Práctica 5

v.1.0

Generado por Doxygen 1.8.10

Lunes, 19 de Abril de 2021 14:58:27

Índice general

1 Índice de clases **1**
1.1 Lista de clases . . . 1

2 Indice de archivos **3**
2.1 Lista de archivos . . . 3

3 Documentación de las clases **5**
3.1 Referencia de la plantilla de la Clase dll_node_t< T > . . . 5
3.1.1 Descripción detallada . . . 5
3.1.2 Documentación del constructor y destructor . . . 5
3.1.2.1 dll_node_t() . . . 5
3.1.2.2 dll_node_t(const T &data) . . . 5
3.1.2.3 ∼dll_node_t(void) . . . 5
3.1.3 Documentación de las funciones miembro . . . 6
3.1.3.1 get_data(void) const . . . 6
3.1.3.2 get_next(void) const . . . 6
3.1.3.3 get_prev(void) const . . . 6
3.1.3.4 set_data(const T &data) . . . 7
3.1.3.5 set_next(dll_node_t< T > ∗next) . . . 7
3.1.3.6 set_prev(dll_node_t< T > ∗prev) . . . 7
3.1.3.7 write(std::ostream &=std::cout) const . . . 8
3.2 Referencia de la plantilla de la Clase dll_t< T > . . . 8
3.2.1 Descripción detallada . . . 9
3.2.2 Documentación del constructor y destructor . . . 9
3.2.2.1 dll_t(void) . . . 9
3.2.2.2 ∼dll_t(void) . . . 9
3.2.3 Documentación de las funciones miembro . . . 9
3.2.3.1 empty(void) const . . . 9
3.2.3.2 erase(dll_node_t< T > ∗) . . . 9
3.2.3.3 get_head(void) const . . . 10
3.2.3.4 get_size(void) const . . . 10
3.2.3.5 get_tail(void) const . . . 10
3.2.3.6 pop_back(void) . . . 10
3.2.3.7 pop_front(void) . . . 11
3.2.3.8 push_back(dll_node_t< T > ∗) . . . 11
3.2.3.9 push_front(dll_node_t< T > ∗) . . . 11
3.2.3.10 write(std::ostream &=std::cout) const . . . 12
3.3 Referencia de la plantilla de la Clase queue_l_t< T > . . . 12
3.3.1 Descripción detallada . . . 13
3.3.2 Documentación del constructor y destructor . . . 13
3.3.2.1 queue_l_t(void) . . . 13
3.3.2.2 ∼queue_l_t(void) . . . 13
3.3.3 Documentación de las funciones miembro . . . 13
3.3.3.1 back(void) const . . . 13
3.3.3.2 empty(void) const . . . 13
3.3.3.3 front(void) const . . . 13

3.3.3.4 pop(void) . . . 14
3.3.3.5 push(const T &dato) . . . 14
3.3.3.6 size(void) const . . . 14
3.3.3.7 write(std::ostream &os=std::cout) const . . . 14
3.4 Referencia de la plantilla de la Clase rpn_t< T > . . . 15
3.4.1 Descripción detallada . . . 15
3.4.2 Documentación del constructor y destructor . . . 15
3.4.2.1 rpn_t(void) . . . 15
3.4.2.2 ∼rpn_t() . . . 15
3.4.3 Documentación de las funciones miembro . . . 15
3.4.3.1 evaluate(queue_l_t< char > &) . . . 15
3.5 Referencia de la plantilla de la Clase stack_l_t< T > . . . 16
3.5.1 Descripción detallada . . . 16
3.5.2 Documentación del constructor y destructor . . . 17
3.5.2.1 stack_l_t(void) . . . 17
3.5.2.2 ∼stack_l_t(void) . . . 17
3.5.3 Documentación de las funciones miembro . . . 17
3.5.3.1 empty(void) const . . . 17
3.5.3.2 pop(void) . . . 17
3.5.3.3 push(const T &) . . . 17
3.5.3.4 top(void) const . . . 17
3.5.3.5 write(std::ostream &os=std::cout) const . . . 17

4 Documentación de archivos **19**
4.1 Referencia del Archivo dll_node_t.h . . . 19
4.2 Referencia del Archivo dll_t.h . . . 20
4.3 Referencia del Archivo main_rpn_t.cc . . . 21
4.3.1 Documentación de las funciones . . . 22
4.3.1.1 main(void) . . . 22
4.4 Referencia del Archivo queue_l_t.h . . . 23
4.4.1 Documentación de las funciones . . . 24
4.4.1.1 operator<<(std::ostream &os, const queue_l_t< T > &q) . . . 24
4.5 Referencia del Archivo rpn_t.h . . . 24
4.6 Referencia del Archivo stack_l_t.h . . . 25

Índice **27**

Capítulo 1

Índice de clases

1.1. Lista de clases

Lista de las clases, estructuras, uniones e interfaces con una breve descripción:

dll_node_t< T > 5
dll_t< T > 8
queue_l_t< T > 12
rpn_t< T > 15
stack_l_t< T > 16

Capítulo 2

Indice de archivos

2.1. Lista de archivos

Lista de todos los archivos con descripciones breves:

dll_node_t.h 19
dll_t.h 20
main_rpn_t.cc 21
queue_l_t.h 23
rpn_t.h 24
stack_l_t.h 25

Capítulo 3

Documentación de las clases

3.1. Referencia de la plantilla de la Clase dll_node_t< T >

`#include <dll_node_t.h>`

Métodos públicos

- dll_node_t ()
- dll_node_t (const T &data)
- ~dll_node_t (void)
- dll_node_t< T > * get_next (void) const
- void set_next (dll_node_t< T > *next)
- dll_node_t< T > * get_prev (void) const
- void set_prev (dll_node_t< T > *prev)
- const T & get_data (void) const
- void set_data (const T &data)
- std::ostream & write (std::ostream &=std::cout) const

3.1.1. Descripción detallada

template<class T>class dll_node_t< T >

Definición en la línea 16 del archivo dll_node_t.h.

3.1.2. Documentación del constructor y destructor

3.1.2.1. template<class T> dll_node_t< T >::dll_node_t ( ) [inline]

Definición en la línea 19 del archivo dll_node_t.h.

3.1.2.2. template<class T> dll_node_t< T >::dll_node_t (const T & *data*) [inline]

Definición en la línea 20 del archivo dll_node_t.h.

3.1.2.3. template<class T> dll_node_t< T >::~dll_node_t (void) [inline]

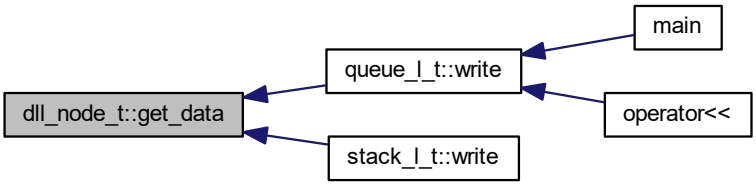
Definición en la línea 23 del archivo dll_node_t.h.

3.1.3. Documentación de las funciones miembro

3.1.3.1. template<class T> const T& dll_node_t< T >::get_data (void) const [inline]

Definición en la línea 32 del archivo dll_node_t.h.

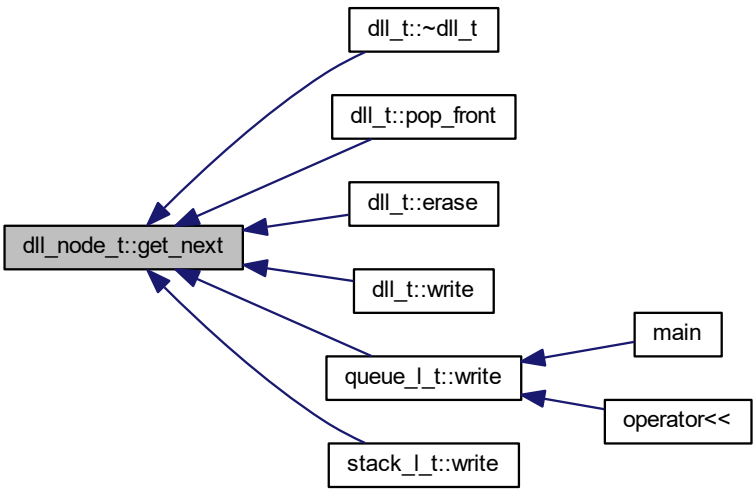
Gráfico de llamadas a esta función:



3.1.3.2. template<class T> dll_node_t<T>* dll_node_t< T >::get_next (void) const [inline]

Definición en la línea 26 del archivo dll_node_t.h.

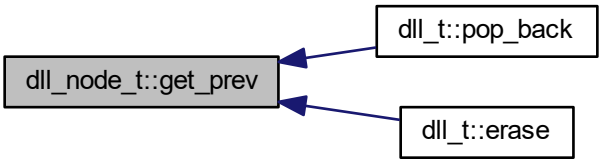
Gráfico de llamadas a esta función:



3.1.3.3. template<class T> dll_node_t<T>* dll_node_t< T >::get_prev (void) const [inline]

Definición en la línea 29 del archivo dll_node_t.h.

Gráfico de llamadas a esta función:



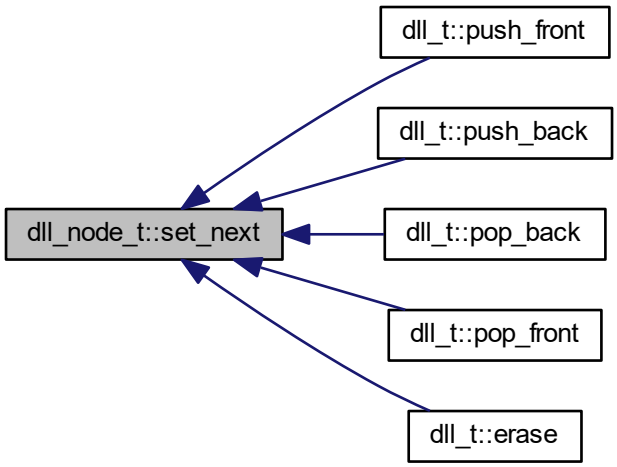
3.1.3.4. template<class T> void dll_node_t< T >::set_data (const T & *data*) [inline]

Definición en la línea 33 del archivo dll_node_t.h.

