

Binary Search Tree - Ernest Landrito

Generated by Doxygen 1.8.5

Thu Oct 31 2013 11:27:54

Contents

1	Class Index	1
1.1	Class List	1
2	File Index	3
2.1	File List	3
3	Class Documentation	5
3.1	AccountRecord Struct Reference	5
3.1.1	Member Data Documentation	5
3.1.1.1	acctID	5
3.1.1.2	balance	5
3.1.1.3	firstName	5
3.1.1.4	lastName	5
3.2	BSTree< DataType, KeyType > Class Template Reference	5
3.2.1	Constructor & Destructor Documentation	6
3.2.1.1	BSTree	6
3.2.1.2	BSTree	7
3.2.1.3	~BSTree	7
3.2.2	Member Function Documentation	7
3.2.2.1	clear	7
3.2.2.2	getCount	7
3.2.2.3	getHeight	8
3.2.2.4	insert	8
3.2.2.5	isEmpty	8
3.2.2.6	operator=	9
3.2.2.7	recClear	9
3.2.2.8	recCopy	9
3.2.2.9	recGetCount	10
3.2.2.10	recGetHeight	10
3.2.2.11	recInsert	11
3.2.2.12	recRemove	11
3.2.2.13	recRetrieve	12

3.2.2.14	recWriteKeys	12
3.2.2.15	recWriteLessThan	12
3.2.2.16	remove	13
3.2.2.17	retrieve	13
3.2.2.18	showHelper	14
3.2.2.19	showStructure	14
3.2.2.20	writeKeys	14
3.2.2.21	writeLessThan	14
3.2.3	Member Data Documentation	14
3.2.3.1	root	14
3.3	BSTree< DataType, KeyType >::BSTreeNode Class Reference	14
3.3.1	Constructor & Destructor Documentation	15
3.3.1.1	BSTreeNode	15
3.3.2	Member Data Documentation	15
3.3.2.1	dataItem	15
3.3.2.2	left	15
3.3.2.3	right	15
3.4	IndexEntry Struct Reference	15
3.4.1	Member Function Documentation	16
3.4.1.1	getKey	16
3.4.1.2	key	16
3.4.2	Member Data Documentation	16
3.4.2.1	acctID	16
3.4.2.2	recNum	16
3.5	TestData Class Reference	16
3.5.1	Member Function Documentation	16
3.5.1.1	getKey	16
3.5.1.2	setKey	16
4	File Documentation	17
4.1	BSTree.cpp File Reference	17
4.2	BSTree.h File Reference	17
4.3	config.h File Reference	17
4.3.1	Macro Definition Documentation	17
4.3.1.1	LAB9_TEST1	17
4.3.1.2	LAB9_TEST2	18
4.3.1.3	LAB9_TEST3	18
4.4	database.cpp File Reference	18
4.4.1	Function Documentation	18
4.4.1.1	main	18

4.4.2	Variable Documentation	18
4.4.2.1	bytesPerRecord	18
4.4.2.2	nameLength	18
4.5	database.cs File Reference	18
4.5.1	Function Documentation	19
4.5.1.1	main	19
4.5.2	Variable Documentation	19
4.5.2.1	bytesPerRecord	19
4.5.2.2	nameLength	19
4.6	show9.cpp File Reference	19
4.7	test9.cpp File Reference	19
4.7.1	Function Documentation	19
4.7.1.1	main	19
4.7.1.2	print_help	19

Chapter 1

Class Index

1.1 Class List

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

AccountRecord	5
BSTree< DataType, KeyType >	5
BSTree< DataType, KeyType >::BSTreeNode	14
IndexEntry	15
TestData	16

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

BSTree.cpp	17
BSTree.h	17
config.h	17
database.cpp	18
database.cs	18
show9.cpp	19
test9.cpp	19

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

3.1.1.4 char AccountRecord::lastName

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

- [database.cpp](#)
- [database.cs](#)

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 **recInsert** (const DataType &newDataItem, **BSTreeNode** *&node)
- bool **recRetrieve** (const KeyType &searchKey, DataType &searchDataItem, **BSTreeNode** *node) const
- bool **recRemove** (const KeyType &deleteKey, **BSTreeNode** *&node)
- void **recWriteKeys** (**BSTreeNode** *node) const
- void **recClear** (**BSTreeNode** *&node)
- void **recGetHeight** (**BSTreeNode** *node, int currentLevel, int &maxLevel) const
- int **recGetCount** (**BSTreeNode** *node) const
- void **recCopy** (**BSTreeNode** *&node, **BSTreeNode** *otherNode)
- void **recWriteLessThan** (const KeyType &searchKey, **BSTreeNode** *node) const
- void **showHelper** (**BSTreeNode** *p, int level) const

