

three-level linked list

0.0.1

Sun May 2 22:10:20 2010

Contents

1	Specification (hungarian)	1
1.1	Pontosított feladat-specifikáció	1
2	Class Index	3
2.1	Class Hierarchy	3
3	Class Index	5
3.1	Class List	5
4	File Index	7
4.1	File List	7
5	Class Documentation	9
5.1	Cmp< T > Class Template Reference	9
5.1.1	Detailed Description	9
5.1.2	Member Function Documentation	9
5.1.2.1	eq	9
5.1.2.2	lt	9
5.2	LinkedList< T >::iterator Class Reference	10
5.2.1	Detailed Description	11
5.2.2	Constructor & Destructor Documentation	11
5.2.2.1	iterator	11
5.2.2.2	iterator	11
5.2.3	Member Function Documentation	11
5.2.3.1	operator!=	11
5.2.3.2	operator*	11
5.2.3.3	operator++	11
5.2.3.4	operator++	11
5.2.3.5	operator->	12
5.2.3.6	operator==	12

5.2.4	Friends And Related Function Documentation	12
5.2.4.1	LinkedList< T >	12
5.2.5	Member Data Documentation	12
5.2.5.1	act	12
5.3	MainLinkedList< R, S, T >::iterator Class Reference	13
5.3.1	Detailed Description	14
5.3.2	Constructor & Destructor Documentation	14
5.3.2.1	iterator	14
5.3.2.2	iterator	14
5.3.3	Member Function Documentation	14
5.3.3.1	getList	14
5.3.3.2	operator!=	15
5.3.3.3	operator*	15
5.3.3.4	operator++	15
5.3.3.5	operator++	15
5.3.3.6	operator->	15
5.3.3.7	operator==	15
5.3.4	Friends And Related Function Documentation	15
5.3.4.1	MainLinkedList< R, S, T >	15
5.3.5	Member Data Documentation	15
5.3.5.1	act	15
5.4	RootLinkedList< R, T >::iterator Class Reference	16
5.4.1	Detailed Description	17
5.4.2	Constructor & Destructor Documentation	17
5.4.2.1	iterator	17
5.4.2.2	iterator	17
5.4.3	Member Function Documentation	17
5.4.3.1	getList	17
5.4.3.2	operator!=	18
5.4.3.3	operator*	18
5.4.3.4	operator++	18
5.4.3.5	operator++	18
5.4.3.6	operator->	18
5.4.3.7	operator==	18
5.4.4	Friends And Related Function Documentation	18
5.4.4.1	RootLinkedList< R, T >	18

5.4.5	Member Data Documentation	18
5.4.5.1	act	18
5.5	LinkedList< T > Class Template Reference	19
5.5.1	Detailed Description	20
5.5.2	Constructor & Destructor Documentation	20
5.5.2.1	LinkedList	20
5.5.2.2	~LinkedList	21
5.5.3	Member Function Documentation	21
5.5.3.1	begin	21
5.5.3.2	deleteall	21
5.5.3.3	end	21
5.5.3.4	insert	21
5.5.3.5	print	21
5.5.3.6	remove	21
5.5.4	Friends And Related Function Documentation	22
5.5.4.1	LinkedList< T >::iterator	22
5.5.5	Member Data Documentation	22
5.5.5.1	endItem	22
5.5.5.2	firstItem	22
5.6	ListException Class Reference	23
5.6.1	Constructor & Destructor Documentation	23
5.6.1.1	ListException	23
5.6.1.2	~ListException	23
5.6.2	Member Function Documentation	23
5.6.2.1	what	23
5.7	MainLinkedList< R, S, T >::ListItem Struct Reference	24
5.7.1	Detailed Description	24
5.7.2	Constructor & Destructor Documentation	25
5.7.2.1	ListItem	25
5.7.3	Member Data Documentation	25
5.7.3.1	data	25
5.7.3.2	next	25
5.7.3.3	nextList	25
5.8	RootLinkedList< R, T >::ListItem Struct Reference	26
5.8.1	Detailed Description	26
5.8.2	Constructor & Destructor Documentation	27

5.8.2.1	ListItem	27
5.8.3	Member Data Documentation	27
5.8.3.1	data	27
5.8.3.2	next	27
5.8.3.3	nextList	27
5.9	LinkedList< T >::ListItem Struct Reference	28
5.9.1	Detailed Description	28
5.9.2	Constructor & Destructor Documentation	28
5.9.2.1	ListItem	28
5.9.3	Member Data Documentation	28
5.9.3.1	data	28
5.9.3.2	next	29
5.10	MainLinkedList< R, S, T > Class Template Reference	30
5.10.1	Detailed Description	31
5.10.2	Constructor & Destructor Documentation	31
5.10.2.1	MainLinkedList	31
5.10.2.2	~MainLinkedList	31
5.10.3	Member Function Documentation	31
5.10.3.1	begin	31
5.10.3.2	deleteall	31
5.10.3.3	end	32
5.10.3.4	insert	32
5.10.3.5	insert	32
5.10.3.6	insert	32
5.10.3.7	print	32
5.10.3.8	remove	32
5.10.3.9	remove	32
5.10.3.10	remove	32
5.10.4	Friends And Related Function Documentation	32
5.10.4.1	MainLinkedList< R, S, T >::iterator	32
5.10.5	Member Data Documentation	32
5.10.5.1	endItem	32
5.10.5.2	firstItem	33
5.11	RootLinkedList< R, T > Class Template Reference	34
5.11.1	Detailed Description	35
5.11.2	Constructor & Destructor Documentation	35

5.11.2.1	RootLinkedList	35
5.11.2.2	~RootLinkedList	35
5.11.3	Member Function Documentation	35
5.11.3.1	begin	35
5.11.3.2	deleteall	36
5.11.3.3	end	36
5.11.3.4	insert	36
5.11.3.5	insert	36
5.11.3.6	print	36
5.11.3.7	remove	36
5.11.3.8	remove	36
5.11.4	Friends And Related Function Documentation	37
5.11.4.1	RootLinkedList< R, T >::iterator	37
5.11.5	Member Data Documentation	37
5.11.5.1	endItem	37
5.11.5.2	firstItem	37
6	File Documentation	39
6.1	list.h File Reference	39
6.2	list.hpp File Reference	41
6.3	main.cpp File Reference	42
6.3.1	Function Documentation	42
6.3.1.1	main	42

Chapter 1

Specification (hungarian)

„Készítsen GENERIKUS rendezett 3 szintű fésűs listát!

Valósítsa meg az összes értelmes műveletet operátor átdefiniálással (overload), de nem kell ragaszkodni mindenáron az operátorok átdefiniálásához! Amennyiben lehetséges használjon iterátort!

Specifikáljon egy egyszerű tesztfeladatot, amiben fel tudja használni az elkészített adatszerkezetet! A tesztprogramot külön modulként fordított programmal oldja meg! A programmal mutassa be a generikus szerkezet használatát több egyszerű adathalmazon, amit fájlból olvas be! A megoldáshoz NE használjon STL tárolót vagy algoritmust!

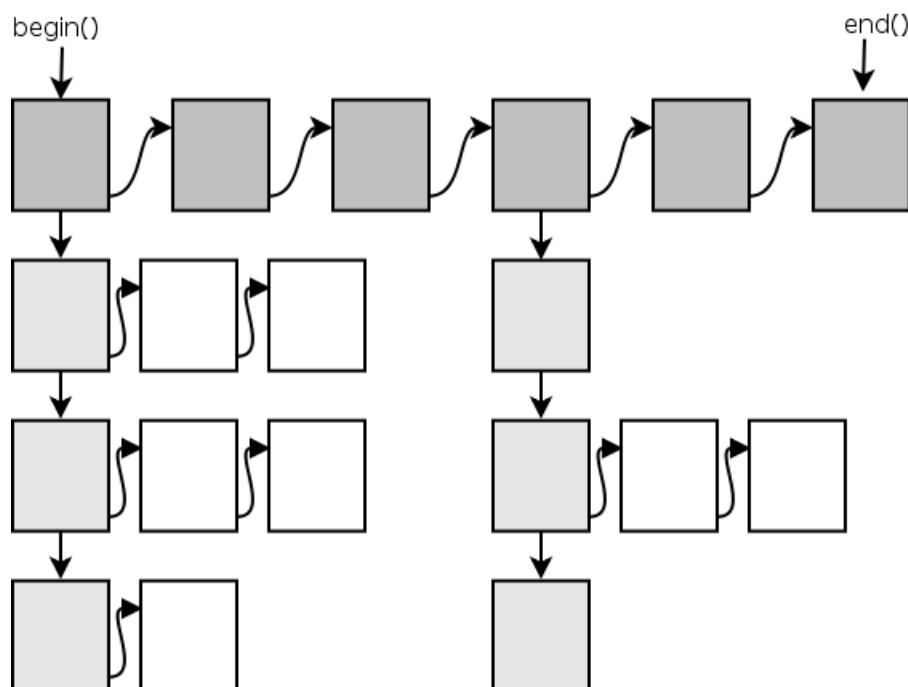
A tesztprogramot úgy specifikálja, hogy az parancssoros batch alkalmazásként (is) működjön, azaz a szabványos bemenetről olvasson, és a szabványos kimenetre, és/vagy a hibakimenetre írjon! Amennyiben a feladat teszteléséhez fájlból, vagy fájlokból kell input adatot olvasnia, úgy a fájl neve *.dat alakú legyen! ”

1.1 Pontosított feladat-specifikáció

A generikus fésűs listához C++ sablonokat fogok használni. A következő operátorok legyenek túlterhelve:

- *operator+=* - egy új elemet hozzáad a listához
- *operator-=* - egy adott elemet kikeres és töröl a listából
- *operator==* - két lista egyenlő
- *operator!=* - két lista nem egyenlő

A 3 szintű fésűs listát az alábbi ábra reprezentálja:



A tároló bejárásához iterátort fogok használni.

Megvalósítandó funkciók:

- Elem hozzáadása a listához
- Elem törlése a listából
- Egy adott elem megkeresése és mutatójának visszaadása

Az iterátor függvényei:

- *begin()* - Első elem mutatójának visszaadása
- *end()* - Utolsó elem mutatójának visszaadása
- *next()* - Következő elem mutatójának visszaadása

A tesztprogram képes lesz szabványos bemenetről és fájlból is olvasni. Mindkét esetben a formátum a következő: Egy sor egy elemet reprezentál. A három szintnek megfelelően 3 elem lehet egy sorban (1, 2 vagy 3).

első szint|második szint|harmadik szint
első szint|második szint

Chapter 2

Class Index

2.1 Class Hierarchy

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

Cmp< T >	9
LinkedList< T >::iterator	10
RootLinkedList< R, T >::iterator	16
MainLinkedList< R, S, T >::iterator	13
LinkedList< T >	19
LinkedList< R >	19
RootLinkedList< R, T >	34
RootLinkedList< R, S >	34
MainLinkedList< R, S, T >	30
ListException	23
LinkedList< T >::ListItem	28
MainLinkedList< R, S, T >::ListItem	24
RootLinkedList< R, T >::ListItem	26
LinkedList< T >::ListItem< R >	28

Chapter 3

Class Index

3.1 Class List

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

Cmp< T > (Compare template class. Compare class containing 2 predicate functions (equals, less than) In some cases specialization is needed)	9
LinkedList< T >::iterator (Iterator class)	10
MainLinkedList< R, S, T >::iterator (Iterator class)	13
RootLinkedList< R, T >::iterator (Iterator class)	16
LinkedList< T > (Linked list class. Simple sorted linked list class)	19
ListException	23
MainLinkedList< R, S, T >::ListItem (Three-level Linked list's item)	24
RootLinkedList< R, T >::ListItem (Root Linked list's item)	26
LinkedList< T >::ListItem (Linked list's item)	28
MainLinkedList< R, S, T > (Three-level linked list)	30
RootLinkedList< R, T > (2 level Root Linked List. Each list item has one child list)	34

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

list.h	39
list.hpp	41
main.cpp	42

Chapter 5

Class Documentation

5.1 `Cmp< T >` Class Template Reference

Compare template class. Compare class containing 2 predicate functions (equals, less than) In some cases specialization is needed.

```
#include <list.h>
```

Static Public Member Functions

- static bool `eq` (const T &a, const T &b)
- static bool `lt` (const T &a, const T &b)

5.1.1 Detailed Description

```
template<typename T> class Cmp< T >
```

Compare template class. Compare class containing 2 predicate functions (equals, less than) In some cases specialization is needed.

5.1.2 Member Function Documentation

5.1.2.1 `template<typename T > static bool Cmp< T >::eq (const T & a, const T & b)`
[inline, static]

5.1.2.2 `template<typename T > static bool Cmp< T >::lt (const T & a, const T & b)` [inline, static]

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

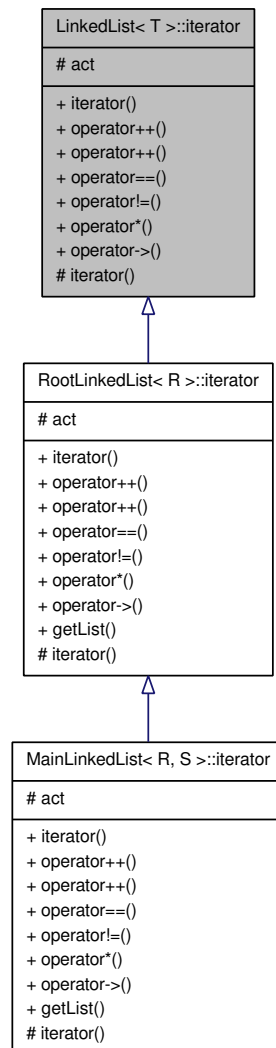
- [list.h](#)

5.2 LinkedList< T >::iterator Class Reference

Iterator class.

```
#include <list.h>
```

Inheritance diagram for LinkedList< T >::iterator:



Public Member Functions

- `iterator` (void)
- `iterator operator++` (void)
- `iterator operator++` (int)
- `bool operator==` (const `iterator` &i) const
- `bool operator!=` (const `iterator` &i) const
- `T & operator*` (void)
- `T * operator->` (void)

Protected Member Functions

- `iterator (ListItem *li)`

Protected Attributes

- `ListItem * act`

Friends

- class `LinkedList< T >`

5.2.1 Detailed Description

`template<typename T> class LinkedList< T >::iterator`

Iterator class.

5.2.2 Constructor & Destructor Documentation

5.2.2.1 `template<typename T> LinkedList< T >::iterator::iterator (void) [inline]`

Reimplemented in `RootLinkedList< R, T >::iterator`, and `MainLinkedList< R, S, T >::iterator`.

5.2.2.2 `template<typename T> LinkedList< T >::iterator::iterator (ListItem * li) [inline, protected]`

5.2.3 Member Function Documentation

5.2.3.1 `template<typename T> bool LinkedList< T >::iterator::operator!=(const iterator & i) const`

5.2.3.2 `template<typename T> T & LinkedList< T >::iterator::operator* (void) [inline]`

Reimplemented in `RootLinkedList< R, T >::iterator`, and `MainLinkedList< R, S, T >::iterator`.

5.2.3.3 `template<typename T> LinkedList< T >::iterator LinkedList< T >::iterator::operator++ (int) [inline]`

Reimplemented in `RootLinkedList< R, T >::iterator`, and `MainLinkedList< R, S, T >::iterator`.

5.2.3.4 `template<typename T> LinkedList< T >::iterator LinkedList< T >::iterator::operator++ (void) [inline]`

Reimplemented in `RootLinkedList< R, T >::iterator`, and `MainLinkedList< R, S, T >::iterator`.

5.2.3.5 `template<typename T> T * LinkedList< T >::iterator::operator-> (void) [inline]`

Reimplemented in [RootLinkedList< R, T >::iterator](#), and [MainLinkedList< R, S, T >::iterator](#).

5.2.3.6 `template<typename T> bool LinkedList< T >::iterator::operator==(const iterator & i) const`

5.2.4 Friends And Related Function Documentation

5.2.4.1 `template<typename T> friend class LinkedList< T > [friend]`

5.2.5 Member Data Documentation

5.2.5.1 `template<typename T> ListItem* LinkedList< T >::iterator::act [protected]`

Reimplemented in [RootLinkedList< R, T >::iterator](#), and [MainLinkedList< R, S, T >::iterator](#).

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

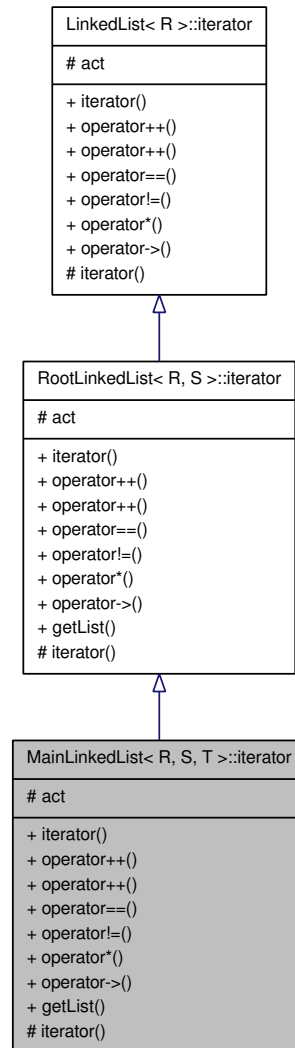
- [list.h](#)
- [list.hpp](#)

5.3 MainLinkedList< R, S, T >::iterator Class Reference

Iterator class.

```
#include <list.h>
```

Inheritance diagram for MainLinkedList< R, S, T >::iterator:



Public Member Functions

- `iterator` (void)
- `iterator operator++` (void)
- `iterator operator++` (int)
- `bool operator==` (const `iterator` &i) const
- `bool operator!=` (const `iterator` &i) const
- `R & operator*` (void)
- `R * operator->` (void)
- `RootLinkedList< S, T > * getList` (void)

Containing list.

Protected Member Functions

- [iterator](#) ([ListItem](#) *li)

Protected Attributes

- [ListItem](#) * act

Friends

- class [MainLinkedList](#)< [R](#), [S](#), [T](#) >

5.3.1 Detailed Description

`template<class R, class S, class T> class MainLinkedList< R, S, T >::iterator`

Iterator class.

5.3.2 Constructor & Destructor Documentation

5.3.2.1 `template<class R , class S , class T > MainLinkedList< R, S, T >::iterator::iterator (void) [inline]`

Reimplemented from [RootLinkedList](#)< [R](#), [T](#) >::iterator.

5.3.2.2 `template<class R , class S , class T > MainLinkedList< R, S, T >::iterator::iterator (ListItem * li) [inline, protected]`

5.3.3 Member Function Documentation

5.3.3.1 `template<typename R , typename S , typename T > RootLinkedList< S, T > * MainLinkedList< R, S, T >::iterator::getList (void) [inline]`

Containing list.

Returns

pointer to the containing list

Reimplemented from [RootLinkedList](#)< [R](#), [T](#) >::iterator.

5.3.3.2 `template<typename R , typename S , typename T > bool MainLinkedList< R, S, T >::iterator::operator!=(const iterator & i) const [inline]`

5.3.3.3 `template<typename R , typename S , typename T > R & MainLinkedList< R, S, T >::iterator::operator*(void) [inline]`

Reimplemented from [RootLinkedList< R, T >::iterator](#).

5.3.3.4 `template<typename R , typename S , typename T > MainLinkedList< R, S, T >::iterator MainLinkedList< R, S, T >::iterator::operator++(int) [inline]`

Reimplemented from [RootLinkedList< R, T >::iterator](#).

5.3.3.5 `template<typename R , typename S , typename T > MainLinkedList< R, S, T >::iterator MainLinkedList< R, S, T >::iterator::operator++(void) [inline]`

Reimplemented from [RootLinkedList< R, T >::iterator](#).

5.3.3.6 `template<typename R , typename S , typename T > R * MainLinkedList< R, S, T >::iterator::operator->(void) [inline]`

Reimplemented from [RootLinkedList< R, T >::iterator](#).

5.3.3.7 `template<typename R , typename S , typename T > bool MainLinkedList< R, S, T >::iterator::operator==(const iterator & i) const [inline]`

5.3.4 Friends And Related Function Documentation

5.3.4.1 `template<class R , class S , class T > friend class MainLinkedList< R, S, T > [friend]`

5.3.5 Member Data Documentation

5.3.5.1 `template<class R , class S , class T > ListItem* MainLinkedList< R, S, T >::iterator::act [protected]`

Reimplemented from [RootLinkedList< R, T >::iterator](#).

