CS235101 Data Structure
Midterm Exam

1. **[12%] General Concepts**
   (a) [2%] It takes ~~O(1)~~ to find the node with min key in a BST, True or False?
   (b) [2%] Suppose a heap is constructed from n integers, what is the time complexity (in Big-O) of inserting a new integer?
   (c) [2%] The recursive function calls are implemented using ~~Heap~~ structure, True or False?
   (d) [2%] BST is always a complete binary tree, True or False?
   (e) [2%] Given a binary tree with k levels, what is the maximum number of tree nodes?
   (f) [2%] Deleting the element with min key from a min heap takes ~~O(1)~~, True or False?

2. **[10%] Performance Analysis**
   (a) [2%] Please define Big-O notation
   (b) [3%] Prove that if $f_1(n) = O(g_1(n))$ and $f_2(n) = O(g_2(n))$, then $f_1(n)f_2(n) = O(g_1(n)g_2(n))$.
   (c) [2%] Please sort the following time complexity expressions in ascending order:
   1. $O(n^2)$; 2. $O(n)$; 3. $O(n\log_2 n)$; 4. $O(\log_2 n)$; 5. $O(2^n)$.
   (d) [3%] Please write the time complexity of function F.
   PS: You have to write the tightest O(•).

```
int F(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;
    }
    return -1;
}
```

3. [10%] Stack/Queue

Given an infix expression $(a + b)/c * (d - e)$

(a) [2%] What is the corresponding prefix expression?

(b) [2%] What is the corresponding postfix expression?

(c) [6%] Please draw the detailed steps of evaluating the prefix expression in (a) using a stack.

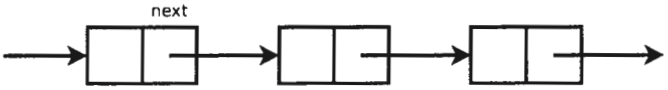
4. [10%] Linked List

Please answer the questions based on the following data structures:

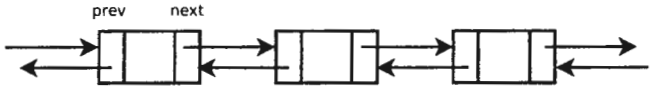
```
//Singly-linked list node
class Node
{
public:
    int data;
    Node* next;
}
```

```
//Doubly-linked list node
class Node
{
public:
    int data;
    Node* prev;
    Node* next;
}
```

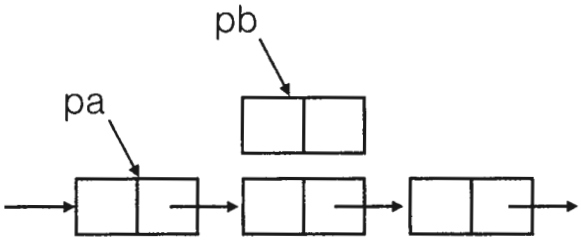
A singly-linked list is illustrated as below:



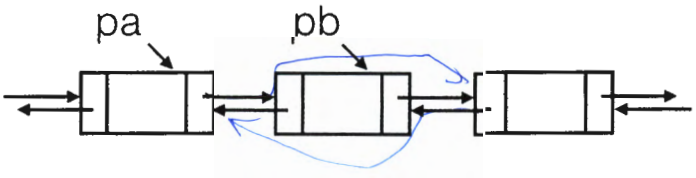
A doubly-linked list is illustrated as below:



(a) [5%] The following diagram shows parts of a **singly-linked list**. Please write pseudo codes (C/C++) to insert a new node **pb**, such that **pb** is next to **pa.**



(b) [5%] The following diagram shows parts of a **doubly-linked list**. Please write pseudo codes (C/C++) to remove the node **pb** that follows the node **pa**.

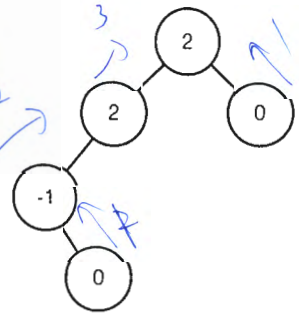


5. [20%] Binary Tree

(a) [10%] Please answer the following question based on the binary tree structure as below:

```
class Node
{
public:
    int bfactor;
    Node *left;
    Node *right;
};
```

The balance-factor of a binary tree is the difference in heights of its two sub-trees, i.e. the height of the left sub-tree minus the height of the right sub-tree. An example can be found on the right inset where the balance factor is labeled in the center of each node. Please write pseudo codes (C/C++) for the following function that calculates the balance factor (bfactor) of each node.




```
int calcBalanceFactors( Node *root );
```

(b) [5%] The **pre-order** and **in-order** traversal of a tree is ABDECFHGI and EDBAHFCGI respectively. Please draw the tree, and explain whether or not the tree exists uniquely.

(c) [5%] The **pre-order** and **post-order** traversal of a tree is ABDECFHGI and EDBHFIGCA respectively. Please draw the tree, and explain whether or not the tree exists uniquely.

6. [18%] Heap

Given an input sequence as 8, 3, 2, 11, 12, 6, 10, 1, 5, 9, 7, 4.

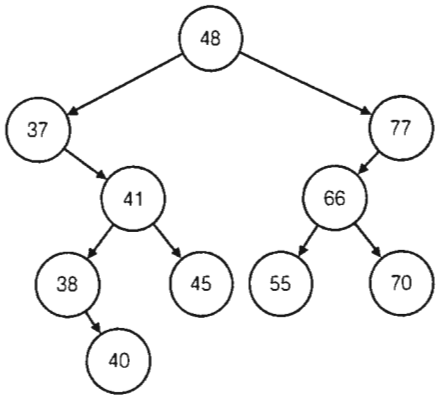
(a) [7%] Please construct the corresponding min heap, and draw the detailed steps when inserting 4.

(b) [7%] Please draw the steps of deleting the min key.

(c) [4%] Please prove that the time complexity is $O(n \log_2 n)$ if we use min heap to sort the sequence.

7. [20%] BST

Given the following binary search tree **T**, please answer the questions.



(a) [5%] Draw all possible BSTs after deleting the node with key 48 from **T**.

(b) [5%] Draw all possible BSTs after inserting a node with key 60 to **T**.

(c) [5%] An indexed binary search tree is a BST with each node containing an additional data field **leftSize**, which is one plus the number of nodes in the left sub-tree. Please explicitly label the **leftSize** of each node in **T**.

(d) [5%] Draw the process of deleting the **fourth** smallest element from **T** (You must explicitly label the leftSize at each step).