CSE 674 – Midterm Exam
Spring 2015

Name: _____

SU Email: _____

Question	Points	Score
1	15	
2	15	
3	15	
4	15	
5	15	
6	15	
7	15	
8	15	
Total	120	

Instructions for the exam:

1. Date: 3/19/2015, in class
2. Closed book. No communications among students are allowed.

3. You may bring in one page of notes (letter size). You may write on both sides. Please submit your notes together with this examination packet.
4. Answer all 8 questions. Read the instructions given in the question carefully. Your presentation counts. In many cases, you may be asked to show steps in order to receive credit.

Instructions for the coding questions:

When completing the coding questions, you may freely create new functions without declaring them. You do not have to worry about the order that the functions are written in.

Please write clearly. If you need to use additional space or you need to cross out code, please indicate which page the actual code is on.

Integrity Pledge:

I pledge that I will not give nor receive help on this exam from any other student or any electronic device. I pledge that the only help I am taking is from one 8.5”x11” sheet of paper. I understand that if I give or receive help in any way whatsoever, I can fail the course.

Printed Name:

_____

Signature:

_____

Question 1. (15 point) Given a doubly linked list with a head and tail pointer, swap the elements in the following pattern: If the input list is {0, 1, 2} then the output should be {2, 1, 0}. Assume that the list has a size that is a positive multiple of 3. Make sure that you fix the head and tail pointers. As in your homework 1, you must change pointers (cannot just move the keys). You must use the API given for full credit. Do not use any STL containers or algorithms.

```
01  class ListNode {
02  public:
03      int key;
04      ListNode * prev;
05      ListNode * next;
06  };
07
08  class DoubleList {
09  public:
10      ListNode * head;
11      ListNode * tail;
12      void swapThree();
13  }
14
15  void DoubleList::swapThree(){
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
```

```
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81 }
```

Question 2. (15 point)

1. (3 point) State the worst case time complexity of heap sort using Big- Θ notation

2. (12 point) Show the intermediate steps (tree diagrams) of building a heap (max-heap) when the initial data array is [5, 4, 2, 1, 3, 7, 6, 10]. You must draw all the intermediate steps (a sequence of trees)

Question 3.

Algorithm analysis (Meaning of Big-O, Big-Θ and Big-Ω]

Let f(n) be the worst case time complexity of an algorithm, where n is the input size of the algorithm. In your own words, explain the meaning of the following notations concisely. You may use diagrams to help your illustrations:

1. (5 point) $f(n) = O(n^3)$

2. (5 point) $f(n) = \Omega(n \lg n)$

3. 3. (5 point) $f(n) = \Theta(1)$

Question 4. (15 point)
Complete the function Rprint given below. The function Rprint will print the keys stored in a doubly linked list in reverse order. Note that this doubly linked list does not have a tail pointer and your algorithm must only use constant amount of extra space. It implies that you cannot use a stack or recursion. You must use the API given for full credit. Do not use any STL containers or algorithms.

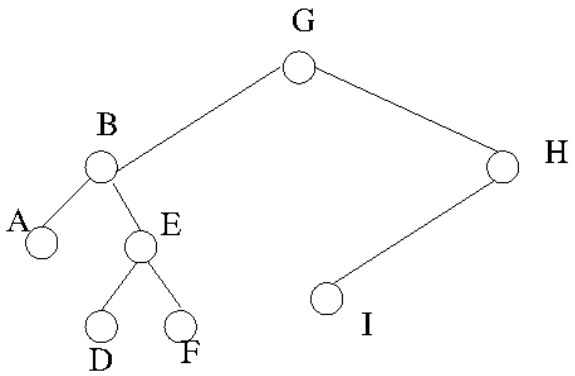
```
01  class ListNode {
02      int key;
03      ListNode * next;
04      ListNode * prev;
05  };
06
07  void Rprint(ListNode * head){
08
09
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
```

```
}
```

Question 5. (15 point)

1. (5 point) State the criteria an AVL tree use to verify if it is balanced. That is, there is no need to perform any forms of rotations to re-balance the tree.

2. (10 point) Suppose that the keys of an AVL tree are arranged in alphabetical order. Suppose that a node with key **C** is being inserted into the following AVL tree



Determine if rebalancing is needed. If so, indicate at which node rotation are needed and draw the resulting tree after inserting the node and rebalance operations are performed.

Question 6

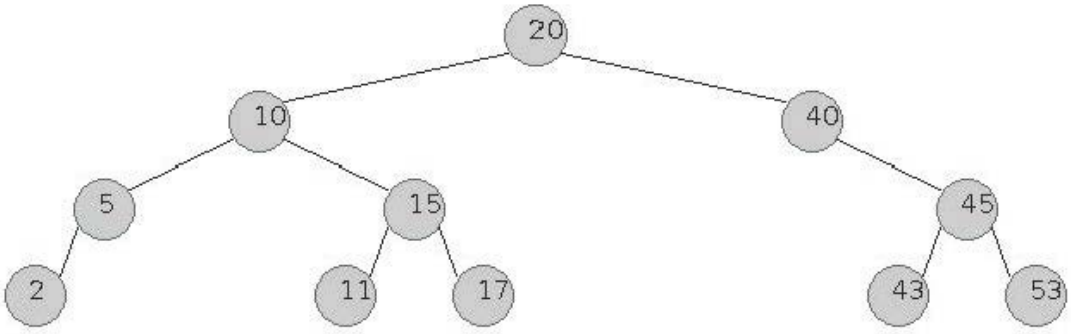
1. (6 point) B-trees, B*-trees and B+ trees are multiway trees commonly used in database applications when one likes to keep the trees in the hard disk. Answer the following True/False Questions by circling your choice. No written explanation(s) is required. Illegible or unclear answers will not be graded.

   a. In B-trees, all nodes (except the root) are required to be 2/3rds full, rather than half.
   True/False

   b. In B+-trees, all leaves are on the same level.
   True/False

   c. In B* Trees, the leaves contain all the keys in the tree (not just the leave keys) and the leaves have links forming a linked list.
   True/False

2. Give the pre-order and post listing of the nodes for the following tree:



Pre-order listing (5 point): Post-order listing (4 point):

Question 7

1. If we have a binary search tree with n nodes. State, in Big-O notation:

    a. (2 point) The best case running time for searching a key:

    b. (2 point) The worst case running time for searching a key:

2. (6 point) Assume that we start with an empty tree. Draw the binary search tree T resulted from inserting the following keys in the given order:

    10, 25, 20, 40, 30, 35, 32, 43

    T:

3. (5 point) Draw the resulting tree if we delete the node with key = 25 from T:

Question 8

1. (5 point) In class, we introduce a vector as an array that can automatically grow or shrink. Explain why inserting an element at the back of the vector has O(1) amortized time.

2. (5 point) Explain why quicksort worst case running time can be improved to O(n lg n).

3. (5 point) When will we say a sorting method is stable? When will we say a sorting method is in place?