Lab 11 Heap - Ernest Landrito

Generated by Doxygen 1.8.5

Mon Nov 18 2013 01:47:46

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Class Documentation

4.1 Greater< KeyType > Class Template Reference

Public Member Functions

- bool operator() (const KeyType &a, const KeyType &b) const

4.1.1 Member Function Documentation

4.1.1.1 template<typename KeyType = int> bool Greater< KeyType >::operator() (const KeyType & *a*, const KeyType & *b*) const [inline]

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

- test11.cpp

4.2 Heap< DataType, KeyType, Comparator > Class Template Reference

```
#include <Heap.h>
```

Public Member Functions

- Heap (int maxNumber=DEFAULT_MAX_HEAP_SIZE)
- Heap (const Heap &other)
- Heap & operator= (const Heap &other)
- ~Heap ()
- void insert (const DataType &newDataItem) throw (logic_error)
- DataType remove () throw (logic_error)
- void clear ()
- bool isEmpty () const
- bool isFull () const
- void showStructure () const
- void writeLevels () const

Static Public Attributes

- static const int DEFAULT_MAX_HEAP_SIZE = 10

4.2.1 Constructor & Destructor Documentation

4.2.1.1 template<typename DataType , typename KeyType , typename Comparator > Heap< DataType, KeyType, Comparator >::Heap (int *maxNumber* = DEFAULT_MAX_HEAP_SIZE)

Precondition

New Heap Class

Postcondition

dynamically allocated array of size maxNumber

Parameters

maxNumber	the size of the array to be formed

Algorithm:

- set size to zero
- set maxSize
- allocate array

Exceptional/Error Conditions:

- none

4.2.1.2 template<typename DataType , typename KeyType , typename Comparator > Heap< DataType, KeyType, Comparator >::Heap (const Heap< DataType, KeyType, Comparator > & *other*)

Precondition

new Heap class

Postcondition

a deep copy of source Heap

Parameters

other	Source HashTable to be deep copied

Algorithm:

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

4.2.1.3 template<typename DataType , typename KeyType , typename Comparator > Heap< DataType, KeyType, Comparator >::~Heap ( )

Precondition

a Heap

Postcondition

deallocated Heap

Algorithm:

- Use clear member function to deallocate the HashTable
- delete the array

4.2.2 Member Function Documentation

4.2.2.1 template<typename DataType , typename KeyType , typename Comparator > void Heap< DataType, KeyType, Comparator >::clear ( )

Precondition

a Heap

Postcondition

an empty heap

Algorithm:

- set the size of the heap to 0;

4.2.2.2 template<typename DataType, typename KeyType , typename Comparator > void Heap< DataType, KeyType, Comparator >::insert (const DataType & *newDataItem*) throw logic_error)

Precondition

a Heap

Postcondition

a new node inserted into the BSTree at the appropriate array location of the Heap

Parameters

newDataItem	Data to be inserted into the Heap

Algorithm:

- if the array isn't full
- loop until the compare is no longer fulfilled
- move the node up the tree if compare is fulfilled
- increase list size

4.2.2.3 template<typename DataType , typename KeyType , typename Comparator > bool Heap< DataType, KeyType, Comparator >::isEmpty ( ) const

Precondition

a Heap

Postcondition

returns if the heap is empty

Returns

returns if the heap is empty

Algorithm:

- return if the size is equal to 0

4.2.2.4 template<typename DataType , typename KeyType , typename Comparator > bool Heap< DataType, KeyType, Comparator >::isFull ( ) const

Precondition

a Heap

Postcondition

returns if the heap is full

Returns

returns if the heap is full

Algorithm:

- return if the size is equal to the max size

4.2.2.5 template<typename DataType , typename KeyType , typename Comparator > Heap< DataType, KeyType, Comparator > & Heap< DataType, KeyType, Comparator >::operator= (const Heap< DataType, KeyType, Comparator > & *other*)

Precondition

a Heap class

Postcondition

a deep copy of source heap

Parameters

source	Source Heap to be deep copied

Algorithm:

- if the heap is not empt and it isnt itself clear the heap
- copy the data

4.2.2.6 template<typename DataType , typename KeyType , typename Comparator > DataType Heap< DataType, KeyType, Comparator >::remove ( ) throw logic_error)

Precondition

a Heap

Postcondition

Heap with root node removed

Returns

returns the data at the front of the array

Algorithm:

- Throw the logic error if the heap is empty
- reduce size
- set return value
- move the item at the end of the list to the front
- move the data item down through the list switching with things it compares to

4.2.2.7 template<typename DataType , typename KeyType , typename Comparator > void Heap< DataType, KeyType, Comparator >::showStructure ( ) const

4.2.2.8 template<typename DataType , typename KeyType , typename Comparator > void Heap< DataType, KeyType, Comparator >::writeLevels ( ) const

Precondition

a Heap

Postcondition

prints each level of the heap

Algorithm:

