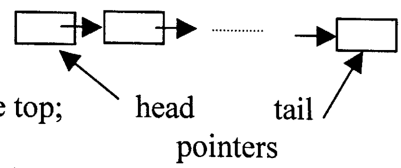
Data Structures

Midterm Exam, Spring 2003 (YEN)

1. (4 pts) Given examples to illustrate the relationship between ***abstract data types*** and ***data structures***?

2. (8 pts) A *Stack* can be implemented by an *array* or a *singly linked list* (with a head pointer and a tail pointer as shown on the right) in the following way:



1. by an array with the beginning of the array as the top;
2. by an array with the end of the array as the top;
3. by a linked list with the beginning of the linked list as the top;
4. by a linked list with the end of the linked list as the top;

What is the worst-case time complexity (in terms of $\theta()$) of the push and pop operations in each of the above four implementations? Fill in the following table: (寫在答案卷上)

implementation	push	pop
1		
2		
3		
4		

3. (4 pts) Characterize, using the big-Oh notation, the worst-case running time of the following algorithm. Why? Let A be a given array of n integers. ('← ' denotes assignment.)

```
for i ← 0 to n-1 do
   for j ← 0 to (i*i)-1 do
     Let A[j mod n] ← j.
   end for
end for
```

4. (12 pts) Draw all *minimal-height binary search trees* which store the following numbers:

```
4   8   16   32   64
```

5. (10 pts) Define a *funny-tree* to be a binary tree such that for each of its nodes *x*, the number of nodes in each sub-tree of *x* is at most 2/3 the number of the nodes in the tree rooted at *x*.

1. Draw the *tallest* funny tree of 5 nodes.
2. What is the height of an *n*-node funny-tree? Write your answer in $\theta()$ notation. Explain why briefly.

6. (10 pts) Draw the binary tree that has the following traversal sequences:

Inorder: C, B, D, A, F, G, E, H, I

Postorder: C, D, B, G, F, I, H, E, A

7. (10 pts) Let T_1 and T_2 be two arbitrary binary trees, each having n (unlabeled) nodes. Show that it is sufficient to apply 2(n-1) single rotations to T_1 so that it becomes T_2.

8. (10 pts) Give an *AVL tree* (try to give the smallest) for which the deletion of a node requires two *double rotations*. Drawing the tree, and explain why two rotations are needed.

9. (10 pts) Show the result of inserting a 35 into the *splay tree* shown in Figure 1 below. (Don't forget to perform the appropriate splay.)

10. (10 pts) Let us consider a *red-black* tree in Figure 2. How the tree has to be modified after deleting node with key 27? With respect to Figure 2 again, how to modify the tree if a node with 5 is to be inserted?

11. (12 pts) Hashing.

1. (3 points) Draw the *linear-probing hash table* that results when you insert the keys

   6 16 8 5 13 1

   in that order into an initially empty table. Use a table of size 11 and the hash function $h(x) = 3x + 4 \pmod{11}$.
2. (9 pts) What is the **worst-case** asymptotic complexity for performing n simple operations (i.e. insert or query) on the following initially empty hash tables? Also briefly explain when does each of the worst-cases happens.

   (i) a hash table where collisions are resolved by chaining into the front of a linked list

   (ii) a hash table where collisions are resolved by chaining into a (not necessarily balanced) binary search tree

   (iii) a hash table of size $2n$ with open addressing and linear probing

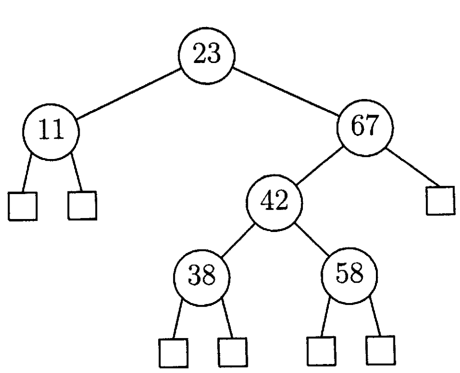


Figure 1

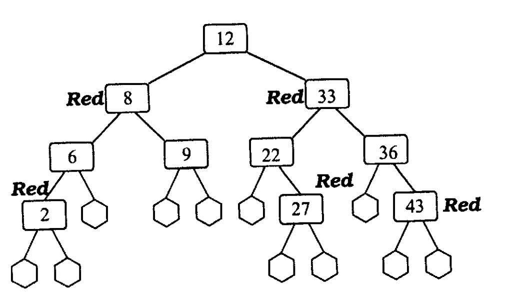


Figure 2