Project 7 - Queue ADT

AUTHOR

Version 2.001

09/10/2014

Table of Contents

Project 7 (Queue ADT)

This program will implement the Linked List version of a Queue.

Author:

Saharath Kleips

Version:

2.00

The specifications of this project match those of the book C++ Data Structures - A Laboratory Course (3rd Edition) Project 7. Data items in the Queue are of generic type DataType. The Queue data items are linearly ordered from least recently added (front) to most recently added (rear.) Data items are inserted at the rear of the Queue (enqueue) and are removed from the front of the Queue (dequeue.)

Todo List

Member QueueLinked< DataType >::isFull () const

Implement overflow check. Currently always returns false.

Class Index

Class Hierarchy

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

Class Index

Class List

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

File Index

File List

Here is a list of all documented files with brief descriptions:

Class Documentation

Queue< DataType > Class Template Reference

Inheritance diagram for Queue< DataType >:

IMAGE

Public Member Functions

virtual void **enqueue** (const DataType &newDataItem)=0 throw (logic_error)
virtual DataType **dequeue** ()=0 throw (logic_error)
virtual void **clear** ()=0
virtual bool **isEmpty** () const =0
virtual bool **isFull** () const =0
virtual void **showStructure** () const =0

Static Public Attributes

static const int **MAX_QUEUE_SIZE** = 8

Detailed Description

template<typename DataType>class Queue< DataType >

Definition at line 21 of file Queue.h.

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

1 Queue.h

QueueLinked< DataType > Class Template Reference

Inheritance diagram for QueueLinked< DataType >:

IMAGE

Classes

class **QueueNode**

Public Member Functions

QueueLinked (int maxNumber=Queue< DataType >::MAX_QUEUE_SIZE)

The default constructor that creates an empty Queue.

QueueLinked (const QueueLinked &other)

The copy constructor that initializes the Queue to be equivalent to the other Queue object parameter.

QueueLinked & operator= (const QueueLinked &other)

The overloaded assignment operator that sets the Queue to be equivalent to the other Queue object parameter and returns a reference to the modified Queue.

~QueueLinked ()

The destructor that deallocates the memory used to store the Queue.

void enqueue (const DataType &newDataItem) throw (logic_error)

Inserts newDataItem at the rear of the Queue.

DataType dequeue () throw (logic_error)

Removes the data item that was least recently added from the Queue and returns it.

void clear ()

Removes all data items in the Queue.

bool isEmpty () const

Returns true if the Queue is empty.

bool isFull () const

Returns true if the Queue is full.

void putFront (const DataType &newDataItem) throw (logic_error)

Inserts newDataItem at the front of the Queue.

DataType getRear () throw (logic_error)

Removes the most recently added data item from the Queue and returns it.

int getLength () const

Returns the number of data items in the Queue.

void **showStructure** () const

Detailed Description

template<typename DataType>class QueueLinked< DataType >

Definition at line 11 of file QueueLinked.h.

Constructor & Destructor Documentation

template<class DataType > QueueLinked< DataType >::QueueLinked (int *maxNumber* = Queue<DataType>::MAX_QUEUE_SIZE)

The default constructor that creates an empty Queue.

Will allocate neough memory for the Queue containing maxNumber data items (if necessary.)

Parameters:

maxNumber	is provided for call compatibility with the array implementation.

template<class DataType > QueueLinked< DataType >::QueueLinked (const QueueLinked< DataType > & *other*)

The copy constructor that initializes the Queue to be equivalent to the other Queue object parameter.

Parameters:

Definition at line 32 of file QueueLinked.cpp.

other	is the Queue to be equivalent to this Queue.

operator=(const QueueLinked<DataType>& other)

Definition at line 44 of file QueueLinked.cpp.

template<class DataType > QueueLinked< DataType >::~QueueLinked ()

The destructor that deallocates the memory used to store the Queue.

See also:

clear()

Definition at line 75 of file QueueLinked.cpp.

Member Function Documentation

template<class DataType > void QueueLinked< DataType >::clear () [virtual]

Removes all data items in the Queue.

It will deallocate memory used for the nodes to store the data. Accomplishes this by iterating with dequeue().

See also:

dequeue()

Implements Queue< DataType >.

Definition at line 125 of file QueueLinked.cpp.

template<class DataType > DataType QueueLinked< DataType >::dequeue () throw (logic_error) [virtual]

Removes the data item that was least recently added from the Queue and returns it.

Precondition:

Queue is not empty.

Returns:

DataType is the data removed from the Queue.

Implements Queue< DataType >.

Definition at line 104 of file QueueLinked.cpp.

template<class DataType > void QueueLinked< DataType >::enqueue (const DataType & *newDataItem*) throw (logic_error) [virtual]

Inserts newDataItem at the rear of the Queue.

Precondition:

Queue is not full.

Parameters:

See also:

newDataItem	is the data to be added to the Queue.

template<class DataType > int QueueLinked< DataType >::getLength () const

Returns the number of data items in the Queue.

Returns:

int is the number of data items in a the Queue.

Definition at line 214 of file QueueLinked.cpp.

template<class DataType > DataType QueueLinked< DataType >::getRear () throw (logic_error)

Removes the most recently added data item from the Queue and returns it.

The remainder of the Queue is left unchanged.

Precondition:

The Queue is not empty.

Returns:

DataType is the data removed from the Queue.

Definition at line 181 of file QueueLinked.cpp.

template<class DataType > bool QueueLinked< DataType >::isEmpty () const [virtual]

Returns true if the Queue is empty.

Otherwise, returns false.

Returns:

The Queue is empty or not.

Implements Queue< DataType >.

Definition at line 137 of file QueueLinked.cpp.

template<class DataType > bool QueueLinked< DataType >::isFull () const [virtual]

Returns true if the Queue is full.

Otherwise, returns false.

Returns:

The Queue is full or not.

Todo:

Implement overflow check. Currently always returns false.

Implements Queue< DataType >.

Definition at line 151 of file QueueLinked.cpp.

template<class DataType > QueueLinked< DataType > & QueueLinked< DataType >::operator= (const QueueLinked< DataType > & *other*)

The overloaded assignment operator that sets the Queue to be equivalent to the other Queue object parameter and returns a reference to the modified Queue.

Parameters:

Implements Queue< DataType >.

Definition at line 86 of file QueueLinked.cpp.

other	is the Queue that this Queue will be made equivalent to.

The reference to this object.

Definition at line 58 of file QueueLinked.cpp.

template<class DataType > void QueueLinked< DataType >::putFront (const DataType & *newDataItem*) throw (logic_error)

Inserts newDataItem at the front of the Queue.

The order of preexisting data items is left unchanged.

Precondition:

The Queue is not full.

Parameters:

Returns:

newDataItem	is the data item to be added to the Queue.

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

2 QueueLinked.h
3 QueueLinked.cpp

File Documentation

QueueLinked.cpp File Reference

```
#include "QueueLinked.h"
```

Detailed Description

Definition in file QueueLinked.cpp.

storesim.cpp File Reference

```
#include <iostream>
#include <iomanip>
#include <cstdlib>
#include <ctime>
#include "config.h"
#include "QueueLinked.cpp"
```

Functions

int **main** ()

Detailed Description

Definition in file storesim.cpp.

Index

INDE