3.1.3.5. template<class T> void dll_node_t< T >::set_next (dll_node_t< T > * *next*) [inline]

Definición en la línea 27 del archivo dll_node_t.h.

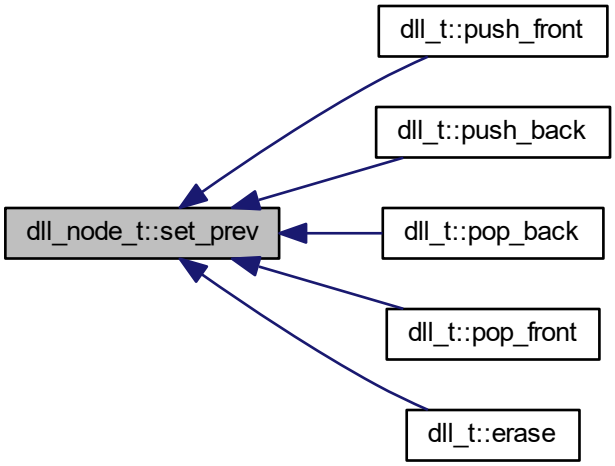
Gráfico de llamadas a esta función:



3.1.3.6. template<class T> void dll_node_t< T >::set_prev (dll_node_t< T > * *prev*) [inline]

Definición en la línea 30 del archivo dll_node_t.h.

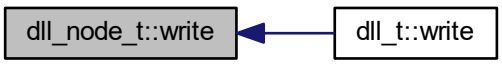
Gráfico de llamadas a esta función:



3.1.3.7. template<class T > std::ostream & dll_node_t< T >::write (std::ostream & *os* = `std::cout`) const

Definición en la línea 46 del archivo dll_node_t.h.

Gráfico de llamadas a esta función:



La documentación para esta clase fue generada a partir del siguiente fichero:

- dll_node_t.h

3.2. Referencia de la plantilla de la Clase dll_t< T >

`#include <dll_t.h>`

Métodos públicos

- dll_t (void)
- ~dll_t (void)
- dll_node_t< T > * get_tail (void) const

- dll_node_t< T > * get_head (void) const
- int get_size (void) const
- bool empty (void) const
- void push_back (dll_node_t< T > *)
- void push_front (dll_node_t< T > *)
- dll_node_t< T > * pop_back (void)
- dll_node_t< T > * pop_front (void)
- dll_node_t< T > * erase (dll_node_t< T > *)
- std::ostream & write (std::ostream &=std::cout) const

3.2.1. Descripción detallada

template<class T>class dll_t< T >

Definición en la línea 19 del archivo dll_t.h.

3.2.2. Documentación del constructor y destructor

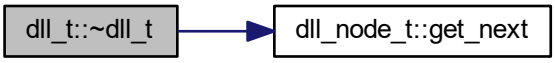
3.2.2.1. template<class T> dll_t< T >::dll_t (void) [inline]

Definición en la línea 22 del archivo dll_t.h.

3.2.2.2. template<class T > dll_t< T >::~dll_t (void)

Definición en la línea 54 del archivo dll_t.h.

Gráfico de llamadas para esta función:



3.2.3. Documentación de las funciones miembro

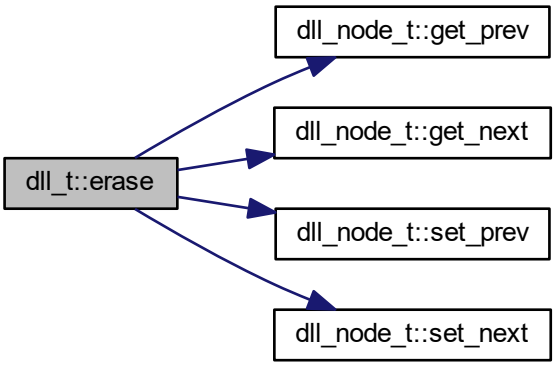
3.2.3.1. template<class T > bool dll_t< T >::empty (void) const

Definición en la línea 65 del archivo dll_t.h.

3.2.3.2. template<class T > dll_node_t< T > * dll_t< T >::erase (dll_node_t< T > * *nodo*)

Definición en la línea 133 del archivo dll_t.h.

Gráfico de llamadas para esta función:



3.2.3.3. template<class T> dll_node_t<T>* dll_t< T >::get_head (void) const [inline]

Definición en la línea 29 del archivo dll_t.h.

3.2.3.4. template<class T> int dll_t< T >::get_size (void) const [inline]

Definición en la línea 30 del archivo dll_t.h.

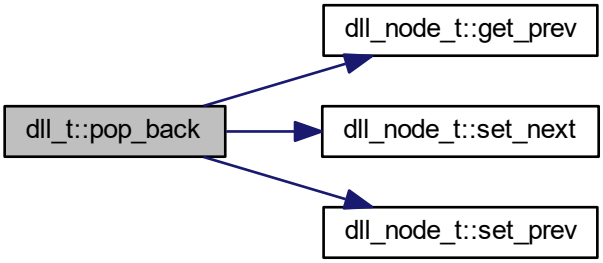
3.2.3.5. template<class T> dll_node_t<T>* dll_t< T >::get_tail (void) const [inline]

Definición en la línea 28 del archivo dll_t.h.

3.2.3.6. template<class T > dll_node_t< T > * dll_t< T >::pop_back (void)

Definición en la línea 103 del archivo dll_t.h.

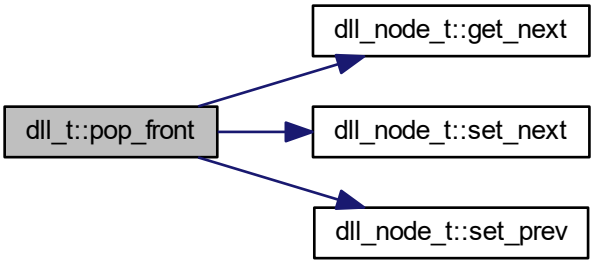
Gráfico de llamadas para esta función:



3.2.3.7. template<class T > dll_node_t< T > * dll_t< T >::pop_front (void)

Definición en la línea 118 del archivo dll_t.h.

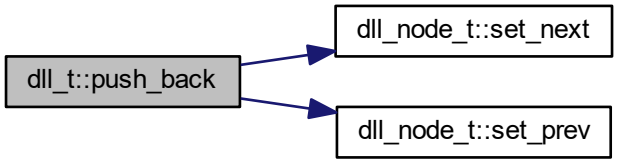
Gráfico de llamadas para esta función:



3.2.3.8. template<class T > void dll_t< T >::push_back (dll_node_t< T > * *nodo*)

Definición en la línea 89 del archivo dll_t.h.

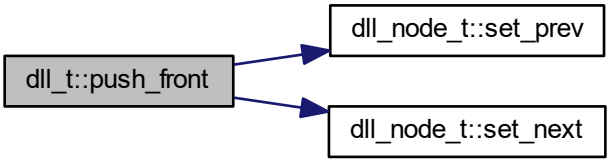
Gráfico de llamadas para esta función:



3.2.3.9. template<class T > void dll_t< T >::push_front (dll_node_t< T > * *nodo*)

Definición en la línea 75 del archivo dll_t.h.

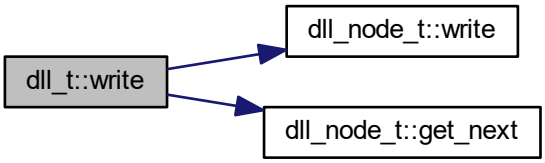
Gráfico de llamadas para esta función:



3.2.3.10. template<class T > std::ostream & dll_t< T >::write (std::ostream & *os* = `std::cout`) const

Definición en la línea 151 del archivo dll_t.h.

Gráfico de llamadas para esta función:



La documentación para esta clase fue generada a partir del siguiente fichero:

- dll_t.h

3.3. Referencia de la plantilla de la Clase queue_l_t< T >

`#include <queue_l_t.h>`

Métodos públicos

- queue_l_t (void)
- ~queue_l_t (void)
- bool empty (void) const
- int size (void) const
- void push (const T &dato)
- void pop (void)
- const T & front (void) const
- const T & back (void) const
- std::ostream & write (std::ostream &os=std::cout) const

3.3.1. Descripción detallada

template<class T>class queue_l_t< T >

Definición en la línea 19 del archivo queue_l_t.h.

3.3.2. Documentación del constructor y destructor

3.3.2.1. template<class T> queue_l_t< T >::queue_l_t (void) [inline]

Definición en la línea 25 del archivo queue_l_t.h.

3.3.2.2. template<class T> queue_l_t< T >::~queue_l_t (void) [inline]

Definición en la línea 28 del archivo queue_l_t.h.

3.3.3. Documentación de las funciones miembro

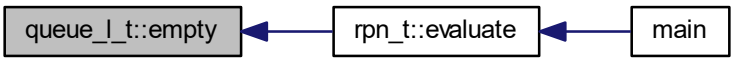
3.3.3.1. template<class T > const T & queue_l_t< T >::back (void) const

Definición en la línea 68 del archivo queue_l_t.h.

3.3.3.2. template<class T > bool queue_l_t< T >::empty (void) const

Definición en la línea 44 del archivo queue_l_t.h.

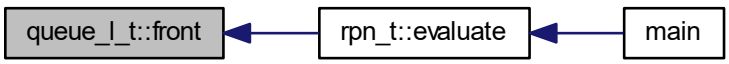
Gráfico de llamadas a esta función:



3.3.3.3. template<class T > const T & queue_l_t< T >::front (void) const

Definición en la línea 63 del archivo queue_l_t.h.