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

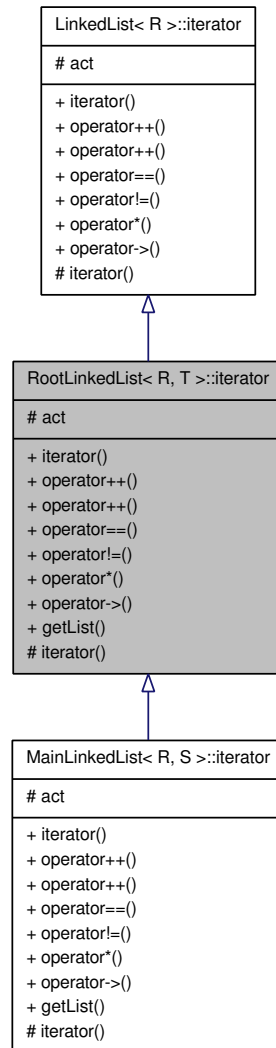
- [list.h](#)
- [list.hpp](#)

5.4 RootLinkedList< R, T >::iterator Class Reference

Iterator class.

```
#include <list.h>
```

Inheritance diagram for RootLinkedList< R, T >::iterator:



Public Member Functions

- `iterator` (void)
- `iterator operator++` (void)
- `iterator operator++` (int)
- `bool operator==` (const `iterator` &i) const
- `bool operator!=` (const `iterator` &i) const
- `R & operator*` (void)
- `R * operator->` (void)
- `LinkedList< T > * getList` (void)

Containing list.

Protected Member Functions

- [iterator](#) ([ListItem](#) *li)

Protected Attributes

- [ListItem](#) * act

Friends

- class [RootLinkedList< R, T >](#)

5.4.1 Detailed Description

`template<typename R, typename T> class RootLinkedList< R, T >::iterator`

Iterator class.

5.4.2 Constructor & Destructor Documentation

5.4.2.1 `template<typename R, typename T> RootLinkedList< R, T >::iterator::iterator (void) [inline]`

Reimplemented from [LinkedList< T >::iterator](#).

Reimplemented in [MainLinkedList< R, S, T >::iterator](#).

5.4.2.2 `template<typename R, typename T> RootLinkedList< R, T >::iterator::iterator (ListItem * li) [inline, protected]`

5.4.3 Member Function Documentation

5.4.3.1 `template<typename R , typename T > LinkedList< T > * RootLinkedList< R, T >::iterator::getList (void) [inline]`

Containing list.

Returns

pointer to the containing list

Reimplemented in [MainLinkedList< R, S, T >::iterator](#).

5.4.3.2 `template<typename R , typename T > bool RootLinkedList< R, T >::iterator::operator!=(const iterator & i) const [inline]`

5.4.3.3 `template<typename R , typename T > R & RootLinkedList< R, T >::iterator::operator*(void) [inline]`

Reimplemented from [LinkedList< T >::iterator](#).

Reimplemented in [MainLinkedList< R, S, T >::iterator](#).

5.4.3.4 `template<typename R , typename T > RootLinkedList< R, T >::iterator RootLinkedList< R, T >::iterator::operator++(int) [inline]`

Reimplemented from [LinkedList< T >::iterator](#).

Reimplemented in [MainLinkedList< R, S, T >::iterator](#).

5.4.3.5 `template<typename R , typename T > RootLinkedList< R, T >::iterator RootLinkedList< R, T >::iterator::operator++(void) [inline]`

Reimplemented from [LinkedList< T >::iterator](#).

Reimplemented in [MainLinkedList< R, S, T >::iterator](#).

5.4.3.6 `template<typename R , typename T > R * RootLinkedList< R, T >::iterator::operator->(void) [inline]`

Reimplemented from [LinkedList< T >::iterator](#).

Reimplemented in [MainLinkedList< R, S, T >::iterator](#).

5.4.3.7 `template<typename R , typename T > bool RootLinkedList< R, T >::iterator::operator==(const iterator & i) const [inline]`

5.4.4 Friends And Related Function Documentation

5.4.4.1 `template<typename R, typename T> friend class RootLinkedList< R, T > [friend]`

5.4.5 Member Data Documentation

5.4.5.1 `template<typename R, typename T> ListItem* RootLinkedList< R, T >::iterator::act [protected]`

Reimplemented from [LinkedList< T >::iterator](#).

Reimplemented in [MainLinkedList< R, S, T >::iterator](#).

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

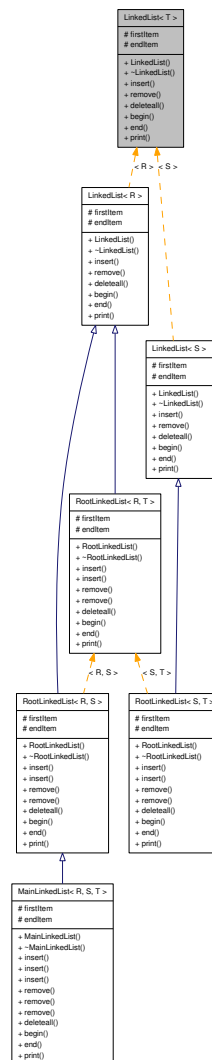
- [list.h](#)
- [list.hpp](#)

5.5 LinkedList< T > Class Template Reference

Linked list class. Simple sorted linked list class.

```
#include <list.h>
```

Inheritance diagram for LinkedList< T >:



Classes

- class [iterator](#)

Iterator class.

- struct [ListItem](#)

Linked list's item.

Public Member Functions

- [LinkedList](#) (void)

Constructor sets the pivot elements.

- [~LinkedList](#) (void)
- void [insert](#) (const T &data)

Inserts one item in the linked list.

- void [remove](#) (const T &data)

Removes one element from linked list.

- void [deleteall](#) (void)

Deletes all element in linked list, except pivot item.

- [iterator begin](#) (void)
- [iterator end](#) (void)
- void [print](#) (void)

Prints the list to standard output.

Protected Attributes

- [ListItem](#) * [firstItem](#)
- [ListItem](#) * [endItem](#)

Friends

- class [LinkedList< T >::iterator](#)

5.5.1 Detailed Description

template<typename T> class [LinkedList< T >](#)

Linked list class. Simple sorted linked list class.

5.5.2 Constructor & Destructor Documentation

5.5.2.1 [template<typename T > \[LinkedList< T >::LinkedList\]\(#\) \(void\) \[\\[inline\\]\]\(#\)](#)

Constructor sets the pivot elements.

5.5.2.2 `template<typename T> LinkedList< T >::~~LinkedList (void) [inline]`

5.5.3 Member Function Documentation

5.5.3.1 `template<typename T> LinkedList< T >::iterator LinkedList< T >::begin (void) [inline]`

Reimplemented in [RootLinkedList< R, T >](#), [MainLinkedList< R, S, T >](#), [RootLinkedList< R, S >](#), and [RootLinkedList< S, T >](#).

5.5.3.2 `template<typename T> void LinkedList< T >::deleteall (void) [inline]`

Deletes all element in linked list, except pivot item.

Reimplemented in [RootLinkedList< R, T >](#), [MainLinkedList< R, S, T >](#), [RootLinkedList< R, S >](#), and [RootLinkedList< S, T >](#).

