BST Operations

insert a node
delete a node
print the tree contents
search for a value

Insert

Iterative and recursive

Iterative

Assume we have a node:

```
node:
    node *parent
    node *leftchild
    node *rightchild
    int key
```

and a BST class, where the root of the tree is stored.

```
BST:
  private:
      node *root
  // the only access to the
     tree is through the
     root
```

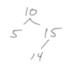
```
insert(value)
    node *parent=nullptr
    node *tmp=root
    node *n=new node(value)
    while(tmp!=nullptr){
        parent=tmp
        if(n→key<tmp→key)
            tmp=tmp→leftChild
        else
            tmp=tmp→rightChild
    }
    if parent==nullptr    //tree is empty
        root=n
    else if (n→key<parent→key)
        parent→leftChild=n
        n→parent=n

    else
        parent→rightChild=n
        n→parent=n
```

Notes:

if parent=nullptr after while loop, the root isn't defined yet and the tree is empty.

After the while exits, we know which node is the parent, but not whether the new node is the left or right child. That's why we have the conditionals after the while.

Ex:



insert(12)

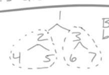
initially,
tmp=10 node.
in the while,
parent set to 10 node, then 15 node, then 14 node.
while exits when the left child of the 14 is null.
After the while, parent points to the 14, and 12 added as the left child.

Search(value)

// similar to insert, but returns a pointer to the node instead of adding a node.

```
node *tmp=root
while(tmp!=nullptr)
    if (tmp→key>value)
        tmp=tmp→leftChild
    else if(tmp→key<value)
        tmp=tmp→rightChild

    else
        return tmp
```

Trees and Subtrees



Binary tree, but not a BST.

root=1
2 is the root of a smaller subtree.
3 is the root of a smaller subtree
Every node in the tree is the root of a subtree, even the leaf nodes.

Self-similarity - Object is similar to a part of itself.

Trees have a recursive structure - defined in terms of itself. Tree defined by the subtrees within it.

Recursive algorithm - alg. that calls itself on smaller and smaller input

Print all nodes in tree

Need to visit all nodes and print key value.
Easiest to do recursively

```
Print(node *n)    // n=root on first print call.
    cout<<n→key
    if(n→leftChild!=nullptr)
        print(n→leftChild)
    if(n→rightChild!=nullptr)
        print(n→rightChild)
```

Ex: