Week 7 Binary Search Tree

Generated by Doxygen 1.8.4

Wed Mar 12 2014 01:03:24

Contents

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 2

Class Documentation

2.1 AccountRecord Struct Reference

Public Attributes

- int **acctID**
- char **firstName** [nameLength]
- char **lastName** [nameLength]
- double **balance**

The documentation for this struct was generated from the following files:

- database.cpp
- database.cs

2.2 BSTree< DataType, KeyType > Class Template Reference

Classes

- class BSTreeNode

Public Member Functions

- BSTree ()
- BSTree (const BSTree< DataType, KeyType > &other)
- BSTree & operator= (const BSTree< DataType, KeyType > &other)
- ~BSTree ()
- void insert (const DataType &newDataItem)
- bool retrieve (const KeyType &searchKey, DataType &searchDataItem) const
- bool remove (const KeyType &deleteKey)
- void writeKeys () const
- void clear ()
- bool isEmpty () const
- void showStructure () const
- int getHeight () const
- int getCount () const
- void **writeLessThan** (const KeyType &searchKey) const

Protected Member Functions

- BSTreeNode * copyHelper (BSTreeNode *current, BSTreeNode *temp)
- void showHelper (BSTreeNode *p, int level) const
- void dHelper (BSTreeNode *&current)
- void iHelper (BSTreeNode *&current, const DataType data)
- bool **retHelper** (BSTreeNode *const &current, const KeyType &searchKey, DataType &searchData) const
- bool **rHelper** (BSTreeNode *&current, const KeyType &deleteKey)
- void **wHelper** (BSTreeNode *current) const
- void **wLTHelper** (BSTreeNode *current, const KeyType &searchKey) const
- int countHelper (BSTreeNode *current) const
- int hHelper (BSTreeNode *current) const

Protected Attributes

- BSTreeNode * **root**

2.2.1 Constructor & Destructor Documentation

2.2.1.1 template<typename DataType , class KeyType > BSTree< DataType, KeyType >::BSTree ( )

The binary search tree constructor sets its root pointer to NULL.

Parameters

none	

Precondition

none

Postcondition

the root pointer is set to NULL

Returns

none

2.2.1.2 template<typename DataType , class KeyType > BSTree< DataType, KeyType >::BSTree (const BSTree< DataType, KeyType > & *src*)

The binary search tree copy constructor checks if the source tree's root is equal to NULL. If not it assigns the return value of the copyhelper function to the value of root.

Parameters

a	source tree for copying

Precondition

if the source tree isn't empty

Postcondition

a copy of the source tree (made using the copy helper) is set assigned to root

Returns

none

2.2.1.3 template<typename DataType , class KeyType > BSTree< DataType, KeyType >::~BSTree ( )

The binary search tree destructor calls the destructor helper function and then sets the root pointer to NULL.

Parameters

none	

Precondition

none

Postcondition

the tree is destructed and the root pointer is set equal to NULL

Returns

none

2.2.2 Member Function Documentation

2.2.2.1 template<typename DataType , class KeyType > void BSTree< DataType, KeyType >::clear ( )

The clear function calls the destructor helper and passes in the root pointer.

Parameters

none	

Precondition

none

Postcondition

the root is passed into the destructor helper

Returns

none

2.2.2.2 template<typename DataType, class KeyType> BSTreeNode* BSTree< DataType, KeyType >::copyHelper (BSTreeNode * *current,* BSTreeNode * *temp*) [inline],[protected]

A helper function used to make a copied version of the Expression Tree whose pointer is in the parameters.

Parameters

the	current pointer in the source tree

Precondition

if the current source pointer is not null

Postcondition

a temp copy tree is created using the source data

Returns

the address of the copied tree

2.2.2.3 template<typename DataType , typename KeyType > int BSTree< DataType, KeyType >::countHelper (BSTreeNode * *current*) const [protected]

Recursive helper for getCount. Returns the total number of nodes in the tree.

Parameters

pointer	to the current node

Precondition

current is equal to NULL

Postcondition

total number of nodes is counted

Returns

returns the number of items in the tree

2.2.2.4 template<typename DataType , class KeyType > void BSTree< DataType, KeyType >::dHelper (BSTreeNode *& *current*) [protected]

The binary seach tree destructor helper loops while the current node being evaluated is an operator, if so it recursively calls itself in both the left and right direction.

If the current node left or right pointers equal NULL, it deletes the node and then sets the pointer to NULL.

Parameters

a	pointer to the current node in a tree to be destructed

Precondition

none

Postcondition

the tree is destructed recursively

Returns

none

2.2.2.5 template<typename DataType , typename KeyType > int BSTree< DataType, KeyType >::getCount ( ) const

The getCount function calls the counter helper and passes in the root pointer.

Parameters

none	

Precondition

none

Postcondition

the root is passed into the counter helper

Returns

none

2.2.2.6 template<typename DataType , typename KeyType > int BSTree< DataType, KeyType >::getHeight ( ) const

The getHeight function calls the height helper and passes in the root pointer.

Parameters

none	

Precondition

none

Postcondition

the root is passed into the height helper

Returns

none

2.2.2.7 template<typename DataType , typename KeyType > int BSTree< DataType, KeyType >::hHelper (BSTreeNode * *current*) const [protected]

Recursive helper for getHeight. Returns the longest height found in the tree.