5.5.3.3 `template<typename T> LinkedList< T >::iterator LinkedList< T >::end (void) [inline]`

Reimplemented in [RootLinkedList< R, T >](#), [MainLinkedList< R, S, T >](#), [RootLinkedList< R, S >](#), and [RootLinkedList< S, T >](#).

5.5.3.4 `template<typename T> void LinkedList< T >::insert (const T & data) [inline]`

Inserts one item in the linked list.

Parameters

data Value of the new item.

Reimplemented in [RootLinkedList< R, T >](#), [MainLinkedList< R, S, T >](#), [RootLinkedList< R, S >](#), and [RootLinkedList< S, T >](#).

5.5.3.5 `template<typename T> void LinkedList< T >::print (void) [inline]`

Prints the list to standard output.

Reimplemented in [RootLinkedList< R, T >](#), [MainLinkedList< R, S, T >](#), [RootLinkedList< R, S >](#), and [RootLinkedList< S, T >](#).

5.5.3.6 `template<typename T> void LinkedList< T >::remove (const T & data) [inline]`

Removes one element from linked list.

Parameters

data Value of the deleted item

Reimplemented in [RootLinkedList< R, T >](#), [MainLinkedList< R, S, T >](#), [RootLinkedList< R, S >](#), and [RootLinkedList< S, T >](#).

5.5.4 Friends And Related Function Documentation

5.5.4.1 `template<typename T> friend class LinkedList< T >::iterator` [**friend**]

5.5.5 Member Data Documentation

5.5.5.1 `template<typename T> ListItem* LinkedList< T >::endItem` [**protected**]

Reimplemented in [RootLinkedList< R, T >](#), [MainLinkedList< R, S, T >](#), [RootLinkedList< R, S >](#), and [RootLinkedList< S, T >](#).

5.5.5.2 `template<typename T> ListItem* LinkedList< T >::firstItem` [**protected**]

Reimplemented in [RootLinkedList< R, T >](#), [MainLinkedList< R, S, T >](#), [RootLinkedList< R, S >](#), and [RootLinkedList< S, T >](#).

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

- [list.h](#)
- [list.hpp](#)

5.6 ListException Class Reference

```
#include <list.h>
```

Public Member Functions

- [ListException](#) (const char *err_)
- [~ListException](#) ()
- virtual const char * [what](#) () const

5.6.1 Constructor & Destructor Documentation

5.6.1.1 `ListException::ListException (const char * err_)` [`inline`]

5.6.1.2 `ListException::~~ListException ()` [`inline`]

5.6.2 Member Function Documentation

5.6.2.1 `virtual const char* ListException::what () const` [`inline`, `virtual`]

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

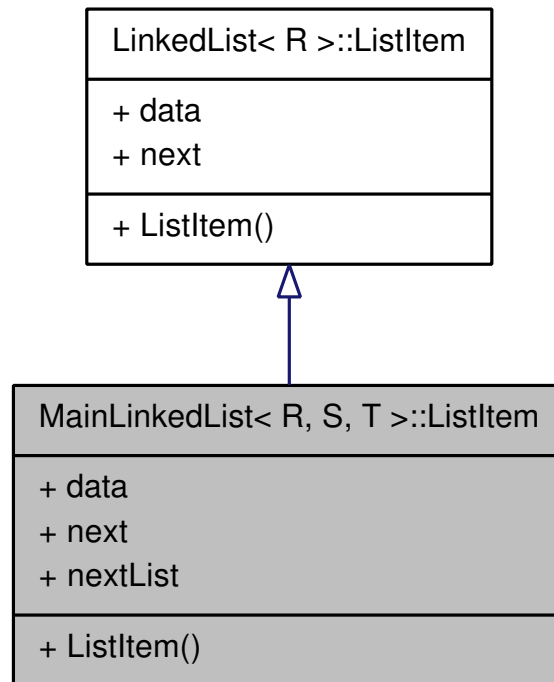
- [list.h](#)

5.7 MainLinkedList< R, S, T >::ListItem Struct Reference

Three-level Linked list's item.

```
#include <list.h>
```

Inheritance diagram for MainLinkedList< R, S, T >::ListItem:



Public Member Functions

- [ListItem](#) ([ListItem](#) *item=0)

Public Attributes

- [R](#) [data](#)
- [ListItem](#) * [next](#)
- [RootLinkedList](#)< [S](#), [T](#) > * [nextList](#)

5.7.1 Detailed Description

```
template<class R, class S, class T> struct MainLinkedList< R, S, T >::ListItem
```

Three-level Linked list's item.

5.7.2 Constructor & Destructor Documentation

5.7.2.1 `template<class R , class S , class T > MainLinkedList< R, S, T >::ListItem::ListItem
(ListItem * item = 0) [inline]`

5.7.3 Member Data Documentation

5.7.3.1 `template<class R , class S , class T > R MainLinkedList< R, S, T >::ListItem::data`

Reimplemented from [LinkedList< T >::ListItem](#).

5.7.3.2 `template<class R , class S , class T > ListItem* MainLinkedList< R, S, T
>::ListItem::next`

Reimplemented from [LinkedList< T >::ListItem](#).

5.7.3.3 `template<class R , class S , class T > RootLinkedList<S, T>* MainLinkedList< R, S, T
>::ListItem::nextList`

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

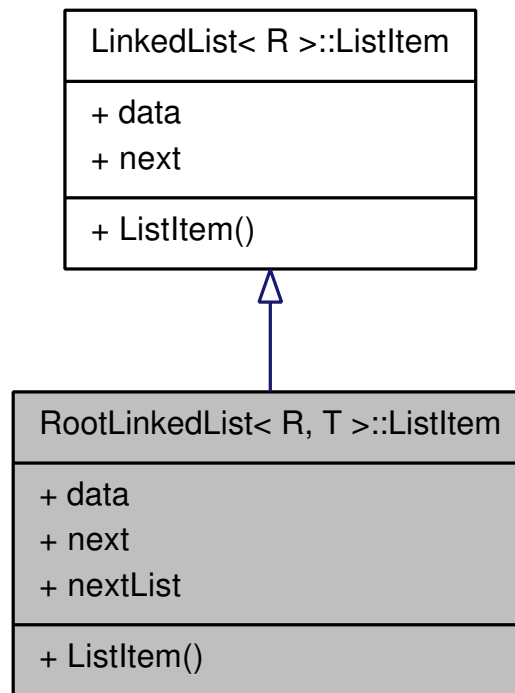
- [list.h](#)

5.8 RootLinkedList< R, T >::ListItem Struct Reference

Root Linked list's item.

```
#include <list.h>
```

Inheritance diagram for RootLinkedList< R, T >::ListItem:



Public Member Functions

- `ListItem (ListItem *item=0)`

Public Attributes

- `R data`
- `ListItem * next`
- `LinkedList< T > * nextList`

5.8.1 Detailed Description

```
template<typename R, typename T> struct RootLinkedList< R, T >::ListItem
```

Root Linked list's item.

