CSE 3010 – Data Structures & Algorithms

Lecture #36

What will be covered today

- Operations on BST … Contd.
  - Inserting into a BST
  - Finding the node type
  - Finding if a node is a left or right child
  - Finding if a node has a left or right child
  - Finding the size of a BST
  - Deleting a node in the tree

Example dataset

Mat, Bat, Pot, Hut, Cat, Lit, Hat, Sat, Met, Lot

Finding if a node is a left or right child

```
bool isLeftChild(BSTNODE* root, BSTNODE* node) {
        if (node == NULL)
                return false;
        if (root == NULL)
                return false;
        if (root->left == node)
                return true;
        else
                if (node->key <= root->key)
                        return(isLeftChild(root->left,node));
                else
                        return(isLeftChild(root->right,node));
}
```

Finding size of tree

```
// Returns the number of nodes in the tree, tree
may be a subtree
int sizeTree(BSTNODE *root) {
    int count = 1;
    if (root->left != NULL)
        count = count + sizeTree(root->left);
    if (root->right != NULL)
        count = count + sizeTree(root->right);
    return count;
}
```

Delete node in a binary search tree

- Three cases to be considered
  1. Node to be deleted is a leaf
     - Delete the node
  2. Node to be deleted has one child
     - Delete the node after parent of the deleted node adjusts the pointer to bypass the deleted node
  3. Node to be deleted has two children
     - Replace the key of the deleted node with the smallest data of its right sub tree
     - Delete the node with the smallest data recursively