	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	

試題評析	本題為Pattern Matching的問題，第一小題為失敗函數的定義，但要注意pattern的index由0開始；第二小題計算出失敗函數；第三小題考運用失敗函數在比對時，如何移動pattern。
考點命中	《資料結構》，高點文化出版，王致強編著，頁3-59～3-60，要點：字型比對。

答：

(一)失敗函數（failure function）之定義

如果$p=p_0p_1p_2\cdots p_{n-1}$ 是一個 pattern，則 failure function f 的定義如下:

$$f(j)=\begin{cases}\text{滿足}p_0p_1\cdots p_k=p_{j-k}p_{j-k+1}\cdots p_j\text{的最大 k 值}\ ,\ \text{其中}0\le k<j\ \text{若 k 存在的話}\\ -1\ \ ,\text{otherwise}\end{cases}$$

(二)

index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
string	a	b	c	d	a	b	c	a	b	c	d	a	b	c	a	b	c	d	a
pattern	a	b	c	d	a	b	c	a	b	c	d	a	b	c	d	a	b	c	
failure function	-1	-1	-1	-1	0	1	2	0	1	2	3	4	5	6	3	4	5	6	

(三)

取最後一個相同的失敗函數值 f(13)=6，就是讓 p_6 移到對準 s_{13}，然後繼續從 p_7 與 s_{14} 開始往後比對下去。如下圖 $p_0\cdots p_6$ 不須再比對，從 p_7 繼續進行比對即可。

index	6	7	8	9	10	11	12	13	14	15	16	17	18								
string	c	a	b	c	d	a	b	c	a	b	c	d	a								
pattern		a	b	c	d	a	b	c	a	b	c	d	a	b	c	d	a	b	c		
failure function		-1	-1	-1	-1	0	1	2	0	1	2	3	4	5	6	3	4	5	6		
		… 不需要再比對							從此處開始比對下去…												