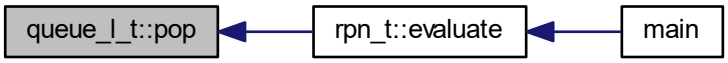
Gráfico de llamadas a esta función:



3.3.3.4. template<class T > void queue_l_t< T >::pop (void)

Definición en la línea 58 del archivo queue_l_t.h.

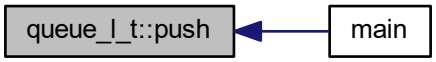
Gráfico de llamadas a esta función:



3.3.3.5. template<class T > void queue_l_t< T >::push (const T & *dato*)

Definición en la línea 52 del archivo queue_l_t.h.

Gráfico de llamadas a esta función:



3.3.3.6. template<class T > int queue_l_t< T >::size (void) const

Definición en la línea 48 del archivo queue_l_t.h.

3.3.3.7. template<class T > std::ostream & queue_l_t< T >::write (std::ostream & *os* = `std::cout`) const

Definición en la línea 74 del archivo queue_l_t.h.

Gráfico de llamadas para esta función:

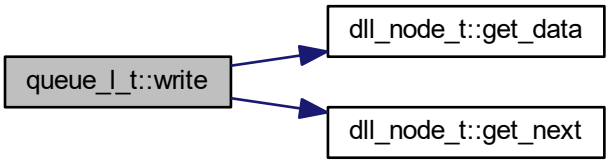
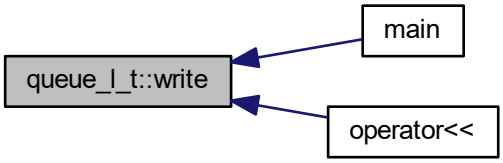


Gráfico de llamadas a esta función:



La documentación para esta clase fue generada a partir del siguiente fichero:

- queue_l_t.h

3.4. Referencia de la plantilla de la Clase rpn_t< T >

`#include <rpn_t.h>`

Métodos públicos

- rpn_t (void)
- ~rpn_t ()
- const int evaluate (queue_l_t< char > &)

3.4.1. Descripción detallada

template<class T>class rpn_t< T >

Definición en la línea 21 del archivo rpn_t.h.

3.4.2. Documentación del constructor y destructor

3.4.2.1. template<class T> rpn_t< T >::rpn_t (void) [inline]

Definición en la línea 24 del archivo rpn_t.h.

3.4.2.2. template<class T> rpn_t< T >::~rpn_t ( ) [inline]

Definición en la línea 27 del archivo rpn_t.h.

3.4.3. Documentación de las funciones miembro

3.4.3.1. template<class T > const int rpn_t< T >::evaluate (queue_l_t< char > & *q*)

Definición en la línea 39 del archivo rpn_t.h.

Gráfico de llamadas para esta función:

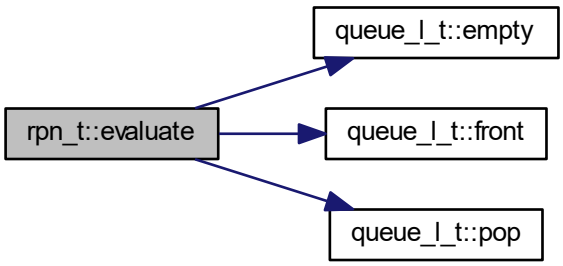
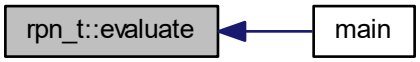


Gráfico de llamadas a esta función:



La documentación para esta clase fue generada a partir del siguiente fichero:

- rpn_t.h

3.5. Referencia de la plantilla de la Clase stack_l_t< T >

`#include <stack_l_t.h>`

Métodos públicos

- stack_l_t (void)
- ~stack_l_t (void)
- void push (const T &)
- void pop (void)
- const T & top (void) const
- bool empty (void) const
- std::ostream & write (std::ostream &os=std::cout) const

3.5.1. Descripción detallada

template<class T>class stack_l_t< T >

Definición en la línea 20 del archivo stack_l_t.h.

3.5.2. Documentación del constructor y destructor

3.5.2.1. template<class T > stack_l_t< T >::stack_l_t (void) [inline]

Definición en la línea 23 del archivo stack_l_t.h.

3.5.2.2. template<class T > stack_l_t< T >::~stack_l_t (void) [inline]

Definición en la línea 26 del archivo stack_l_t.h.

3.5.3. Documentación de las funciones miembro

3.5.3.1. template<class T > bool stack_l_t< T >::empty (void) const

Definición en la línea 59 del archivo stack_l_t.h.

3.5.3.2. template<class T > void stack_l_t< T >::pop (void)

Definición en la línea 49 del archivo stack_l_t.h.

3.5.3.3. template<class T > void stack_l_t< T >::push (const T & *dato*)

Definición en la línea 43 del archivo stack_l_t.h.

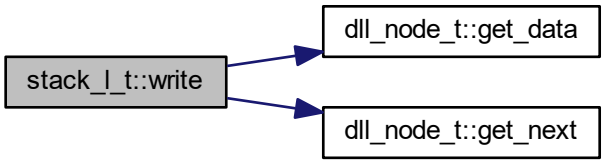
3.5.3.4. template<class T > const T & stack_l_t< T >::top (void) const

Definición en la línea 54 del archivo stack_l_t.h.