5.8.2 Constructor & Destructor Documentation

5.8.2.1 `template<typename R, typename T> RootLinkedList< R, T >::ListItem::ListItem
(ListItem * item = 0) [inline]`

5.8.3 Member Data Documentation

5.8.3.1 `template<typename R, typename T> R RootLinkedList< R, T >::ListItem::data`

Reimplemented from [LinkedList< T >::ListItem](#).

5.8.3.2 `template<typename R, typename T> ListItem* RootLinkedList< R, T >::ListItem::next`

Reimplemented from [LinkedList< T >::ListItem](#).

5.8.3.3 `template<typename R, typename T> LinkedList<T>* RootLinkedList< R, T
>::ListItem::nextList`

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

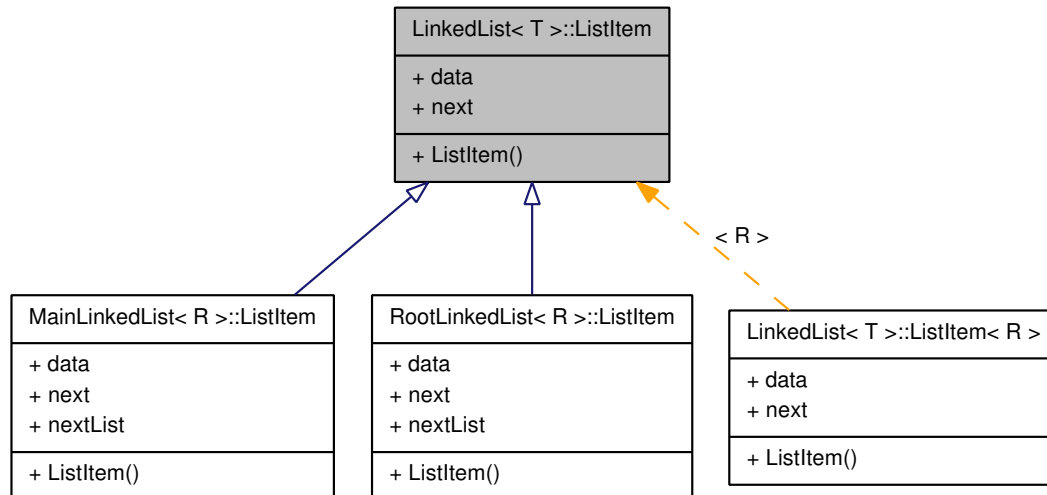
- [list.h](#)

5.9 LinkedList< T >::ListItem Struct Reference

Linked list's item.

```
#include <list.h>
```

Inheritance diagram for LinkedList< T >::ListItem:



Public Member Functions

- [ListItem](#) ([ListItem](#) *item=0)

Public Attributes

- [T](#) `data`
- [ListItem](#) * `next`

5.9.1 Detailed Description

```
template<typename T> struct LinkedList< T >::ListItem
```

Linked list's item.

5.9.2 Constructor & Destructor Documentation

5.9.2.1 `template<typename T> LinkedList< T >::ListItem::ListItem (ListItem * item = 0)`
`[inline]`

5.9.3 Member Data Documentation

5.9.3.1 `template<typename T> T LinkedList< T >::ListItem::data`

Reimplemented in [RootLinkedList< R, T >::ListItem](#), and [MainLinkedList< R, S, T >::ListItem](#).

5.9.3.2 `template<typename T> ListItem* LinkedList< T >::ListItem::next`

Reimplemented in [RootLinkedList< R, T >::ListItem](#), and [MainLinkedList< R, S, T >::ListItem](#).

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

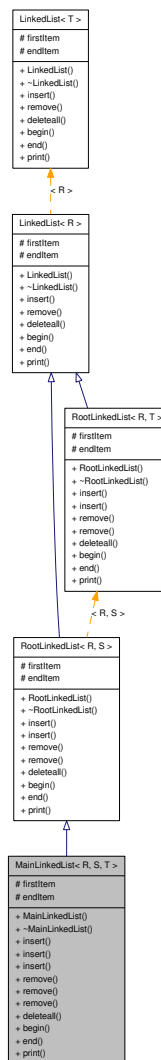
- [list.h](#)

5.10 MainLinkedList< R, S, T > Class Template Reference

Three-level linked list.

```
#include <list.h>
```

Inheritance diagram for MainLinkedList< R, S, T >:



Classes

- class [iterator](#)

Iterator class.

- struct [ListItem](#)

Three-level Linked list's item.

Public Member Functions

- [MainLinkedList](#) (void)
- [~MainLinkedList](#) (void)
- void [insert](#) (const R &r_data)
- void [insert](#) (const R &r_data, const S &S_data)
- void [insert](#) (const R &r_data, const S &S_data, const T &T_data)
- void [remove](#) (const R &data)
- void [remove](#) (const R &r_data, const S &t_data)
- void [remove](#) (const R &r_data, const S &S_data, const T &T_data)
- void [deleteall](#) (void)
- [iterator begin](#) (void)
- [iterator end](#) (void)
- void [print](#) (void)

Protected Attributes

- [ListItem](#) * [firstItem](#)
- [ListItem](#) * [endItem](#)

Friends

- class [MainLinkedList< R, S, T >::iterator](#)

5.10.1 Detailed Description

template<class R, class S, class T> class MainLinkedList< R, S, T >

Three-level linked list.

5.10.2 Constructor & Destructor Documentation

5.10.2.1 `template<typename R , typename S , typename T > MainLinkedList< R, S, T >::MainLinkedList (void) [inline]`

5.10.2.2 `template<typename R , typename S , typename T > MainLinkedList< R, S, T >::~~MainLinkedList (void) [inline]`

5.10.3 Member Function Documentation

5.10.3.1 `template<typename R , typename S , typename T > MainLinkedList< R, S, T >::iterator MainLinkedList< R, S, T >::begin (void) [inline]`

Reimplemented from [RootLinkedList< R, S >](#).

5.10.3.2 `template<typename R , typename S , typename T > void MainLinkedList< R, S, T >::~deleteall (void) [inline]`

Reimplemented from [RootLinkedList< R, S >](#).

5.10.3.3 `template<typename R , typename S , typename T > MainLinkedList< R, S, T >::iterator
MainLinkedList< R, S, T >::end (void) [inline]`

Reimplemented from [RootLinkedList< R, S >](#).

5.10.3.4 `template<typename R , typename S , typename T > void MainLinkedList< R, S, T
>::insert (const R & r_data, const S & S_data, const T & T_data) [inline]`

5.10.3.5 `template<typename R , typename S , typename T > void MainLinkedList< R, S, T
>::insert (const R & r_data, const S & S_data) [inline]`

Reimplemented from [RootLinkedList< R, S >](#).

5.10.3.6 `template<typename R , typename S , typename T > void MainLinkedList< R, S, T
>::insert (const R & r_data) [inline]`

Reimplemented from [RootLinkedList< R, S >](#).

5.10.3.7 `template<typename R , typename S , typename T > void MainLinkedList< R, S, T
>::print (void) [inline]`

Reimplemented from [RootLinkedList< R, S >](#).

