1.

(a) p.201 下面第二個definition

(b) A binary search tree is a binary tree . It may be empty. If it is

not empty then it satisfies the following properties:

(1)Each nodes has exactly one key and keys in the tree are distinct.

(2)The keys in the left subtree are smaller than the key in the root.

(3)The keys in the right subtree are larger than the key in the root.

(4)The left and right subtrees are also binary search tree.

(c) FIFO : first in first out ex: queue

LIFO : last in first out ex: stack

(d) p.186\_\_A linked list that each node has two link fields, one linking the forward direction and another linking in the backward direction, and a data field.

(e)

(f)

(g) p.22\_\_Criteria of performance evaluation can be divided into two fields: and one of the field is performance analysis, which estimates the space and time complecity that are machine independent.

(h) p.205\_\_It's an idea of visiting each node in the tree exactly once. A full traversal produces a linear order for the nodes in a tree.

(i)

(j)

2.

(a) T 和課本p.41 1.(e) 一樣

　0^3 + 1^3 + 2^3 + ... +n^3 = (n^4 + 2n^3 +n^2)/4

依課本定理1.4得證 1^3 + ... +n^3 = Θ(n^4)

(b) T 和課本p.41 1.(i) 一樣 可是不知道怎麼證明= =||

(c) F 和課本p.41 2.(c) 一樣 應該是 原式=Θ(n^2)/(logn) 吧??

(d) T 和課本p.41 1.(b) 一樣

3.

int A[5][6][10];

(a) 2000 + (2\*60 + 3\*10 + 7) \*4 = 2628 // integer 是 4 bits

　　 → address : 2628

(b) 2300-2000=300

300/4=75

75/60=1...15

15/10=1...5

→ element : A[1][1][5]

(c) 2000 + [ (1-3)\*60 + 5\*10 + 9 ]\*4 = 1756

→ address : 1756

4. p.201

5.

P . 179~186

(a)

(b)

6.

J

╱ ╲

E I

╱ ╲ ╲

B D H

╱ ╲ 　 ╱

A C G

╲

F

( inorder:LVR postorder:LRV )

7.

(i) ABD\*-EF/+AD\*+C+

(ii) AB-D\*EFAD\*+/+C+

9.

(a) 如果要刪掉一個node，就要把他的值用"在左邊分支的最大數"或是

"在右邊分支的最小數"取代.

如果是leaf當然就直接刪掉，因為他沒有分支

complexity = O(h) h是tree的高度(深度)

(b)

10.

(a) Circular queue by using an array

假設共有 MAX 個空間，為了辨別是否full或empty，必須留一個空位

初始：front = MAX-1 rear = MAX-1

加入：rear = (rear+1) % MAX

刪除：front = (front+1) % MAX

full：front==(rear+1) % MAX

empty：front == rear

(b) Doubly linked circular list

　　　 tail->next = head

　　　 ←←←←←←←

　　　↓口→口→口→口↑

　　　 →→→→→→→

head->prev = tail

(c) 給兩個STACK應用的例子

1. Railroad switching network p.112

2. A mazing problem

3. Tower of Honoi Problem

4. The 8 Queens Problem

5. 副程式的呼叫和返回

6. 運算式的轉換