3.5.3.5. template<class T > std::ostream & stack_l_t< T >::write (std::ostream & *os* = std::cout) const

Definición en la línea 64 del archivo stack_l_t.h.

Gráfico de llamadas para esta función:



La documentación para esta clase fue generada a partir del siguiente fichero:

- stack_l_t.h

Capítulo 4

Documentación de archivos

4.1. Referencia del Archivo dll_node_t.h

```
#include <iostream>
```

Dependencia gráfica adjunta para dll_node_t.h:

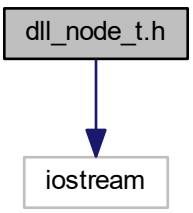
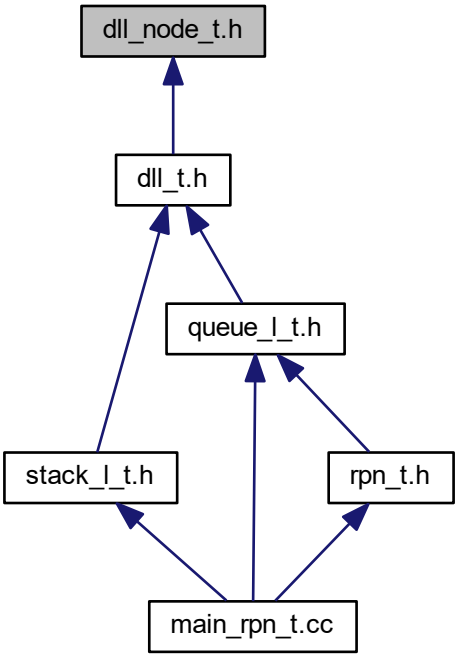


Gráfico de los archivos que directa o indirectamente incluyen a este archivo:



Clases

- class dll_node_t< T >

4.2. Referencia del Archivo dll_t.h

```
#include <cassert>
#include <iostream>
#include "dll_node_t.h"
```

Dependencia gráfica adjunta para dll_t.h:

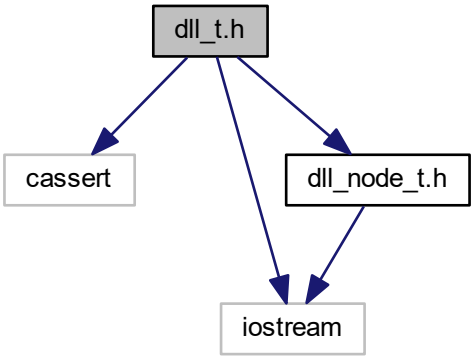
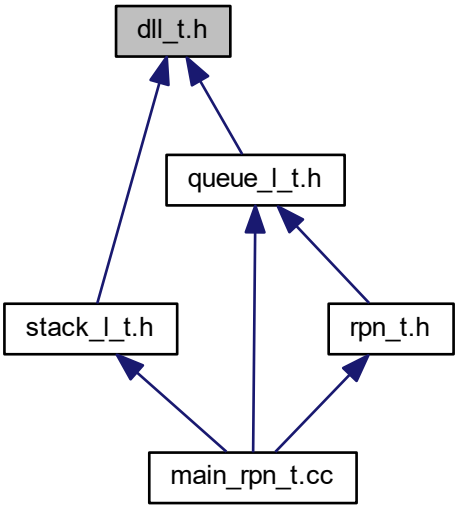


Gráfico de los archivos que directa o indirectamente incluyen a este archivo:



Clases

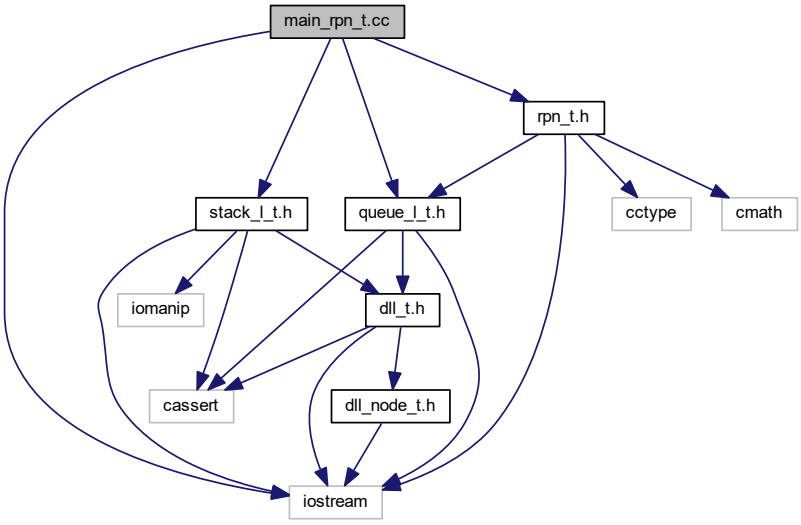
- class dll_t< T >

4.3. Referencia del Archivo main_rpn_t.cc

```
#include <iostream>
```

```
#include "stack_l_t.h"
#include "queue_l_t.h"
#include "rpn_t.h"
```

