1. Given the implementation of the binary search tree presented in class, what is the best order to insert the following numbers {0, 1, 2, 3, 4, 5, 6, 7, 8, 9} so that the tree has minimal height. Show the tree that would be created if they were added in that order.

The best order to insert the given numbers is: {5, 2, 7, 1, 3, 6, 8, 0, 4, 5, 9}

5

/ \

2 7

/ \ / \

1 3 6 8

/ \ / \

0 4 5 9

1. (a) Show the result of inserting 2, 1, 4, 5, 9, 3, 6, 7 into an initially empty Search Tree.

2

/ \

1 4

\ \

5 9

/ /

3 6

\

7

(b) What is the output of an in-order traversal of your tree obtained in (a).

1, 3,5,2,4,6,7,9

(c) What is the output of a preorder traversal of your tree obtained in (a).

2, 1,5,3,4,9,6,7

(d) What is the output of a post-order traversal of your tree obtained in (a).

3, 5,1,7,6,9,4,2

1. (a) Show the result of inserting 2, 1, 4, 5, 9, 3, 6, 7 into an initially empty Red-Black Tree as described in class. Include the color of each node.

2-B

/ \

1-B 5-R

/ \

4-B 7-B

/ / \

3-R 6-R 9-R

(b) What is the output of an in-order traversal of your tree obtained in (a).

1,2,3,5,6,7,9

(c) What is the output of a preorder traversal of your tree obtained in (a).

2, 1,5,4,3,7,6,9

(d) What is the output of a post-order traversal of your tree obtained in (a).

1, 3,4,6,9,7,5,2

1. Do a post order traversal of expression tree (by hand) and evaluate.

\*

/ \

+ 3

/ \

4 -

/ \

5 6

4 5 6 - + 3 \*

1. Given the following binary search tree:

31

/ \

12 53

/ \ / \  
4 25 46 67

Remove 31 from the tree using the algorithm from class.

If the node has two children, the node is replaced with the smallest item in right subtree.

/ \

12 53

/ \ / \  
4 25 46 67

46

/ \

12 53

/ \ \

4 25 67

template <class Comparable>

void BinarySearchTree<Comparable>::

remove( const Comparable & x, Node \* & t ) const {

if( t == NULL )

throw ItemNotFoundException( );

if( x < t->element )

remove( x, t->left );

else if( t->element < x )

remove( x, t->right );

else if( t->left != NULL && t->right != NULL ){ // Two children

t->element = findMin( t->right )->element;

removeMin( t->right ); // Remove minimum

}

else{

BinaryNode<Comparable> \*oldNode = t;

t = ( t->left != NULL ) ? t->left : t->right; // Reroot t

delete oldNode; // delete old root

return;

}!

t->size--;

1. What is the maximum height a binary search tree with **n** nodes can have? Explain your answer, very briefly.

If the binary tree is sorted the max height of a binary search tree with **n** nodes is **n-1** because the height of a node is the number of edges from the node to the deepest leaf.

1. Given the input [4371, 1323, 6173, 4199, 4344, 9679, 1989], a fixed table size of 10, and a hash function H(x) = x mod 10, show the resulting

(a) Linear probing hash table.

(b) Separate chaining has table.