1. Insert the given keys one by one in the given order into a BST which is originally empty. What will be the shape of the resulting tree?
14 24 5 13 1 3 22 10 19 11 12

2. Delete the keys 14, 24, 5, 13 in this order from the BST created in 1. What will be the shape of the resulting tree? If necessary consider substituting key X by the biggest key in the left subtree of X.

3. We are given keys 1, 2, 3, ..., *n*. Value of *n* is odd. BST is originally empty. First, we insert all even keys in to the BST in ascending order. je liché. Next, we insert into the BST all odd keys.
A. What will be the depth of the resulting BST?
B. Suppose that the odd numbers are inserted into the BST in random order. What will be the depth of the BST in this case?

4. Write a code or a pseudocode of a function which will return a reference to the node with the smallest key value in a given BST.

5. Write a code or a pseudocode of a function which will reverse the order of the keys in a BST. Reversed order means that the inorder traversal of the reversed tree will produce a sequence which is equal to the reversed sequence produced by the inorder traversal of the original tree.

6. Write a code or a pseudocode of a function which will check if the two given BST's are clones of each other, i.e. if they share exactly the same structure and same keys on the same places in the tree.

7. Write a function which will assign to a node X in the BST a reference to the node with closest highest key value in the BST. Node X will be the input parameter of the function.

8. Write a function which will assign to each node in the BST a reference to the node with closest highest key value in the BST. The root of the tree will be the input parameter of the function. The time complexity of the function should be proportional to the number of nodes in the BST.

9. Describe an algorithm which will merge two BST's T1 and T2. During the merging process no node should be deleted and no new node should be created. Merging must be done solely with the help of manipulating the left right and parent references in the tree nodes. Assess the asymptotic complexity of this algorithm.

10A. Move all three light frogs to the rocks on the right and move all three dark frogs to the rocks on the left. Frogs cannot move backwards and can hop to the neighbouring free rock or over one frog at a time. There cannot be two or more frogs on one rock at the same time. (http://akidsheart.com/math/mathgames/leapfrog.htm)

10B. Describe the algorithm which will produce the correct sequence of frog leaps for any positive number of light and dark frogs. Suppose that the initial configuration contains the free rock in the middle.

10C. Describe an algorithm which will attempt to find the correct sequence of frog leaps for any initial positiions of the light and dark frogs. Suppose that light ones travel to the right and the dark ones travel to the left. Also suppose that there may be more free stones than just one.

11A. You have a 3 liter jug and a 5 liter jug . The jugs have no measurement lines on them either. How could you measure exactly 4 liter using only those jugs and as much extra water as you need?

11B. You have an X liter jug and an Y liter jug. Describe the algorithm producing the sequence of pouring particular volumes of water between the jugs which will result in exact volume of Z liters in one of the jugs. As before, there are no volume marks on either of the jugs.