TCSS 342 Spring 2017

Assignment 4
Due February 23

1. [**Binary Tree** , 20%] Consider a binary search tree that contains n nodes with keys $1, 2, 3, \cdots, n$. The shape of the tree depends on the order in which the keys have been inserted in the tree.

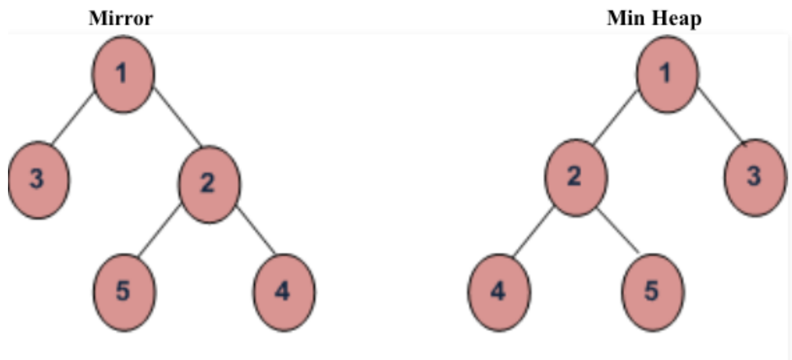
   a In what order should the keys be inserted into the binary search tree to obtain a tree with minimal height ? [5%]

   b On the tree obtained in a, what would be the worst-case running time of a find, insert, or remove operation? Use the big-Oh notation. [5%]

   c In what order should the keys be inserted into the binary search tree to obtain a tree with maximal height? [5%]

   d On the tree obtained in c, what would be the worst-case running time of a find, insert, or remove operation? Use the big-Oh notation. [5%]

2. **[Heap**, 20pt]. Add to the Heap class (minimum heap) provided to you in assignment 3 a method that takes a heap and outputs its mirror image. For example;

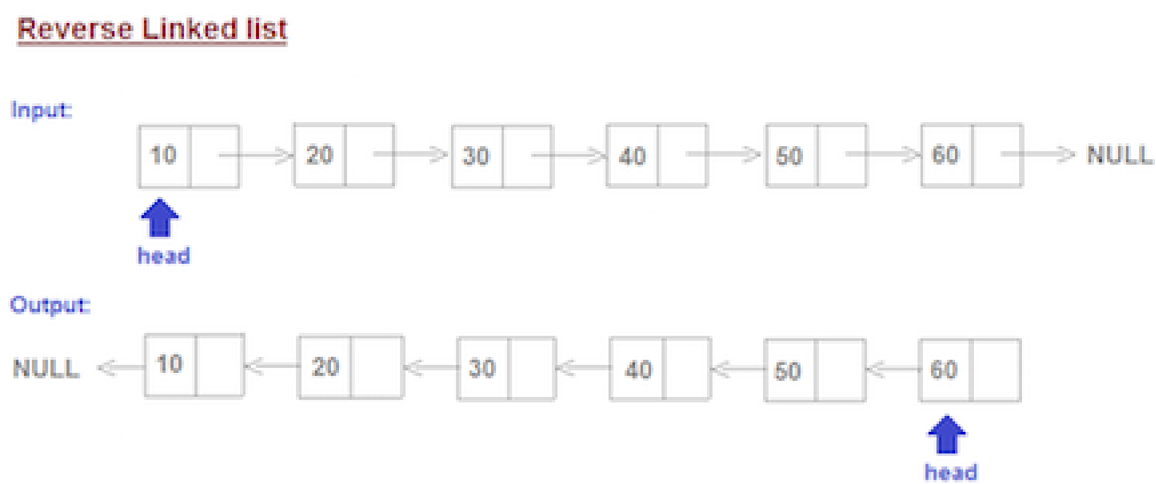


Your method should be called mirrorHeap(Heap h).

- Test your code with example above. Print out the heap and its mirror. Note you can take the content of each node the string "mirror".

[Hint]: It is easier to do this question iteratively.

3. [**LinkedList**, 20 pts]. For the given single linked list implementation, add a method that takes a linkedlist as an input and reverse it. Your method should be called public: public SingleLinkedList(SingleLinkedList L). If list is empty then you should throw an EmptyListException. Your method is allowed to use $O(1)$ a additional space.



4. [**BST**, 40 pts]. The purpose of this questions is to observe the behavior of binary search trees when inserting keys (integers). You have already given a code for binary search tree (see zip file BST).

a Add the function insert(int k) to the BST class. [**10%**]

b Add a function height(Node root) to the BST class. This function will compute the height of the BST. [**5%**]

c Add a function function remove(int k) to the BST class [**10%**]

d Let n denote the number of nodes. Construct binary search trees for $n = 10$, $n = 100$, $n = 500$, $n = 1000$, $n = 2000$, $n = 5000$, $n = 10000$, $n = 100000$, $n = 1$million. For each n you will pick uniformly random keys in the range $[-2^{31}, 2^{31} - 1]$. For each n repeat the experiment several time and calculate the average height of the tree for each n. [**10%**]

e Compare the average height to $\log_2(n+1) - 1$ for each n. Calculate constants that relate the average height to $\log_2(n+1) - 1$ for each n. Is there any relationship betweens constants for each n. [**5%**]

5. **[Worst AVL Tree**, Bonus 25%]. We saw in class that the height h with minimal number of nodes (denote as $N(h)$) is given by the Fibonacci recurrence

$$N(h+1) = N(h) + N(h-1) + 1, \text{ with } N(0) = 1 \text{ and } N(1) = 2.$$

Show that the height of an AVL tree in the worst-case is $O(\log N(h))$. Hint: Consider the close form of the Fibonacci recurrence

$$F(k) = \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^k - \frac{1}{\sqrt{5}}\left(\frac{1-\sqrt{5}}{2}\right)^k$$