Protected Attributes

- **BSTreeNode** * **root**

3.2.1 Constructor & Destructor Documentation

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

Precondition

New **BSTree** Class

Postcondition

root pointing to null

Algorithm:

- point root to null

Exceptional/Error Conditions:

- none

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

Precondition

new [BSTree](#) class

Postcondition

a deep copy of source [BSTree](#)

Parameters

<i>source</i>	Source BSTree to be deep copied
---------------	---

Algorithm:

- Use overloaded assignment operator to copy the data from the source to this class

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

Precondition

a [BSTreeClass](#)

Postcondition

deallocated [BSTreeClass](#)

Algorithm:

- Use clear member function to deallocate the tree

3.2.2 Member Function Documentation

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

Precondition

a [BSTreeClass](#)

Postcondition

a deallocated tree

Algorithm:

- Use recClear member function to make the expression tree

3.2.2.2 `template<typename DataType , class KeyType > int BSTree< DataType, KeyType >::getCount () const`

Precondition

a [BSTreeClass](#)

Postcondition

returns the number of nodes in the tree

Returns

returns the height of the tree

Algorithm:

- use recGetCount member function

3.2.2.3 `template<typename DataType , class KeyType > int BSTree< DataType, KeyType >::getHeight () const`

Precondition

a BSTreeClass

Postcondition

returns the height of the tree

Returns

returns the height of the tree

Algorithm:

- use recGetHeight member function

3.2.2.4 `template<typename DataType , class KeyType > void BSTree< DataType, KeyType >::insert (const DataType & newDataType)`

Precondition

a BSTreeClass

Postcondition

a new node inserted into the [BSTree](#)

Parameters

<i>newDataType</i>	Data to be inserted into the BSTree
--------------------	---

Algorithm:

- Use reInsert member function

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

Precondition

a BSTreeClass

Postcondition

returns if the tree is empty

Returns

Returns true if root is pointing to NULL

Algorithm:

- return true if root is pointing to NULL

3.2.2.6 `template<typename DataType , class KeyType > BSTree< DataType, KeyType > & BSTree< DataType, KeyType >::operator= (const BSTree< DataType, KeyType > & other)`

Precondition

a BSTreeClass

Postcondition

a deep copy of source [BSTree](#)

Parameters

<i>source</i>	Source BSTree to be deep copied
---------------	---

Algorithm:

- Use recCopy to make a copy of each node in the [BSTree](#)

3.2.2.7 `template<typename DataType , class KeyType > void BSTree< DataType, KeyType >::recClear (BSTreeNode *& node) [protected]`

Precondition

an BSTreeClass

Postcondition

deallocated BSTreeClass

Parameters

<i>node</i>	the node to be deleted.
-------------	-------------------------

Algorithm:

- Post order traverse the tree, deleting the node when bottom of tree is reached

3.2.2.8 `template<typename DataType , class KeyType > void BSTree< DataType, KeyType >::recCopy (BSTreeNode *& node, BSTreeNode * otherNode) [protected]`

Precondition

an BSTreeClass

Postcondition

a deep copy of source [BSTree](#)

Parameters

<i>node</i>	location in this tree
<i>otherNode</i>	location in source tree

Algorithm:

- if the source node is not at a NULL location -create temp nodes ptrs and copy those from source -create a node and point to the temp nodes

3.2.2.9 `template<typename DataType , class KeyType > int BSTree< DataType, KeyType >::recGetCount (BSTreeNode * node) const [protected]`

Precondition

a [BSTreeNode](#)

Postcondition

returned amount of nodes in the tree

Returns

Amount of nodes in the tree

Parameters

<i>node</i>	the location in the tree
-------------	--------------------------

Algorithm:

- if at a null position return 0
- else return 1 plus recursive call on the children

3.2.2.10 `template<typename DataType , class KeyType > void BSTree< DataType, KeyType >::recGetHeight (BSTreeNode * node, int currentLevel, int & maxLevel) const [protected]`

Precondition

a [BSTreeNode](#)

Postcondition

returned value of the height of the tree

Parameters

<i>node</i>	the location in the tree
<i>currentLevel</i>	the current level of the tree
<i>maxLevel</i>	reference to the value to be returned

Algorithm:

- if at a leaf check if level is greater than max, if so return the value
- if not at a leaf go to children

