My Project

Generated by Doxygen 1.8.13

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 documented files with brief descriptions:

Chapter 3

Class Documentation

3.1 LinkList< T > Class Template Reference

We define a class called LinkList and define some functions using the class node.

Public Member Functions

- void insertFront (T item)
- void insertRear (T item)
- void removeFront ()
- void removeRear ()

Static Public Member Functions

- static void main (String a[ ])

  The driving function to create a double ended queue.

3.1.1 Detailed Description

We define a class called LinkList and define some functions using the class node.

Here we create queue data structure,where adding and removing elements is possible at both ends using linkedlist.

3.1.2 Member Function Documentation

3.1.2.1 insertFront()

```
void LinkList< T >.insertFront (
             T item )   [inline]
```

To add an element at the beginning of the queue.We create a node with the 'item' input as its value and as the new node is nd and is at the front so its previous node is set to null and the next node is the front node previously,if the queue not empty,else it is also set to null. if previously queue was not empty then the preivous front node's previous node is set to the present front node.

3.1.2.2 insertRear()

```
void LinkList< T >.insertRear (
             T item )  [inline]
```

To insert an element at the end of the queue.We create a new node nd with item as its value and set its previous node to the previous end node if queue was not empty else null and the next node to null. And the previous rear node would have its next node as nd. /∗

Parameters

item	-> element to insert at the end .

Returns

void

3.1.2.3 main()

```
static void LinkList< T >.main (
             String a[] )  [inline], [static]
```

The driving function to create a double ended queue.

We define deque as linklist variable of integers.

3.1.2.4 removeFront()

```
void LinkList< T >.removeFront ( )  [inline]
```

To remove the item from beginning of the queue If queue was not empty then remove the node at front,else unable to remove message is printed.

3.1.2.5 removeRear()

```
void LinkList< T >.removeRear ( )  [inline]
```

To remove the last element of the double ended queue If queue was not empty then remove node at the end ,else print unable to remove message.

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

- LinkList.java

Chapter 4

File Documentation

4.1 LinkList.java File Reference

Implementing the Linkedlists We use this to remove or add elements at the both ends in constant time.

Classes

- class LinkList$<$ T $>$

  We define a class called LinkList and define some functions using the class node.
- class **Node**$<$ **T** $>$

  Define a class called 'node' to store information abt the value at that node and reference to previous node and the next node.

4.1.1 Detailed Description

Implementing the Linkedlists We use this to remove or add elements at the both ends in constant time.