Data Structure Midterm

2018.11.6

Close-book exam. Lecturer: Chih-Ya Shen (沈之涯)

Please answer the following questions in the answer sheet 4 pages (12 questions) on 2 pieces of papers

---

1. (14%, 2% each) True or False. No explanation is needed.

A. The growth rate of n^5 is larger than 1.001^n for any n>1.

B. If f(n) = O(g(n)), we can say that g(n) ≥ f(n) for n>1.

C. If f(n) = O(n^2), we can also say that f(n) = O(n^3).

D.

```
int foo(int n)
   if(n<1) return 1
   else return n*foo(n-1)
```

Function **foo** calculates the value of n! if n is a positive integer. The time complexity of **foo** is θ(n!).

E.

```
int bs(int *A, const int x, const int n)
int left=0, right=n-1
   while (left <= right)
      int middle = (left+right)/2
      if (x < A[middle]) right = middle-1
      else if (x > A[middle]) left = middle+1
      else return middle
   end while
   return -1 // not found
```

Function **bs** finds the index of a number x in a sorted array A of length n (in ascending order). The time complexity of **bs** is O(log n).

F. Finding an element in the binary search tree takes O(log n) time.

G. Suppose we have numbers between 1 and 1000 in a binary search tree and we want to search for the number 363. The sequence [925, 202, 913, 240, 921, 245, 363] is **impossible** to be the sequence of nodes visited in the search.

2. (6%) Consider a binary tree T whose preorder and inorder traversal sequences are as follows.

Preorder: F,B,A,D,C,E,G,I,H.

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

A. Please show the postorder traversal sequence of T. (3%)

B. Please show the level-order traversal sequence of T. (3%)

3. (6%) Consider a sequence of keys: 9,4,8,7,12,15,3,5,14,18.

A. Please draw the result after inserting all these keys into an empty min heap. (Result only) (3%)

B. Following A, please draw the result after deleting the root. (3%)

4. (8%) Consider a sequence of keys 5,19,23,13,7,17,3,2,11. Please draw the result after inserting all these keys into an empty AVL tree. (Result only)

5. (6%) Suppose there is a singly linked list without any duplicate numbers. The node (87) is in the list and the node (40) is the next node of (87). Now we want to insert a new node (94) between (87) and (40). We decide to call the function **InsertAfter**(94,87), and the pseudo codes are listed below.

```
void LinkedList::InsertAfter(int data,int data_ref){
    ListNode *newNode = new ListNode(data);
    if(first==NULL)
        return;
    ListNode *current = first;
    while (current!= NULL) {
        if(current->data==data_ref){
            current->next = newNode;
            newNode->next = current->next;
            return;
        }
        current = current->next;
    }
}
```

Something goes wrong after the function completes. We find that the next node of (87) is (94), but the next node of (94) is not (40).

Can you explain what causes this error (2%) and how it can be corrected? (4%)

Note that the simpler the way you correct it, the higher the score you'll obtain.

6. (8%) In a queue, elements can be pushed to one side and popped from the other. Different from queues, a "*deque*" allows elements to be pushed and popped from both sides. Now, we push 1,2,3,4,5,6,7 sequentially into a deque, and each element can be popped if there is no other element blocking its way out. Is the sequence [5,1,7,4,2,3,6] a possible pop-sequence? Please explain why or why not?

7. (6%) A complete binary tree B containing 100 nodes (with indices 1, 2, …, 100) is stored in an one-dimension array. Let node i be in position i of the array (array[0] is empty).

A. What is the height of tree B? (2%)

B. What is the index of **the parent** of array[71]? (2%)

C. How many nodes are leaf nodes? (2%)

8. (6%) A rooted tree is called an n-ary tree if every internal node has no more than n children, while a complete tree is a tree in which all the levels are full except for the bottom level and the bottom level is filled from left to right. Consider a complete n-ary tree T that has 64 leaf nodes, and T's height is 4. Please list **all** the possible values of n.

9. (12%) An m*n Young tableau is an m*n matrix such that the entries of all rows are in a sorted order from left to right and the entries of each column are in a sorted order from top to bottom. Some of the entries of a Young tableau may be ∞, which we treat as non-existing elements. Thus, a Young tableau can be used to hold r <= m*n finite numbers. Two examples of Young tableau are shown below.

2	3	4	5
8	9	12	14
16	∞	∞	∞
∞	∞	∞	∞

2	3	4	5
3	4	5	6
6	87	∞	∞
∞	∞	∞	∞

A. Given a nonempty Young tableau, we define the operation EXTRACT-MIN as removing the minimum-value number from the table. If there are multiple minimum-value numbers, remove any one (and only one) of them. Please design an algorithm to implement EXTRACT-MIN on a nonempty m*n Young tableau that runs in O(m+n) time. Please also provide the time-complexity analysis. (Hint: Think about how to do pop in a heap.) (6%)

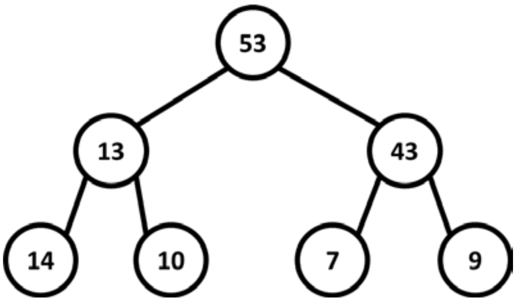
B. Show how to insert a new element into a non-full m*n Young tableau in O(m+n) time. Please also provide the time-complexity analysis. (6%)

10.(8%) Please design a O(n)-time recursive function that calculates the minimum difference between any two ***adjacent*** nodes in a tree.

Definition of function is: int **MinDiff**(Node *root)

Definition of a tree node is:

```
class Node
{
    int data;
    Node *leftNode;
    Node *rightNode;
}
```



For example, in the above tree, the minimum difference is 1 (from 14 and 13). Please also analyze the time complexity of your algorithm.

11. (10%) Please design a O(nlogk)-time algorithm to merge k sorted lists into one sorted list, where n (n>k) is the total number of elements in all the input lists. Please analyze the time complexity of your algorithm. (Hint: Use a min heap for k-way merging)

12. (10%) Please prove that given the postorder and inorder traversal sequences of a binary tree T, we can uniquely reconstruct T. (Hint: prove with contradiction)