Dependencia gráfica adjunta para main_rpn_t.cc:



Funciones

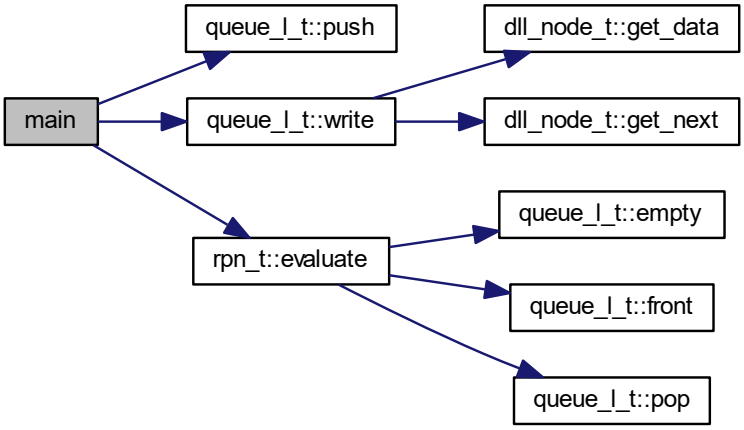
- int main (void)

4.3.1. Documentación de las funciones

4.3.1.1. int main (void)

Definición en la línea 27 del archivo main_rpn_t.cc.

Gráfico de llamadas para esta función:



4.4. Referencia del Archivo queue_l_t.h

```
#include <iostream>
#include <cassert>
#include "dll_t.h"
```

Dependencia gráfica adjunta para queue_l_t.h:

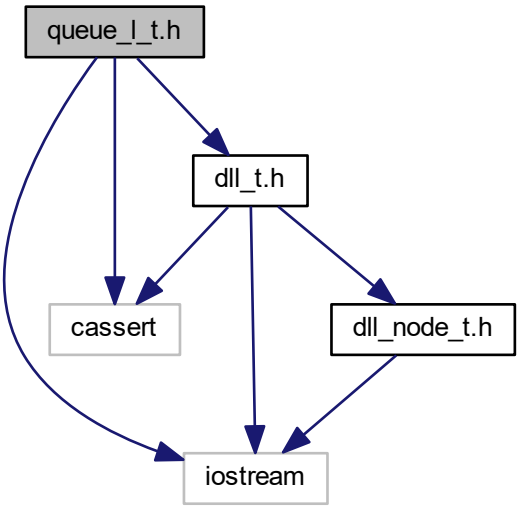
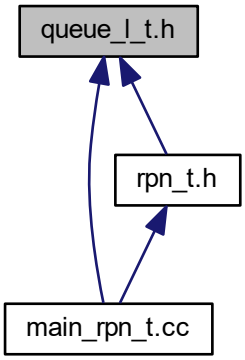


Gráfico de los archivos que directa o indirectamente incluyen a este archivo:



Clases

- class queue_l_t< T >

Funciones

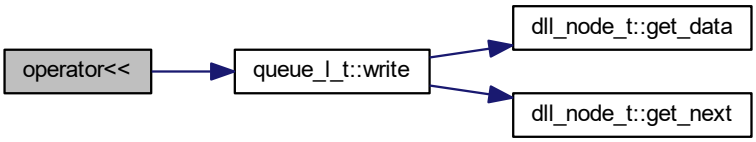
- template<class T >
  std::ostream & operator<< (std::ostream &os, const queue_l_t< T > &q)

4.4.1. Documentación de las funciones

4.4.1.1. template<class T > std::ostream& operator<< (std::ostream & *os*, const queue_l_t< T > & *q*)

Definición en la línea 84 del archivo queue_l_t.h.

Gráfico de llamadas para esta función:



4.5. Referencia del Archivo rpn_t.h

```
#include <iostream>
```

```
#include <cctype>
#include <cmath>
#include "queue_l_t.h"
```

Dependencia gráfica adjunta para rpn_t.h:

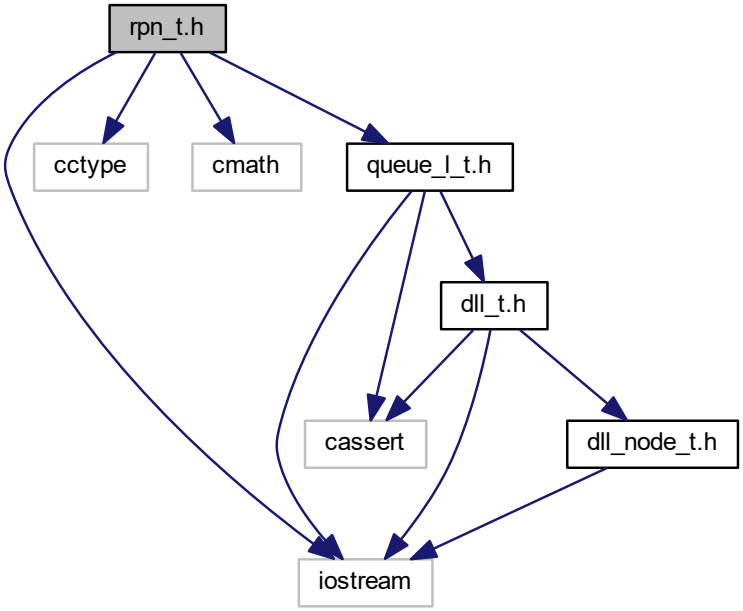
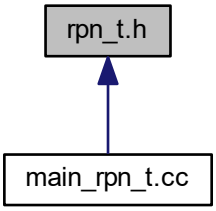