Parameters

pointer	to the current node

Precondition

current is equal to NULL

Postcondition

total number of nodes is counted per side from the root

Returns

returns the height of the tree

2.2.2.8 template<typename DataType , class KeyType > void BSTree< DataType, KeyType >::iHelper (BSTreeNode *& *current,* const DataType *data*) [protected]

The binary search tree recursively loops through the tree until it has correctly gone to the correct null pointer in the tree by doing comparisons between the passed in data and the data at the pointer. Once the pointer is null, it is assigned to a node with the parameterized data.

Parameters

the	current node pointer in the expression tree being inserted

Precondition

none

Postcondition

the pointer passed into the final instance of the function is assigned the leaf of the new data

Returns

none

2.2.2.9 template<typename DataType , class KeyType > void BSTree< DataType, KeyType >::insert (const DataType & *newdataItem*)

The inser function calls the insert helper and passes in the root pointer.

Parameters

data	to be inserted

Precondition

none

Postcondition

the root is passed in by reference and assigned when insert helper is finished

Returns

none

2.2.2.10 template<typename DataType , class KeyType > bool BSTree< DataType, KeyType >::isEmpty ( ) const

The isEmpty function returns true if empty false if not.

Parameters

none	

Precondition

none

Postcondition

none

Returns

boolean value of if true of not

2.2.2.11 template<typename DataType , class KeyType > BSTree< DataType, KeyType > & BSTree< DataType, KeyType >::operator= (const BSTree< DataType, KeyType > & *src*)

The overloaded assignment operator checks the addresses of source and this, and if not equal it clears any data left in this and calls assigns the return value from copy helper to root.

Parameters

a	source tree for copying

Precondition

if the addresses of the source and this don't match

Postcondition

a copy of the source tree (made using the copy helper) is set assigned to root

Returns

this binary search tree is returned (for chaining)

2.2.2.12 template<typename DataType , class KeyType > bool BSTree< DataType, KeyType >::remove (const KeyType & *deleteKey*)

The remove function calls the remove helper and passes in the root pointer and deleteKey.

Parameters

data	to be removed

Precondition

none

Postcondition

data will be removed

Returns

boolean statement on whether removed

2.2.2.13 template<typename DataType , class KeyType > bool BSTree< DataType, KeyType >::retrieve (const KeyType & *searchKey,* DataType & *searchdataItem*) const

The retreive function calls the reteive helper and passes in the root pointer.

Parameters

data	to be retrieved, and a variable to assign the data to

Precondition

if root not equal to null

Postcondition

the root and the two data variables is passed into the helper by reference and assigned when retireve helper is finished

Returns

boolean statement on whether retrived

2.2.2.14 template<typename DataType , typename KeyType > void BSTree< DataType, KeyType >::showHelper (BSTreeNode * *p,* int *level*) const [protected]

Recursive helper for showStructure. Outputs the subtree whose root node is pointed to by p. Parameter level is the level of this node within the tree.

Parameters

pointer	to the current node and a level int

Precondition

p not equal to 0

Postcondition

the tree is output

Returns

void

2.2.2.15 template<typename DataType , typename KeyType > void BSTree< DataType, KeyType >::showStructure ( ) const

Outputs the keys in a binary search tree. The tree is output rotated counterclockwise 90 degrees from its conventional orientation using a "reverse" inorder traversal. This operation is intended for testing and debugging purposes only.

Parameters

none	

Precondition

if not empty

Postcondition

the root is passed into the show helper & outputs an endl

Returns

none

2.2.2.16 template<typename DataType , class KeyType > void BSTree< DataType, KeyType >::writeKeys ( ) const

The writeKey function calls the writeKey helper and passes in the root pointer.

Parameters

none	

Precondition

none

Postcondition

helper function called with a endl after

Returns

none

The documentation for this class was generated from the following files:

- BSTree.h
- bs.cpp
- BSTree.cpp
- show9.cpp

2.3 BSTree< DataType, KeyType >::BSTreeNode Class Reference

Public Member Functions

- BSTreeNode (const DataType &nodeDataItem, BSTreeNode *leftPtr, BSTreeNode *rightPtr)

Public Attributes

- DataType **dataItem**
- BSTreeNode * **left**
- BSTreeNode * **right**

2.3.1 Constructor & Destructor Documentation

2.3.1.1 template<typename DataType , class KeyType > BSTree< DataType, KeyType >::BSTreeNode::BSTreeNode (const DataType & *nodeDataItem,* BSTreeNode * *leftPtr,* BSTreeNode * *rightPtr*)

The Node constructor takes in parameterized data and sets it to the appropriate data members.

Parameters

a	template value of data, a node left pointer, and a node right pointer

Precondition

none

Postcondition

a node is created with the parametrized data

Returns

none

The documentation for this class was generated from the following files:

- BSTree.h
- BSTree.cpp

2.4 IndexEntry Struct Reference

Public Member Functions

- int **getKey** () const
- int **key** () const

Public Attributes

- int **acctID**
- long **recNum**

The documentation for this struct was generated from the following files:

- database.cpp
- database.cs

2.5 TestData Class Reference

Public Member Functions

- void **setKey** (int newKey)
- int **getKey** () const

The documentation for this class was generated from the following file:

- test9.cpp

Index