- set how many to print per level to 1
- set how many printed to 0
- loop until the every item is printed;
- after printing each level multiply to print by 2 and return printed counter to 0

4.2.3 Member Data Documentation

4.2.3.1 template<typename DataType, typename KeyType = int, typename Comparator = Less<KeyType>> const int Heap< DataType, KeyType, Comparator >::DEFAULT_MAX_HEAP_SIZE = 10 [static]

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

- Heap.h
- Heap.cpp
- Heap2.cpp
- show11.cpp

4.3 Less< KeyType > Class Template Reference

`#include <Heap.h>`

Public Member Functions

- bool operator() (const KeyType &a, const KeyType &b) const

4.3.1 Member Function Documentation

4.3.1.1 template<typename KeyType = int> bool Less< KeyType >::operator() (const KeyType & *a*, const KeyType & *b*) const [inline]

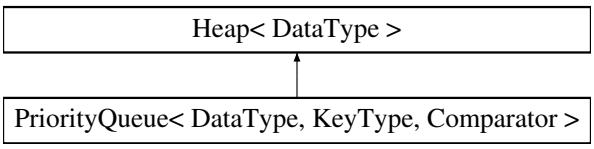
The documentation for this class was generated from the following file:

- Heap.h

4.4 PriorityQueue< DataType, KeyType, Comparator > Class Template Reference

`#include <PriorityQueue.h>`

Inheritance diagram for PriorityQueue< DataType, KeyType, Comparator >:



Public Member Functions

- PriorityQueue (int maxNumber=defMaxQueueSize)
- PriorityQueue (const Heap< DataType, KeyType, Comparator > &other)
- PriorityQueue & operator= (const PriorityQueue &other)
- ~PriorityQueue ()
- void enqueue (const DataType &newDataItem)
- DataType dequeue ()
- void clear ()
- bool isEmpty ()
- bool isFull ()

Additional Inherited Members

4.4.1 Constructor & Destructor Documentation

4.4.1.1 template<typename DataType , typename KeyType = int, typename Comparator = Less<KeyType>> PriorityQueue< DataType, KeyType, Comparator >::PriorityQueue (int *maxNumber* = defMaxQueueSize) [inline]

4.4.1.2 template<typename DataType , typename KeyType = int, typename Comparator = Less<KeyType>> PriorityQueue< DataType, KeyType, Comparator >::PriorityQueue (const Heap< DataType, KeyType, Comparator > & *other*) [inline]

4.4.1.3 template<typename DataType , typename KeyType = int, typename Comparator = Less<KeyType>> PriorityQueue< DataType, KeyType, Comparator >::~PriorityQueue ( )

4.4.2 Member Function Documentation

4.4.2.1 template<typename DataType , typename KeyType , typename Comparator > void PriorityQueue< DataType, KeyType, Comparator >::clear ( )

Precondition

a Priority Queue

Postcondition

a cleared Priority Queue

Algorithm:

- use heap clear function

4.4.2.2 template<typename DataType , typename KeyType , typename Comparator > DataType PriorityQueue< DataType, KeyType, Comparator >::dequeue ( )

Precondition

a Priority Queue

Postcondition

Priority Queue with front removed

Returns

returns the data at the front of the Queue

Algorithm:

- use heap remove function

4.4.2.3 template<typename DataType , typename KeyType , typename Comparator > void PriorityQueue< DataType, KeyType, Comparator >::enqueue (const DataType & *newDataItem*)

Precondition

a Priority Queue

Postcondition

a new data inserted at the appropriate array location of the Heap

Parameters

newDataItem	Data to be inserted into the Heap

Algorithm:

- use the Heap insert function

4.4.2.4 template<typename DataType , typename KeyType , typename Comparator > bool PriorityQueue< DataType, KeyType, Comparator >::isEmpty ( )

Precondition

a Priority Queue

Postcondition

returns if the Priority Queue is empty

Returns

returns if the Priority Queue is empty

Algorithm:

- use heap is empty function

4.4.2.5 template<typename DataType , typename KeyType , typename Comparator > bool PriorityQueue< DataType, KeyType, Comparator >::isFull ( )

Precondition

a Priority Queue

Postcondition

returns if the Priority Queue is full

Returns

returns if the Priority Queue is full

Algorithm:

- use heap is empty function

4.4.2.6 template<typename DataType , typename KeyType , typename Comparator > PriorityQueue< DataType, KeyType, Comparator > & PriorityQueue< DataType, KeyType, Comparator >::operator= (const PriorityQueue< DataType, KeyType, Comparator > & *other*)

Precondition

a Heap class

Postcondition

a deep copy of source heap

Parameters

source	Source Heap to be deep copied

Algorithm:

- if the heap is not empt and it isnt itself clear the heap
- copy the data

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

- PriorityQueue.h
- PriorityQueue.cpp

4.5 TaskData Struct Reference

Public Member Functions

- int getPriority () const
- int getPriority () const

Public Attributes

- int priority
- int arrived

4.5.1 Member Function Documentation

4.5.1.1 int TaskData::getPriority ( ) const [inline]

4.5.1.2 int TaskData.getPriority ( ) const [inline]

4.5.2 Member Data Documentation

4.5.2.1 int TaskData::arrived

4.5.2.2 int TaskData::priority

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

- ossim.cpp
- ossim.cs

4.6 TestData Class Reference

Public Member Functions

- void setPriority (int newPriority)
- int getPriority () const
- void setPriority (int newPriority)
- int getPriority () const

4.6.1 Member Function Documentation

4.6.1.1 int TestData::getPriority ( ) const [inline]

4.6.1.2 int TestData::getPriority ( ) const [inline]

4.6.1.3 void TestData::setPriority (int *newPriority*) [inline]

4.6.1.4 void TestData::setPriority (int *newPriority*) [inline]

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

- test11hs.cpp
- test11pq.cpp

4.7 TestDataItem< KeyType > Class Template Reference

Public Member Functions

- TestDataItem ()
- void setPriority (KeyType newPty)
- KeyType getPriority () const

4.7.1 Constructor & Destructor Documentation

4.7.1.1 template<typename KeyType > TestDataItem< KeyType >::TestDataItem ( ) [inline]

4.7.2 Member Function Documentation

4.7.2.1 template<typename KeyType > KeyType TestDataItem< KeyType >::getPriority ( ) const [inline]

4.7.2.2 template<typename KeyType > void TestDataItem< KeyType >::setPriority (KeyType *newPty*) [inline]

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

- test11.cpp

Chapter 5

File Documentation

5.1 config.h File Reference

Macros

- #define LAB11_TEST1 1

5.1.1 Macro Definition Documentation

5.1.1.1 #define LAB11_TEST1 1

Heap class configuration file. Activate test #N by defining the corresponding LAB11_TESTN to have the value 1.

5.2 Heap.cpp File Reference

```
#include <stdexcept>
#include <iostream>
#include "Heap.h"
```

5.3 Heap.h File Reference

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

Classes

- class Less< KeyType >
- class Heap< DataType, KeyType, Comparator >

5.4 Heap2.cpp File Reference

```
#include <stdexcept>
#include <iostream>
#include "Heap.h"
```

5.5 heapsort.cs File Reference

Functions

- void moveDown (DataType dataItems[], int root, int size)
- void heapSort (DataType dataItems[], int size)

5.5.1 Function Documentation

5.5.1.1 void heapSort (DataType *dataItems[],* int *size*)

5.5.1.2 void moveDown (DataType *dataItems[],* int *root,* int *size*)

5.6 ossim.cpp File Reference

```
#include <iostream>
#include <cstdlib>
#include "PriorityQueue.cpp"
```

Classes

- struct TaskData

Functions

- int main ()

5.6.1 Function Documentation

5.6.1.1 int main ( )

5.7 ossim.cs File Reference

```
#include <iostream>
#include <cstdlib>
#include "PriorityQueue.cpp"
```

Classes

- struct TaskData

Functions

- int main ()

5.7.1 Function Documentation

5.7.1.1 int main ( )

5.8 PriorityQueue.cpp File Reference

```
#include <stdexcept>
#include <iostream>
#include "PriorityQueue.h"
```

5.9 PriorityQueue.h File Reference

```
#include <stdexcept>
#include <iostream>
#include "Heap.cpp"
```

Classes

- class PriorityQueue< DataType, KeyType, Comparator >

Variables

- const int defMaxQueueSize = 10

5.9.1 Variable Documentation

5.9.1.1 const int defMaxQueueSize = 10

5.10 show11.cpp File Reference

5.11 test11.cpp File Reference

```
#include <iostream>
#include <string>
#include <cctype>
#include "Heap.cpp"
#include "config.h"
```

Classes

- class TestDataItem< KeyType >
- class Greater< KeyType >

Functions

- void printHelp ()
- int main ()

5.11.1 Function Documentation

5.11.1.1 int main ( )

5.11.1.2 void printHelp ( )

5.12 test11hs.cpp File Reference

```
#include <iostream>
#include "heapsort.cpp"
```

Classes

- class TestData

Functions

- int main ()

Variables

- const int MAX_NUM_DATA_ITEMS = 10

5.12.1 Function Documentation

5.12.1.1 int main ( )

5.12.2 Variable Documentation

5.12.2.1 const int MAX_NUM_DATA_ITEMS = 10

5.13 test11pq.cpp File Reference

```
#include <iostream>
#include <cctype>
#include "PriorityQueue.cpp"
```

Classes

- class TestData

Functions

- void printHelp ()
- int main ()

5.13.1 Function Documentation

5.13.1.1 int main ( )

5.13.1.2 void printHelp ( )

Index