PA07

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 ()

  default BSTree constructor
- BSTree (const BSTree< DataType, KeyType > &other)

  copy BSTree constructor
- BSTree & operator= (const BSTree< DataType, KeyType > &other)

  overloaded assignment operator
- ~BSTree ()

  default BSTree destructor
- void insert (const DataType &newDataItem)

  insert() function that calls insert_Helper()
- bool retrieve (const KeyType &searchKey, DataType &searchDataItem) const

  retrieve() function that calls retrieve_Helper()
- bool remove (const KeyType &deleteKey)

  remove() function that calls remove_Helper()
- void writeKeys () const

  writeKeys() function that calls writeKeys_Helper()
- void clear ()

  clear() function that calls clear_Helper()
- bool isEmpty () const

  isEmpty() function that checks if root is equal to NULL
- void showStructure () const
- int getHeight () const

  getHeight() function that calls getHeight_Helper()
- int getCount () const

  getCount() function that calls getCount_Helper()

Protected Member Functions

- void showHelper (BSTreeNode *p, int level) const
- void copyConstructor_Helper (BSTreeNode *&end, BSTreeNode *start)

  Copy constructor helper that sets current BSTree equal to another.
- void insert_Helper (const DataType &newDataItem, BSTreeNode *&ptr)

  Inserts a new node containing newDataItem. If it exists, then it updates the data item with newDataItem.
- bool retrieve_Helper (const KeyType &searchKey, DataType &searchDataItem, BSTreeNode *ptr) const

  Searches BSTree for data item with the same key as searchKey; if found, replaces data item with searchDataItem and returns true.
- bool remove_Helper (const KeyType &deleteKey, BSTreeNode *&ptr)

  Delete data item with key deleteKey from the BSTree. If data is found, returns true. Else, it returns false.
- void writeKeys_Helper (BSTreeNode *ptr) const

  Outputs the keys of the data items in the BSTree. They are output in ascending order.
- void clear_Helper (BSTreeNode *&ptr)

  Removes all data from BSTree.
- int getHeight_Helper (BSTreeNode *ptr, int currentHeight) const

  Counts the number of data items in the BSTree and returns its value.
- int getCount_Helper (BSTreeNode *ptr) const

  Counts the height of the BSTree while recursively traversing itself.

Protected Attributes

- BSTreeNode * root

3.2.1 Constructor & Destructor Documentation

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

default BSTree constructor

Precondition

none

Postcondition

creates a BSTree

Parameters

none	

Returns

none

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

copy BSTree constructor

Precondition

none

Postcondition

creates a parameterized BSTree

Parameters

BSTree	variable

Returns

none

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

default BSTree destructor

Precondition

clear()

Postcondition

none

Parameters

none	

Returns

none

3.2.2 Member Function Documentation

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

clear() function that calls clear_Helper()

Precondition

clear_Helper()

Postcondition

none

Parameters

none	

Returns

none

3.2.2.2 template<typename DataType , class KeyType > void BSTree< DataType, KeyType >::clear_Helper (BSTreeNode *& *ptr*) [protected]

Removes all data from BSTree.

Precondition

none

Postcondition

Removes all data from BSTree

Parameters

BSTreeNode	

Returns

none

3.2.2.3 template<typename DataType , class KeyType > void BSTree< DataType, KeyType >::copyConstructor_Helper (BSTreeNode *& *end*, BSTreeNode * *start*) [protected]

Copy constructor helper that sets current BSTree equal to another.

Precondition

none

Postcondition

sets the current BSTree equal to another

Parameters

(2)	BSTreeNodes

Returns

none

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

getCount() function that calls getCount_Helper()

Precondition

getCount_Helper()

Postcondition

none

Parameters

none	

Returns

getCount_Helper(root)

3.2.2.5 template<typename DataType , class KeyType > int BSTree< DataType, KeyType >::getCount_Helper (BSTreeNode * *ptr*) const [protected]

Counts the height of the BSTree while recursively traversing itself.