3.2.2.11 `template<typename DataType , class KeyType > void BSTree< DataType, KeyType >::reclInsert (const DataType & newDatalItem, BSTreeNode *& node) [protected]`

Precondition

an new [BSTreeNode](#)

Postcondition

inserted Value in the tree

Parameters

<i>newDatalItem</i>	new node data to be put into the tree
<i>*node</i>	Pointer to current node

Algorithm:

- check if youre at a leaf, if so insert the data
- check if you found a value of the same key, if so update
- check if your key is greater than current, if so traverse right
- else traverse left

3.2.2.12 `template<typename DataType , class KeyType > bool BSTree< DataType, KeyType >::recRemove (const KeyType & deleteKey, BSTreeNode *& node) [protected]`

Precondition

a [BSTreeNode](#)

Postcondition

node removed from the list

Returns

Returns if successful

Parameters

<i>deleteKey</i>	The key to be searched for in the tree.
<i>node</i>	the location in the tree to be checked

Algorithm:

- check if youre at the end of the list
- check if you're at the right location if so move on
- check if you are at a leaf if so delete the node
- else check if there is one child that is null
- if there is one child that is null delete the node
- else find the node just less than the node, replace it and delete the node
- if reaches end of the list return false

3.2.2.13 `template<typename DataType , class KeyType > bool BSTree< DataType, KeyType >::recRetrieve (const KeyType & searchKey, DataType & searchDataItem, BSTreeNode * node) const` [protected]

Precondition

a [BSTreeNode](#)

Postcondition

a copy of the search data item

Returns

Returns if successful

Parameters

<i>searchKey</i>	The key to be searched for in the tree.
<i>searchDataItem</i>	The reference variable to store the found data
<i>node</i>	the location in the tree to be checked

Algorithm:

- check if youre at the end of the list
- check if you are at the right key if so assign data
- else check if key is less than location, if so call function to left
- else call function to the right
- if reaches end of the list return false

3.2.2.14 `template<typename DataType , class KeyType > void BSTree< DataType, KeyType >::recWriteKeys (BSTreeNode * node) const` [protected]

Precondition

a [BSTreeNode](#)

Postcondition

printed list of the keys

Parameters

<i>node</i>	the location in the tree to be checked
-------------	--

Algorithm:

- in order traversal of the tree printing the nodes

3.2.2.15 `template<typename DataType , class KeyType > void BSTree< DataType, KeyType >::recWriteLessThan (const KeyType & searchKey, BSTreeNode * node) const` [protected]

Precondition

a [BSTreeNode](#)

Postcondition

printed list of the keys

Parameters

<i>node</i>	the location in the tree to be checked
-------------	--

Algorithm:

- in order traversal of the tree printing the nodes only printing the trees if the key is less than the search key

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

Precondition

a BSTreeClass

Postcondition

a data item removed from the list if found

Returns

Returns true if deleted, false if not

Parameters

<i>deleteKey</i>	The key value to be searched for in the tree
------------------	--

Algorithm:

- Use recRemove member function

3.2.2.17 `template<typename DataType , class KeyType > bool BSTree< DataType, KeyType >::retrieve (const KeyType & searchKey, DataType & searchDataItem) const`

Precondition

a BSTreeClass

Postcondition

a copy of the dataItem found

Returns

Returns true if found, false if not

Parameters

<i>searchKey</i>	The key value to be searched for in the tree
<i>searchDataItem</i>	The reference value to be changed if found

Algorithm:

- Use recRetrieve member function

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

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

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

Precondition

a BSTreeClass

Postcondition

The key values of the list printed in ascending order

Algorithm:

- Use `recWriteKeys` member function

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

Precondition

a BSTreeClass

Postcondition

prints the key values less than the one given

Parameters

<i>searchKey</i>	the key value to upper bound the list
------------------	---------------------------------------

Algorithm:

- use `recWriteLessThan` member function

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< DataType, KeyType >::BSTreeNode::BSTreeNode (const DataType & nodeDataItem, BSTreeNode * leftPtr, BSTreeNode * rightPtr)`

Precondition

an new [BSTreeNode](#)

Postcondition

initialized [BSTreeNode](#)

Parameters

<i>nodeDataItem</i>	Element to be stored in the node
<i>*leftPtr</i>	Pointer to the next left node
<i>*rightPtr</i>	Pointer to the next right node

Algorithm:

- Assign parameters to the corresponding variables

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
- int [key](#) () const

Public Attributes

- int [acctID](#)
- long [recNum](#)