5.10.3.8 `template<typename R , typename S , typename T > void MainLinkedList< R, S, T
>::remove (const R & r_data, const S & S_data, const T & T_data) [inline]`

5.10.3.9 `template<typename R , typename S , typename T > void MainLinkedList< R, S, T
>::remove (const R & r_data, const S & t_data) [inline]`

Reimplemented from [RootLinkedList< R, S >](#).

5.10.3.10 `template<typename R , typename S , typename T > void MainLinkedList< R, S, T
>::remove (const R & data) [inline]`

Reimplemented from [RootLinkedList< R, S >](#).

5.10.4 Friends And Related Function Documentation

5.10.4.1 `template<class R , class S , class T > friend class MainLinkedList< R, S, T >::iterator
[friend]`

5.10.5 Member Data Documentation

5.10.5.1 `template<class R , class S , class T > ListItem* MainLinkedList< R, S, T >::endItem
[protected]`

Reimplemented from [RootLinkedList< R, S >](#).

5.10.5.2 `template<class R , class S , class T > ListItem* MainLinkedList< R, S, T >::firstItem` `[protected]`

Reimplemented from [RootLinkedList< R, S >](#).

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

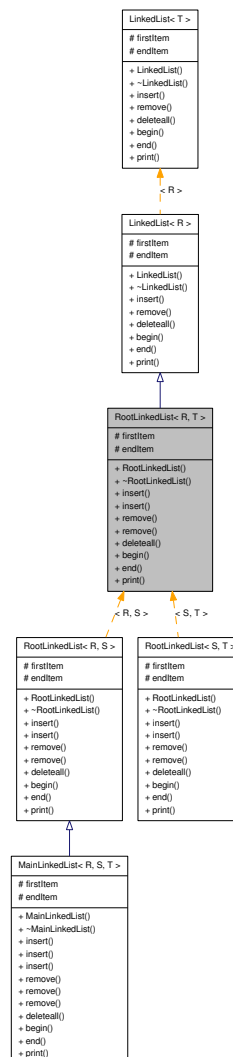
- [list.h](#)
- [list.hpp](#)

5.11 RootLinkedList< R, T > Class Template Reference

2 level Root Linked List. Each list item has one child list.

```
#include <list.h>
```

Inheritance diagram for RootLinkedList< R, T >:



Classes

- class [iterator](#)

Iterator class.

- struct [ListItem](#)

Root Linked list's item.

Public Member Functions

- [RootLinkedList](#) (void)
Constructor sets pivot elements.
- [~RootLinkedList](#) (void)
- void [insert](#) (const R &r_data)
Inserts an element into the list.
- void [insert](#) (const R &r_data, const T &t_data)
Inserts one element in the containing list.
- void [remove](#) (const R &data)
- void [remove](#) (const R &r_data, const T &t_data)
- void [deleteall](#) (void)
- [iterator begin](#) (void)
- [iterator end](#) (void)
- void [print](#) (void)

Protected Attributes

- [ListItem](#) * [firstItem](#)
- [ListItem](#) * [endItem](#)

Friends

- class [RootLinkedList< R, T >::iterator](#)

5.11.1 Detailed Description

template<typename R, typename T> class RootLinkedList< R, T >

2 level Root Linked List. Each list item has one child list.

5.11.2 Constructor & Destructor Documentation

5.11.2.1 template<typename R , typename T > RootLinkedList< R, T >::RootLinkedList (void) [inline]

Constructor sets pivot elements.

5.11.2.2 template<typename R , typename T > RootLinkedList< R, T >::~~RootLinkedList (void) [inline]

5.11.3 Member Function Documentation

5.11.3.1 template<typename R , typename T > RootLinkedList< R, T >::iterator RootLinkedList< R, T >::begin (void) [inline]

Reimplemented from [LinkedList< R >](#).

Reimplemented in [MainLinkedList< R, S, T >](#).

5.11.3.2 `template<typename R, typename T> void RootLinkedList< R, T >::deleteall (void) [inline]`

Reimplemented from [LinkedList< R >](#).

Reimplemented in [MainLinkedList< R, S, T >](#).

5.11.3.3 `template<typename R, typename T> RootLinkedList< R, T >::iterator RootLinkedList< R, T >::end (void) [inline]`

Reimplemented from [LinkedList< R >](#).

Reimplemented in [MainLinkedList< R, S, T >](#).

5.11.3.4 `template<typename R, typename T> void RootLinkedList< R, T >::insert (const R & r_data, const T & t_data) [inline]`

Inserts one element in the containing list.

Reimplemented in [MainLinkedList< R, S, T >](#).

5.11.3.5 `template<typename R, typename T> void RootLinkedList< R, T >::insert (const R & data) [inline]`

Inserts an element into the list.

Parameters

data Value of the new item

Reimplemented from [LinkedList< R >](#).

Reimplemented in [MainLinkedList< R, S, T >](#).

5.11.3.6 `template<typename R, typename T> void RootLinkedList< R, T >::print (void) [inline]`

Reimplemented from [LinkedList< R >](#).

Reimplemented in [MainLinkedList< R, S, T >](#).

5.11.3.7 `template<typename R, typename T> void RootLinkedList< R, T >::remove (const R & r_data, const T & t_data) [inline]`

Reimplemented in [MainLinkedList< R, S, T >](#).

5.11.3.8 `template<typename R, typename T> void RootLinkedList< R, T >::remove (const R & data) [inline]`

Reimplemented from [LinkedList< R >](#).

Reimplemented in [MainLinkedList< R, S, T >](#).

5.11.4 Friends And Related Function Documentation

5.11.4.1 `template<typename R, typename T> friend class RootLinkedList< R, T >::iterator`
[friend]

5.11.5 Member Data Documentation

5.11.5.1 `template<typename R, typename T> ListItem* RootLinkedList< R, T >::endItem`
[protected]

Reimplemented from [LinkedList< R >](#).

Reimplemented in [MainLinkedList< R, S, T >](#).

5.11.5.2 `template<typename R, typename T> ListItem* RootLinkedList< R, T >::firstItem`
[protected]

Reimplemented from [LinkedList< R >](#).

Reimplemented in [MainLinkedList< R, S, T >](#).

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

- [list.h](#)
- [list.hpp](#)

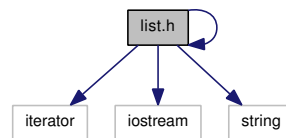
Chapter 6

File Documentation

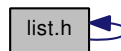
6.1 list.h File Reference

```
#include <iterator>
#include <iostream>
#include <string>
#include "list.hpp"
```

Include dependency graph for list.h:



This graph shows which files directly or indirectly include this file:



Classes

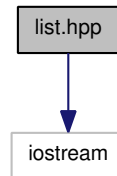
- class `Cmp< T >`
Compare template class. Compare class containing 2 predicate functions (equals, less than) In some cases specialization is needed.
- class `LinkedList< T >`
Linked list class. Simple sorted linked list class.
- struct `LinkedList< T >::ListItem`
Linked list's item.

