Lab09 Binary Search Tree

Generated by Doxygen 1.8.11

Contents

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Class Documentation

3.1 AccountRecord Struct Reference

Public Attributes

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

3.1.1 Member Data Documentation

3.1.1.1 int AccountRecord::acctID

3.1.1.2 double AccountRecord::balance

3.1.1.3 char AccountRecord::firstName[nameLength]

3.1.1.4 char AccountRecord::lastName[nameLength]

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

- database.cpp

3.2 BSTree< DataType, KeyType > Class Template Reference

```
#include <BSTree.h>
```

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

- void showHelper (BSTreeNode ∗p, int level) const
- void insertHelper (BSTreeNode ∗&node, const DataType &data)
- void clearHelper (BSTreeNode ∗&node)
- void copyHelper (BSTreeNode ∗&node, BSTreeNode ∗other)
- bool retrieveHelper (BSTreeNode ∗node, const KeyType &key, DataType &data) const
- void writeKeyHelper (BSTreeNode ∗node) const
- int getHeightHelper (BSTreeNode ∗node) const
- int max (int i, int j) const
- int getCountHelper (BSTreeNode ∗node) const
- void writeLessThanHelper (BSTreeNode ∗node, const KeyType &key) const
- bool removeHelper (BSTreeNode ∗&root, const KeyType &key)

Protected Attributes

- BSTreeNode ∗ root

3.2.1 Constructor & Destructor Documentation

3.2.1.1 template<class T , class Key > BSTree< T, Key >::BSTree ( )

BStree Binary search tree empty constructor

Precondition

empty binary search tree

Postcondition

binary search tree with root set to NULL

3.2.1.2 template<typename DataType, class KeyType> BSTree< DataType, KeyType >::BSTree (const BSTree< DataType, KeyType > & *other*)

3.2.1.3 template<class T , class Key > BSTree< T, Key >::∼BSTree ( )

∼BStree destructor

Precondition

a (potentially) non - empty binary search tree

Postcondition

a binary search tree with a NULL root

3.2.2 Member Function Documentation

3.2.2.1 template<class T , class Key > void BSTree< T, Key >::clear ( )

clear clears the contents of the BST

Returns

void

Precondition

binary search with data in it

Postcondition

empty binary search tree

3.2.2.2 template<class T , class Key > void BSTree< T, Key >::clearHelper (BSTreeNode ∗& *node*) [protected]

clearHelper recursive helper funtion that uses a post order traversal to delete nodes.

Returns

void

Parameters

node	node passed by pointer reference to the current node in the tree

Precondition

a binary seawrch tree with n + 1 nodes

Postcondition

a binary search tree with n nodes (current will have been deleted)

3.2.2.3 template<class T , class Key > void BSTree< T, Key >::copyHelper (BSTreeNode ∗& *node,* BSTreeNode ∗ *other*) `[protected]`

copyHelper recursive helper function for operator equals and copy constructor. Uses preorder traversal.

Returns

void

Parameters

other	binary search tree node to copy from
node	binary search tree node passed by reference

Precondition

a binary search tree with n - 1 nodes

Postcondition

a binary search tree with n nodes. The new node is a copy of other.

3.2.2.4 template<class T , class Key > int BSTree< T, Key >::getCount ( ) const

getCount returns the number of nodes in the tree.

Returns

int number of nodes in the tree

Precondition

binary search tree with n nodes

Postcondition

the number of nodes in the tree

3.2.2.5 template<class T , class Key > int BSTree< T, Key >::getCountHelper (BSTreeNode ∗ *node*) const `[protected]`

getCountHelper recursive helper function that returns the count of nodes in the tree.

Returns

int number of nodes in the tree

Parameters

node	node pointer to the current node

Precondition

a node in the binary search tree

Postcondition

an incremented count if the node found is not null

3.2.2.6 template<class T , class Key > int BSTree< T, Key >::getHeight ( ) const

getHeight returns the height of the tree

Returns

int the (max) height of the tree

Precondition

a binary search tree

Postcondition

the height of binary search tree is returned

3.2.2.7 template<class T , class Key > int BSTree< T, Key >::getHeightHelper (BSTreeNode ∗ *node*) const [protected]

getHeightHelper recursive helper function that returns the hieght of the tree

Returns

int height of the tree

Parameters

node	current node in the tree

Precondition

binary search tree node

Postcondition

count incremented by one if the node was not NULL

3.2.2.8 template<typename DataType, class KeyType> void BSTree< T, Key >::insert (const DataType & *newDataItem*)

insert inserts a new node into the BST

Returns

void

Parameters

newData	data for the new node

Precondition

a binary search tree with n -1 nodes.