3.4.1 Member Function Documentation

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

3.4.1.2 `int IndexEntry.key () 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 files:

- [database.cpp](#)
- [database.cs](#)

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 <stdexcept>
#include <iostream>
#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 0](#)

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 0`

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 database.cs File Reference

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

Classes

- struct [AccountRecord](#)
- struct [IndexEntry](#)

Functions

- void [main](#) ()

Variables

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

4.5.1 Function Documentation

4.5.1.1 void main ()

4.5.2 Variable Documentation

4.5.2.1 const long bytesPerRecord = 38

4.5.2.2 const int nameLength = 11

4.6 show9.cpp File Reference

4.7 test9.cpp File Reference

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

Classes

- class `TestData`

Functions

- void `print_help` ()
- int `main` ()

4.7.1 Function Documentation

4.7.1.1 int main ()

4.7.1.2 void print_help ()

Index

- ~BSTree
 - BSTree, 7
- AccountRecord, 5
 - acctID, 5
 - balance, 5
 - firstName, 5
 - lastName, 5
- acctID
 - AccountRecord, 5
 - IndexEntry, 16
- BSTree
 - ~BSTree, 7
 - BSTree, 6
 - BSTree, 6
 - clear, 7
 - getCount, 7
 - getHeight, 8
 - insert, 8
 - isEmpty, 8
 - operator=, 9
 - recClear, 9
 - recCopy, 9
 - recGetCount, 10
 - recGetHeight, 10
 - reInsert, 10
 - recRemove, 11
 - recRetrieve, 11
 - recWriteKeys, 12
 - recWriteLessThan, 12
 - remove, 13
 - retrieve, 13
 - root, 14
 - showHelper, 13
 - showStructure, 14
 - writeKeys, 14
 - writeLessThan, 14
- BSTree< DataType, KeyType >, 5
- BSTree< DataType, KeyType >::BSTreeNode, 14
- BSTree.cpp, 17
- BSTree.h, 17
- BSTree::BSTreeNode
 - BSTreeNode, 15
 - dataltem, 15
 - left, 15
 - right, 15
- BSTreeNode
 - BSTree::BSTreeNode, 15
- balance
 - AccountRecord, 5
- bytesPerRecord
 - database.cpp, 18
 - database.cs, 19
- clear
 - BSTree, 7
- config.h, 17
 - LAB9_TEST1, 17
 - LAB9_TEST2, 17
 - LAB9_TEST3, 18
- dataltem
 - BSTree::BSTreeNode, 15
- database.cpp, 18
 - bytesPerRecord, 18
 - main, 18
 - nameLength, 18
- database.cs, 18
 - bytesPerRecord, 19
 - main, 19
 - nameLength, 19
- firstName
 - AccountRecord, 5
- getCount
 - BSTree, 7
- getHeight
 - BSTree, 8
- getKey
 - IndexEntry, 16
 - TestData, 16
- IndexEntry, 15
 - acctID, 16
 - getKey, 16
 - key, 16
 - recNum, 16
- insert
 - BSTree, 8
- isEmpty
 - BSTree, 8
- key
 - IndexEntry, 16
- LAB9_TEST1
 - config.h, 17
- LAB9_TEST2
 - config.h, 17

- LAB9_TEST3
 - config.h, 18
- lastName
 - AccountRecord, 5
- left
 - BSTree::BSTreeNode, 15
- main
 - database.cpp, 18
 - database.cs, 19
 - test9.cpp, 19
- nameLength
 - database.cpp, 18
 - database.cs, 19
- operator=
 - BSTree, 9
- print_help
 - test9.cpp, 19
- recClear
 - BSTree, 9
- recCopy
 - BSTree, 9
- recGetCount
 - BSTree, 10
- recGetHeight
 - BSTree, 10
- recInsert
 - BSTree, 10
- recNum
 - IndexEntry, 16
- recRemove
 - BSTree, 11
- recRetrieve
 - BSTree, 11
- recWriteKeys
 - BSTree, 12
- recWriteLessThan
 - BSTree, 12
- remove
 - BSTree, 13
- retrieve
 - BSTree, 13
- right
 - BSTree::BSTreeNode, 15
- root
 - BSTree, 14
- setKey
 - TestData, 16
- show9.cpp, 19
- showHelper
 - BSTree, 13
- showStructure
 - BSTree, 14
- test9.cpp, 19
- main, 19
- print_help, 19
- TestData, 16
 - getKey, 16
 - setKey, 16
- writeKeys
 - BSTree, 14
- writeLessThan
 - BSTree, 14