- class `LinkedList< T >::iterator`
Iterator class.
- class `RootLinkedList< R, T >`
2 level Root Linked List. Each list item has one child list.
- struct `RootLinkedList< R, T >::ListItem`
Root Linked list's item.
- class `RootLinkedList< R, T >::iterator`
Iterator class.
- class `MainLinkedList< R, S, T >`
Three-level linked list.
- struct `MainLinkedList< R, S, T >::ListItem`
Three-level Linked list's item.
- class `MainLinkedList< R, S, T >::iterator`
Iterator class.
- class `ListException`

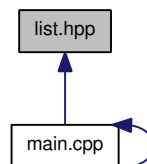
6.2 list.hpp File Reference

```
#include <iostream>
```

Include dependency graph for list.hpp:



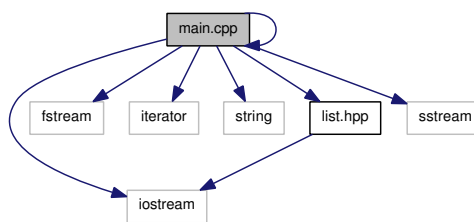
This graph shows which files directly or indirectly include this file:



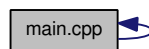
6.3 main.cpp File Reference

```
#include <iostream>
#include <fstream>
#include "list.h"
#include <iterator>
#include <string>
#include "list.hpp"
#include <sstream>
```

Include dependency graph for main.cpp:



This graph shows which files directly or indirectly include this file:



Functions

- int [main](#) (void)

6.3.1 Function Documentation

6.3.1.1 int main (void)

Index

- ~LinkedList
 - LinkedList, [20](#)
- ~ListException
 - ListException, [23](#)
- ~MainLinkedList
 - MainLinkedList, [31](#)
- ~RootLinkedList
 - RootLinkedList, [35](#)
- act
 - LinkedList::iterator, [12](#)
 - MainLinkedList::iterator, [15](#)
 - RootLinkedList::iterator, [18](#)
- begin
 - LinkedList, [21](#)
 - MainLinkedList, [31](#)
 - RootLinkedList, [35](#)
- Cmp, [9](#)
 - eq, [9](#)
 - lt, [9](#)
- data
 - LinkedList::ListItem, [28](#)
 - MainLinkedList::ListItem, [25](#)
 - RootLinkedList::ListItem, [27](#)
- deleteall
 - LinkedList, [21](#)
 - MainLinkedList, [31](#)
 - RootLinkedList, [36](#)
- end
 - LinkedList, [21](#)
 - MainLinkedList, [31](#)
 - RootLinkedList, [36](#)
- endItem
 - LinkedList, [22](#)
 - MainLinkedList, [32](#)
 - RootLinkedList, [37](#)
- eq
 - Cmp, [9](#)
- firstItem
 - LinkedList, [22](#)
 - MainLinkedList, [32](#)
- RootLinkedList, [37](#)
- getList
 - MainLinkedList::iterator, [14](#)
 - RootLinkedList::iterator, [17](#)
- insert
 - LinkedList, [21](#)
 - MainLinkedList, [32](#)
 - RootLinkedList, [36](#)
- iterator
 - LinkedList::iterator, [11](#)
 - MainLinkedList::iterator, [14](#)
 - RootLinkedList::iterator, [17](#)
- LinkedList, [19](#)
 - ~LinkedList, [20](#)
 - begin, [21](#)
 - deleteall, [21](#)
 - end, [21](#)
 - endItem, [22](#)
 - firstItem, [22](#)
 - insert, [21](#)
 - LinkedList, [20](#)
 - LinkedList< T >::iterator, [22](#)
 - print, [21](#)
 - remove, [21](#)
- LinkedList< T >
 - LinkedList::iterator, [12](#)
- LinkedList< T >::iterator
 - LinkedList, [22](#)
- LinkedList::iterator, [10](#)
 - act, [12](#)
 - iterator, [11](#)
 - LinkedList< T >, [12](#)
 - operator*, [11](#)
 - operator++, [11](#)
 - operator->, [11](#)
 - operator==, [12](#)
- LinkedList::ListItem, [28](#)
 - data, [28](#)
 - ListItem, [28](#)
 - next, [28](#)
- list.h, [39](#)
- list.hpp, [41](#)

- ListException, 23
 - ~ListException, 23
 - ListException, 23
 - what, 23
- ListItem
 - LinkedList::ListItem, 28
 - MainLinkedList::ListItem, 25
 - RootLinkedList::ListItem, 27
- lt
 - Cmp, 9
- main
 - main.cpp, 42
- main.cpp, 42
 - main, 42
- MainLinkedList, 30
 - ~MainLinkedList, 31
 - begin, 31
 - deleteall, 31
 - end, 31
 - endItem, 32
 - firstItem, 32
 - insert, 32
 - MainLinkedList, 31
 - MainLinkedList< R, S, T >::iterator, 32
 - print, 32
 - remove, 32
- MainLinkedList< R, S, T >
 - MainLinkedList::iterator, 15
- MainLinkedList< R, S, T >::iterator
 - MainLinkedList, 32
- MainLinkedList::iterator, 13
 - act, 15
 - getList, 14
 - iterator, 14
 - MainLinkedList< R, S, T >, 15
 - operator*, 15
 - operator++, 15
 - operator->, 15
 - operator==, 15
- MainLinkedList::ListItem, 24
 - data, 25
 - ListItem, 25
 - next, 25
 - nextList, 25
- next
 - LinkedList::ListItem, 28
 - MainLinkedList::ListItem, 25
 - RootLinkedList::ListItem, 27
- nextList
 - MainLinkedList::ListItem, 25
 - RootLinkedList::ListItem, 27
- operator*
 - LinkedList::iterator, 11
 - MainLinkedList::iterator, 15
 - RootLinkedList::iterator, 18
- operator++
 - LinkedList::iterator, 11
 - MainLinkedList::iterator, 15
 - RootLinkedList::iterator, 18
- operator->
 - LinkedList::iterator, 11
 - MainLinkedList::iterator, 15
 - RootLinkedList::iterator, 18
- operator==
 - LinkedList::iterator, 12
 - MainLinkedList::iterator, 15
 - RootLinkedList::iterator, 18
- print
 - LinkedList, 21
 - MainLinkedList, 32
 - RootLinkedList, 36
- remove
 - LinkedList, 21
 - MainLinkedList, 32
 - RootLinkedList, 36
- RootLinkedList, 34
 - ~RootLinkedList, 35
 - begin, 35
 - deleteall, 36
 - end, 36
 - endItem, 37
 - firstItem, 37
 - insert, 36
 - print, 36
 - remove, 36
 - RootLinkedList, 35
 - RootLinkedList< R, T >::iterator, 37
- RootLinkedList< R, T >
 - RootLinkedList::iterator, 18
- RootLinkedList< R, T >::iterator
 - RootLinkedList, 37
- RootLinkedList::iterator, 16
 - act, 18
 - getList, 17
 - iterator, 17
 - operator*, 18
 - operator++, 18
 - operator->, 18
 - operator==, 18
 - RootLinkedList< R, T >, 18
- RootLinkedList::ListItem, 26
 - data, 27
 - ListItem, 27

next, [27](#)
nextList, [27](#)

what
 ListException, [23](#)