Postcondition

a binary search tree with n nodes (inserts new node)

3.2.2.9 template<typename DataType, class KeyType> void BSTree< T, Key >::insertHelper (BSTreeNode *& *node*, const DataType & *data*) [protected]

insertHelper recursive helper function for insert.

Returns

void

Parameters

node	node passed by pointer reference (will be passed or attached to)
data	data for the new node

Precondition

binary search tree with n - 1 nodes

Postcondition

binary search tree with a new node (satisfies the binary search tree property)

3.2.2.10 template<class T , class Key > bool BSTree< T, Key >::isEmpty ( ) const

isEmpty checks whether or not the tree is empty

Returns

bool returns true if the tree is empty

Precondition

a binary search tree

Postcondition

returns true if empty

3.2.2.11 template<class T , class Key > int BSTree< T, Key >::max (int *i*, int *j*) const `[protected]`

max returns the max between the two parameteres

Returns

int max between i and j

Parameters

i	int to be compared
j	int to be compared

Precondition

two numbers with unkown max

Postcondition

max of i and j

3.2.2.12 template<typename DataType, class KeyType> BSTree< T, Key > & BSTree< T, Key >::operator= (const BSTree< DataType, KeyType > & *other*)

operator= overloaded assignment operator

Returns

returns the address of (this) binary search tree

Parameters

other	const binary search tree to copy from

Precondition

a binary search tree (initialized or unitialized)

Postcondition

a binary search tree that is a deep copy of other

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

remove

Returns

bool returns true if a node with the specified key was removed

Parameters

key	key of the item to be removed

Precondition

a tree with n + 1 items

Postcondition

a tree with the key specified removed

3.2.2.14 template<typename DataType, class KeyType> bool BSTree< T, Key >::removeHelper (BSTreeNode ∗& *root,* const KeyType & *key*) [protected]

removeHelper uses pre order traversal and logic to remove a node from the tree

Returns

bool returns true if the key was found and the node was removed

Parameters

node	node pointer to the current node in the tree
key	key of the data item to be removed

Precondition

tree with n + 1 nodes

Postcondition

tree with n nodes. Node with the key has been removed

3.2.2.15 template<typename DataType, class KeyType> bool BSTree< T, Key >::retrieve (const KeyType & *searchKey,* DataType & *searchDataItem*) const

retrieve retrieves the data item with a specified key

Returns

bool true if the key exists, false otherwise

Parameters

key	key of the data item in the tree
data	passed by reference (will hold the data if found)

Precondition

unfilled generic data type

Postcondition

generic data type that will contain the retrieved data if found

3.2.2.16 template<typename DataType, class KeyType> bool BSTree< T, Key >::retrieveHelper (BSTreeNode ∗ *node,* const KeyType & *key,* DataType & *data*) const [protected]

retrieveHelper recursive helper for retreive. Uses pre order traversal

Returns

bool returns true if the data item was found

Parameters

node	node pointer to the current node in the tree
data	conatainer for potenital value
key	key for comparison of values within the tree

Precondition

unfilled dataItem

Postcondition

generic data type containing the value associated with the key

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

showHelper recursive helper that prints the tree to console

Returns

void

Parameters

p	binary search tree node (current) node in the tree
level	level of the tree used to tab children node

Precondition

binary search tree node

Postcondition

one node of the tree printed to console

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

showStructure factory print method (provided)

Returns

void

Precondition

binary search tree

Postcondition

binary search tree has been printed to console

3.2.2.19 template< class T , class Key > void BSTree< T, Key >::writeKeyHelper (BSTreeNode ∗ *node*) const `[protected]`

writeKeyHelper recursive helper function. uses in order traversal to print the keys.

Returns

void

Parameters

node	current node in the tree whos data will be printed.