Gráfico de los archivos que directa o indirectamente incluyen a este archivo:



Clases

- class rpn_t< T >

4.6. Referencia del Archivo stack_l_t.h

```
#include <iostream>
```

```
#include <iomanip>
#include <cassert>
#include "dll_t.h"
```

Dependencia gráfica adjunta para stack_l_t.h:

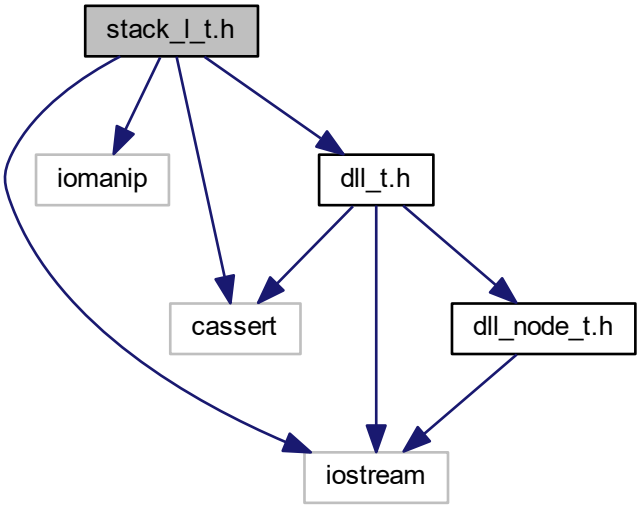
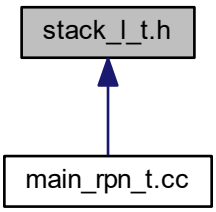


Gráfico de los archivos que directa o indirectamente incluyen a este archivo:



Clases

- class stack_l_t< T >

Índice alfabético

~dll_node_t
    dll_node_t, 5
~dll_t
    dll_t, 9
~queue_l_t
    queue_l_t, 13
~rpn_t
    rpn_t, 15
~stack_l_t
    stack_l_t, 17

back
    queue_l_t, 13

dll_node_t
    ~dll_node_t, 5
    dll_node_t, 5
    get_data, 6
    get_next, 6
    get_prev, 6
    set_data, 7
    set_next, 7
    set_prev, 7
    write, 8
dll_node_t< T >, 5
dll_node_t.h, 19
dll_t
    ~dll_t, 9
    dll_t, 9
    empty, 9
    erase, 9
    get_head, 10
    get_size, 10
    get_tail, 10
    pop_back, 10
    pop_front, 11
    push_back, 11
    push_front, 11
    write, 12
dll_t< T >, 8
dll_t.h, 20

empty
    dll_t, 9
    queue_l_t, 13
    stack_l_t, 17
erase
    dll_t, 9
evaluate
    rpn_t, 15

front
    queue_l_t, 13

get_data
    dll_node_t, 6
get_head
    dll_t, 10
get_next
    dll_node_t, 6
get_prev
    dll_node_t, 6
get_size
    dll_t, 10
get_tail
    dll_t, 10

main
    main_rpn_t.cc, 22
main_rpn_t.cc, 21
    main, 22

operator<<
    queue_l_t.h, 24

pop
    queue_l_t, 13
    stack_l_t, 17
pop_back
    dll_t, 10
pop_front
    dll_t, 11
push
    queue_l_t, 14
    stack_l_t, 17
push_back
    dll_t, 11
push_front
    dll_t, 11

queue_l_t
    ~queue_l_t, 13
    back, 13
    empty, 13
    front, 13
    pop, 13
    push, 14
    queue_l_t, 13
    size, 14
    write, 14
queue_l_t< T >, 12

queue_l_t.h, 23
    operator<<, 24

rpn_t
    ~rpn_t, 15
    evaluate, 15
    rpn_t, 15
rpn_t< T >, 15
rpn_t.h, 24

set_data
    dll_node_t, 7
set_next
    dll_node_t, 7
set_prev
    dll_node_t, 7
size
    queue_l_t, 14
stack_l_t
    ~stack_l_t, 17
    empty, 17
    pop, 17
    push, 17
    stack_l_t, 17
    top, 17
    write, 17
stack_l_t< T >, 16
stack_l_t.h, 25

top
    stack_l_t, 17

write
    dll_node_t, 8
    dll_t, 12
    queue_l_t, 14
    stack_l_t, 17