

Joshua Anaya

CST 370

Final

02/21/18

#1

Function has no created output. InOrder() shows BST below. Balanced BST achieved by sorting array and finding median item in array as root. As it uses select sort, time consumption would be $O(n^2)$ as it runs through the loop n times to update the location of each element.

SelectSort ->A

Median = (0-A.length)/2

Pull median to front of array(a[0])

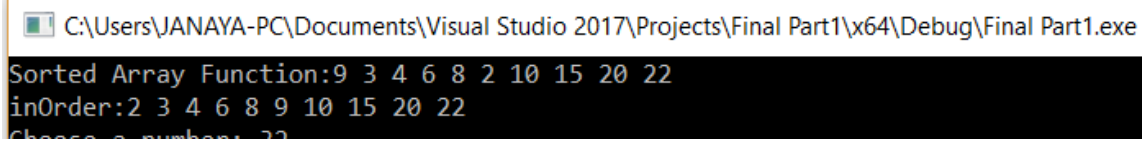
For()

Insert(A[i])

#2

inOrder displays BST from left, root, right.

inOrder display:



```
C:\Users\JANAYA-PC\Documents\Visual Studio 2017\Projects\Final Part1\x64\Debug\Final Part1.exe
Sorted Array Function:9 3 4 6 8 2 10 15 20 22
inOrder:2 3 4 6 8 9 10 15 20 22
Choose a number: 22
```

#3

LeafHeight uses recursion to find a leaf and track the links to find the height of the leaf (from the root). The time complexity is $O(n)$ as with recursion, we loop through the BST once to find which node is a leaf and which is not.

LeafHeight()

{ Return LeafHeight(myRoot)//access myRoot }

LeafHeight(BinNode *root)

{

if(root == NULL)

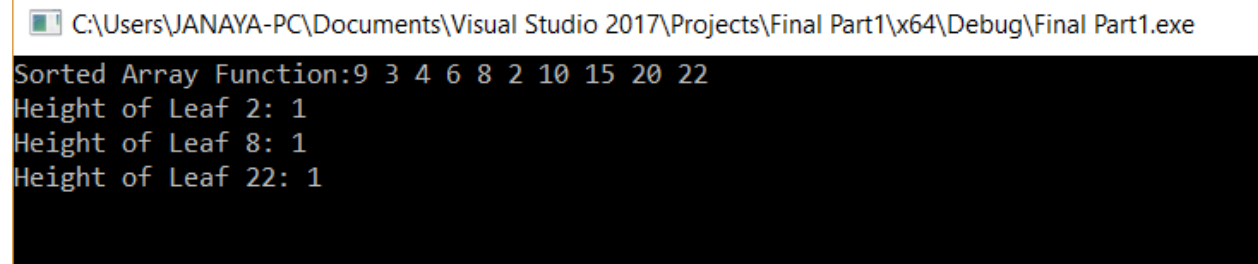
return; //no root means no leaf;

```

if(no children) //means leaf
    cout <<"height of Leaf" << root->data << : << leafHeight(root->left) + LeafHeight(root->right) +1;
if (root->left)
    LeafHeight(root->left); recursion until leaf found;
If(root->right)
    LeafHeight(root->right); recursion until leaf found;

```

LeafHeight displays found leafs but not showing height correctly:



```

C:\Users\JANAYA-PC\Documents\Visual Studio 2017\Projects\Final Part1\x64\Debug\Final Part1.exe
Sorted Array Function:9 3 4 6 8 2 10 15 20 22
Height of Leaf 2: 1
Height of Leaf 8: 1
Height of Leaf 22: 1

```

#4

NewSearch() reused parts of search provided to loop through BST. Num variable used to hold value of closest larger number to selected number. Results displayed if found, not found but larger number in array, or chosen number larger than all numbers in array. Time complexity of $O(n)$ as it cycles through each element to find match.

```

Locptr = myRoot
Int num;
Bool found;
While(not found and locptr not null)
    If(item > locptr->data) and num > locptr->data)
        num=locptr->data
    if(item< locptr->data)
        locptr = locptr->left //update to new locptr for another pass
    else if(same for right)
    else
        update num <- locptr as it was found
        update found to true because found

```

output results
 found
 not found but larger number in array
 not found no larger number in array

New search chosen number found in array:

```
C:\Users\JANAYA-PC\Documents\Visual Studio 2017\Projects\Final Part1\x64\Debug\Final Part1.exe
Sorted Array Function:9 3 4 6 8 2 10 15 20 22
inOrder:2 3 4 6 8 9 10 15 20 22
Choose a number: 3
Chosen number found in array.
```

New Search chosen number not found:

```
C:\Users\JANAYA-PC\Documents\Visual Studio 2017\Projects\Final Part1\x64\Debug\Final Part1.exe
Sorted Array Function:9 3 4 6 8 2 10 15 20 22
inOrder:2 3 4 6 8 9 10 15 20 22
Choose a number: 7
Chosen number not found but 8 was the closest in the array.
```

New Search chosen number too large:

```
C:\Users\JANAYA-PC\Documents\Visual Studio 2017\Projects\Final Part1\x64\Debug\Final Part1.exe
Sorted Array Function:9 3 4 6 8 2 10 15 20 22
inOrder:2 3 4 6 8 9 10 15 20 22
Choose a number: 32
Chosen number not found and larger than any number in array.
```