CITY UNIVERSITY OF HONG KONG

Course code & title	:	CS3334 Data Structures
Session	:	Semester B 2019/20
Time allowed	:	Two hours

This paper has NINE pages (including this cover page).

1. This paper consists of 12 questions in 2 sections.
2. Answer ALL questions in Section A and in Section B.

*This is an **open-book** examination.*

Candidates are allowed to use the following materials/aids:

Approved Calculator, lecture slides, tutorials

Candidates will be subject to disciplinary action if any unauthorized materials or aids are found on them.

Section A (64%)

1. (10 pt) Explain whether the following statements are correct or not:

i) The running time of Merge sort and Heap sort are both O(nlogn).

ii) In a binary search tree, the smallest key can be stored in an internal node.

iii) In a hash table of prime size, using quadratic probing can always find an empty slot for the new key as long as the hash table is not full.

iv) In any data structure, linked implementation is always better than array implementation.

v) In a 4*4 maze generated by using stacks, no matter which way is chosen when selecting new rooms, the number of walls broken will always be 15.

2. (6 pt) Do Big-Oh analysis for the following function:

$T(n)=n^3 \log n+100000n^4$

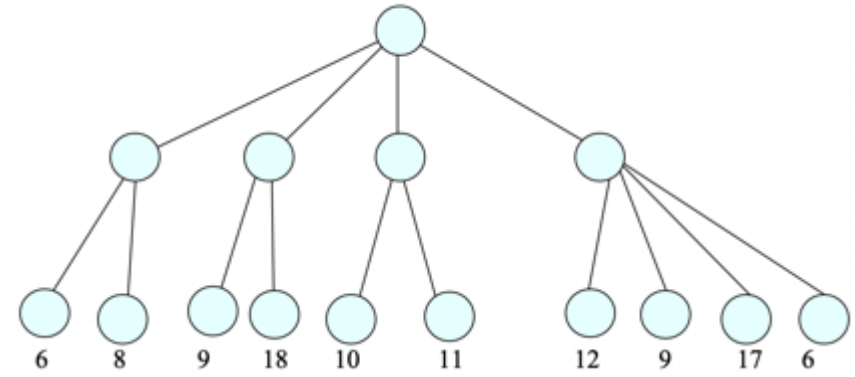
$T(n)=n^{3000}+2^n$

3. (4 pt) Given the following traversal sequences, reconstruct the corresponding binary tree:

Postorder sequence: CBDEFGA

Inorder sequence: CBADGEF

4. (4 pt) **i)** You are given the following game tree where player 1 takes the first move and the total amount of 25 dollars will be divided between player 1 and player 2 after the game ends. The value specified for a certain leaf is the amount of dollars player 2 can get if the game process follows the path from root to that leaf. Please decide the best move for player 1 as the first step using the minimax rule. Does this tree allow you to cut some branches using α - β pruning if you search the children from left to right?



(4 pt). **ii)** Insert 3,1,2,8,4,9 in sequence into a binary search tree step by step. (Show the detailed steps for each insertion). Then delete 3 from the above tree.

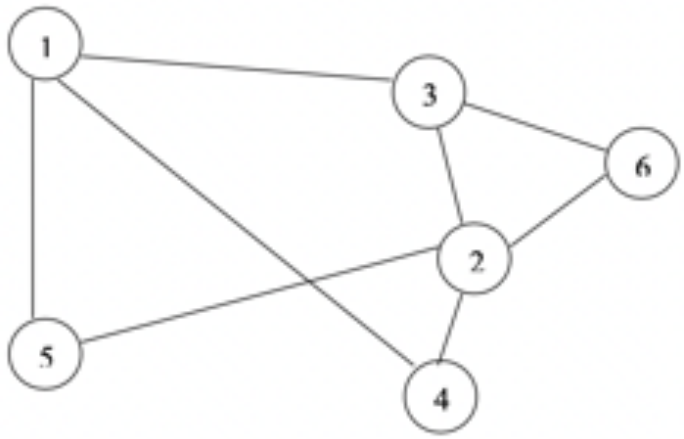
5. (7pt) **i)** Insert the following keys step by step into a hash table of size 7 using quadratic probing with rehashing: 16, 5, 30, 2, 88, 53, 69. The hash function used is key%tablesize. You need to show the status of the hash table after each insertion (When doing rehash, we do it from left to right within the array).

(3pt) **ii)** Explain why we use the above method (quadratic probing with rehashing) to do insertion in hash tables.

6. (10 pt) When generating the convex hull for the following point set, please write down the steps about how those points are pushed into or popped out from the stack.

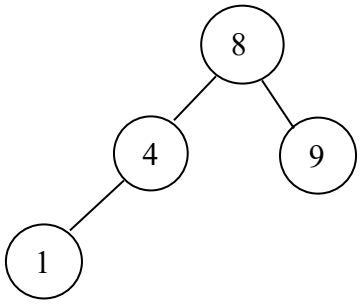
ww

7. (6 pt) Do DFS and BFS search (starting from node 1) on the following graph.



8. (6 pt) We have an array of size 7 with the following numbers: 5, 7, 1, 6, 9, 8, 4. Show step by step how the array changes when we use heap sort to sort them (you are only allowed to use the original array with these 7 elements, with only one slot of extra memory storage provided for swapping purpose). You can use a tree structure to represent contents in the array.

9. (4 pt) Please specify what kind of rotation needs to be done when inserting 10 into the following AVL tree.



Section B (36%)

10. (12pt) You are given three sorted (increasing order) singly linked list A, B and C. Write a function to output all the elements that appear in all the three lists A, B and C. For example, if A contains 1,2,3,4,5, B contains 3,5,7, and C contains 3,6, then you should output 3. Suppose the size of A, B and C are m, n and k respectively, your program should run in time O(m+n+k). You are also required to write the class definition of **List**.

OutputCommon(List* A, List* B, List* C){}

11. (12pt) Design the binary tree class (not binary search tree). Write a member function **int Tree::MaxInternal( )** which returns the largest value stored in all the internal nodes of the binary tree (All the values stored in the tree are positive integers). You should use the recursive method to write your program.

12. (12pt)

(a) Prove that using Union by Height, the height of any tree obtained in a Disjoint Set data structure is O(log n).

(b) Prove that in a hash table with load factor x with no deletion, using linear probing, the expected running time of function Search(int) is O(1) (hint: you can assume that every slot in the hash table has the same probability x of holding an integer).