Precondition

none

Postcondition

Returns the height of the BSTree

Parameters

BSTreeNode	

Returns

it will return 0 if empty, otherwise it will recursively traverse itself and add the total plus 1

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

getHeight() function that calls getHeight_Helper()

Precondition

getHeight_Helper()

Postcondition

none

Parameters

none	

Returns

getHeight_Helper(root, 0)

3.2.2.7 template<typename DataType , class KeyType > int BSTree< DataType, KeyType >::getHeight_Helper (BSTreeNode * *ptr,* int *currentHeight*) const [protected]

Counts the number of data items in the BSTree and returns its value.

Precondition

none

Postcondition

Returns the count of the number of data items in BSTree

Parameters

BSTreeNode	and Int

Returns

maxHeight

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

insert() function that calls insert_Helper()

Precondition

insert_Helper()

Postcondition

none

Parameters

Datatype	variable

Returns

none

3.2.2.9 template<typename DataType , class KeyType > void BSTree< DataType, KeyType >::insert_Helper (const DataType & *newDataItem,* BSTreeNode *& *ptr*) [protected]

Inserts a new node containing newDataItem. If it exists, then it updates the data item with newDataItem.

Precondition

none

Postcondition

Inserts a new node containing newDataItem

Parameters

DataType	variable and a BSTreeNode

Returns

none

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

isEmpty() function that checks if root is equal to NULL

Precondition

none

Postcondition

none

Parameters

none	

Returns

true is BSTree = NULL, false if not

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

overloaded assignment operator

Precondition

copyConstructor_Helper()

Postcondition

sets the current BSTree equal to another

Parameters

BSTree	variable

Returns

returns the current BSTree

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

remove() function that calls remove_Helper()

Precondition

remove_Helper()

Postcondition

none

Parameters

KeyType	variable

Returns

none

3.2.2.13 template<typename DataType , class KeyType > bool BSTree< DataType, KeyType >::remove_Helper (const KeyType & *deleteKey*, BSTreeNode *& *ptr*) [protected]

Delete data item with key deleteKey from the BSTree. If data is found, returns true. Else, it returns false.

Precondition

none

Postcondition

Deletes the data item from BSTree

Parameters

KeyType	variable and BSTreeNode

Returns

false if ptr is equal to NULL, true if it completes any of the 4 cases below, else it recursively searches itself

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

retrieve() function that calls retrieve_Helper()

Precondition

retrieve_Helper()

Postcondition

none

Parameters

KeyType	and Datatype variables

Returns

none

3.2.2.15 template<typename DataType , class KeyType > bool BSTree< DataType, KeyType >::retrieve_Helper (const KeyType & *searchKey,* DataType & *searchDataItem,* BSTreeNode * *ptr*) const [protected]

Searches BSTree for data item with the same key as searchKey; if found, replaces data item with searchDataItem and returns true.

Precondition

none

Postcondition

Searches BSTree using searchKey

Parameters

KeyType,DataType	variables & BSTreeNode

Returns

false if ptr is equal to NULL, true if searchDataItem is equal to data item, else recursively searches itself

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

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

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

writeKeys() function that calls writeKeys_Helper()

Precondition

writeKeys_Helper()

Postcondition

none

Parameters

none	

Returns

none

3.2.2.19 template<typename DataType , class KeyType > void BSTree< DataType, KeyType >::writeKeys_Helper (BSTreeNode * *ptr*) const [protected]

Outputs the keys of the data items in the BSTree. They are output in ascending order.

Precondition

none

Postcondition

Outputs keys of data items in BSTree

Parameters

BSTreeNode	

Returns

none

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

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

`#include <BSTree.h>`

Public Member Functions

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

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*)

default BSTreeNode constructor

Precondition

none

Postcondition

creates a BSTreeNode

Parameters

DataType	variable and two BSTreeNodes

Returns

none

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

Chapter 4

File Documentation

4.1 BSTree.cpp File Reference

An implementation file for a Binary Search Tree.

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

4.1.1 Detailed Description

An implementation file for a Binary Search Tree.

Author

Christopher Eichstedt

4.2 BSTree.h File Reference

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

Classes

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

4.3 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

4.3.1 Function Documentation

4.3.1.1 int main ( )

4.3.2 Variable Documentation

4.3.2.1 const long bytesPerRecord = 38

4.3.2.2 const int nameLength = 11

Index