Precondition

binary search tree

Postcondition

one element of the binary search tree node will be printed to console

3.2.2.20 template<class T , class Key > void BSTree< T, Key >::writeKeys ( ) const

writeKeys writes the keys to console in ascending order

Returns

void

Precondition

binary search tree

Postcondition

contents of binary search tree are written to console

3.2.2.21 template<typename DataType, class KeyType> void BSTree< T, Key >::writeLessThan (const KeyType & *searchKey*) const

writeLessThan writes the keys less than (key) to console

Returns

void

Parameters

key	key to be compared to

Precondition

a binary search tree

Postcondition

all data less than key k printed to console

3.2.2.22 template<typename DataType, class KeyType> void BSTree< T, Key >::writeLessThanHelper (BSTreeNode ∗ *node,* const KeyType & *key*) const [protected]

writeLessThanHeper recursive helper function that prints nodes with key < k to console

Returns

void

Parameters

node	pointer to current node in the tree
key	key used for comparison

Precondition

binary search tree

Postcondition

data less than key printed to console

3.2.3 Member Data Documentation

3.2.3.1 template<typename DataType, class KeyType> BSTreeNode∗ BSTree< DataType, KeyType >::root [protected]

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

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

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

```
#include <BSTree.h>
```

Public Member Functions

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

Public Attributes

- DataType dataItem
- BSTreeNode ∗ left
- BSTreeNode ∗ right

3.3.1 Constructor & Destructor Documentation

3.3.1.1 template<typename DataType, class KeyType> BSTree< T, Key >::BSTreeNode::BSTreeNode (const DataType & *nodeDataItem,* BSTreeNode ∗ *leftPtr,* BSTreeNode ∗ *rightPtr*)

BStreeNode Binary search tree node constructor

Parameters

nodeDataItem	generic data item to be stored in the node
leftPtr	pointer to the left child of the node
rightPtr	pointer to the right child of the node

Precondition

uninitialized Binary search tree node

Postcondition

new binary search tree node

3.3.2 Member Data Documentation

3.3.2.1 template<typename DataType, class KeyType> DataType BSTree< DataType, KeyType >::BSTreeNode::dataItem

3.3.2.2 template<typename DataType, class KeyType> BSTreeNode∗ BSTree< DataType, KeyType >::BSTreeNode::left

3.3.2.3 template<typename DataType, class KeyType> BSTreeNode ∗ BSTree< DataType, KeyType >::BSTreeNode::right

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

- BSTree.h
- BSTree.cpp

3.4 IndexEntry Struct Reference

Public Member Functions

- int getKey () const

Public Attributes

- int acctID
- long recNum

3.4.1 Member Function Documentation

3.4.1.1 int IndexEntry::getKey ( ) const `[inline]`

3.4.2 Member Data Documentation

3.4.2.1 int IndexEntry::acctID

3.4.2.2 long IndexEntry::recNum

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

- database.cpp

3.5 TestData Class Reference

Public Member Functions

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

3.5.1 Member Function Documentation

3.5.1.1 int TestData::getKey ( ) const `[inline]`

3.5.1.2 void TestData::setKey (int *newKey*) `[inline]`

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

- test9.cpp

Chapter 4

File Documentation

4.1 BSTree.cpp File Reference

```
#include "BSTree.h"
```

4.2 BSTree.h File Reference

```
#include <stdexcept>
#include <iostream>
```

Classes

- class BSTree< DataType, KeyType >
- class BSTree< DataType, KeyType >::BSTreeNode

4.3 config.h File Reference

Macros

- #define LAB9_TEST1 1
- #define LAB9_TEST2 1
- #define LAB9_TEST3 1

4.3.1 Macro Definition Documentation

4.3.1.1 #define LAB9_TEST1 1

BSTree class (Lab 9) configuration file. Activate test 'N' by defining the corresponding LAB9_TESTN to have the value 1. Deactive test 'N' by setting the value to 0.

4.3.1.2 #define LAB9_TEST2 1

4.3.1.3 #define LAB9_TEST3 1

4.4 database.cpp File Reference

```
#include <iostream>
#include <fstream>
#include "BSTree.cpp"
```

Classes

- struct AccountRecord
- struct IndexEntry

Functions

- int main ()

Variables

- const int nameLength = 11
- const long bytesPerRecord = 38

4.4.1 Function Documentation

4.4.1.1 int main ( )

4.4.2 Variable Documentation

4.4.2.1 const long bytesPerRecord = 38

4.4.2.2 const int nameLength = 11

4.5 show9.cpp File Reference

4.6 test9.cpp File Reference

```
#include <iostream>
#include "BSTree.cpp"
#include "config.h"
```

Classes

- class TestData

Functions

- void print_help ()
- int main ()

4.6.1 Function Documentation

4.6.1.1 int main ( )

4.6.1.2 void print_help ( )

Index