QueueList

1

Generato da Doxygen 1.9.1

1 README

Questo progetto si occupa di implementare una struttura di tipo CODA utilizzando le liste doppiamente linkate con sentinelle

2 Indice dei tipi composti

2.1 Elenco dei tipi composti

Queste sono le classi, le struct, le union e le interfacce con una loro breve descrizione:

3 Documentazione delle classi

3.1 Template per la classe Node< T >

Membri pubblici

- Node (T value)
- Node ()
- void setNext (Node< T > *next)
- void setPrev (Node< T > *prev)
- Node< T > * getNext ()
- Node< T > * getPrev ()
- T getData ()
- ~Node ()

Friend

- ostream & operator<< (ostream &out, Node< T > &n)

3.1.1 Documentazione dei costruttori e dei distruttori

3.1.1.1 Node() [1/2] template<class T >

```
Node< T >::Node (
            T value )  [inline]
```

Costruttore che riceve in ingresso un valore generico

3.1.1.2 Node() [2/2] template<class T >

```
Node< T >::Node ( )  [inline]
```

Costruttore di default per un Node vuoto

3.1.1.3 ~Node() template<class T >

```
Node< T >::~Node ( )  [inline]
```

Decostruttore per eliminare il nodo

3.1.2 Documentazione delle funzioni membro

3.1.2.1 getData() template<class T >

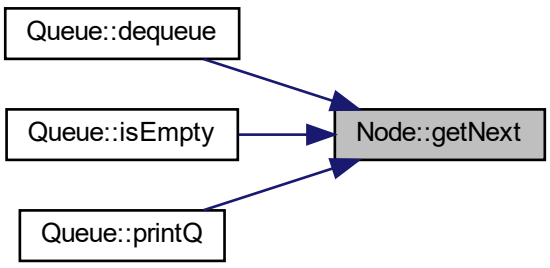
```
T Node< T >::getData ( )  [inline]
```

Metodo che restituisce il valore del Node

3.1.2.2 getNext() template<class T >

```
Node<T>* Node< T >::getNext ( )  [inline]
```

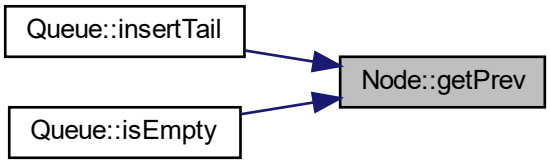
Metodo che ritorna il puntatore al Node successivo Questo è il grafo dei chiamanti di questa funzione:



3.1.2.3 getPrev() template<class T >

```
Node<T>* Node< T >::getPrev ( )  [inline]
```

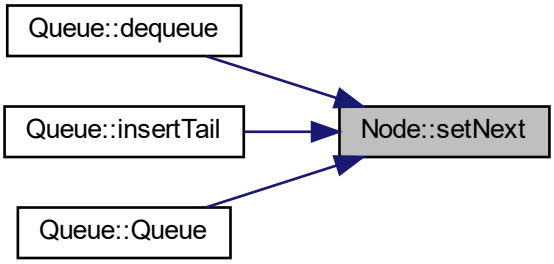
Metodo che ritorna il puntatore al Node precedente Questo è il grafo dei chiamanti di questa funzione:



3.1.2.4 setNext() template<class T >

```
void Node< T >::setNext (
            Node< T > * next )  [inline]
```

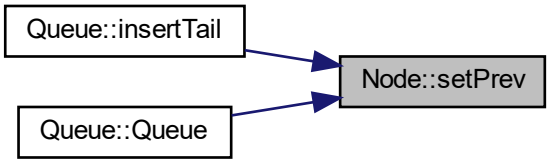
Metodo per impostare il Node successivo Questo è il grafo dei chiamanti di questa funzione:



3.1.2.5 setPrev() template<class T >

```
void Node< T >::setPrev (
            Node< T > * prev )  [inline]
```

Metodo per impostare il Node precedente Questo è il grafo dei chiamanti di questa funzione:



3.1.3 Documentazione dei friend e delle funzioni collegate

3.1.3.1 operator<< template<class T > ostream& operator<< (ostream & out, Node< T > & n) [friend]

Overriding dell <<operator per stampare i dati dei nodi tramite cout

La documentazione per questa classe è stata generata a partire dal seguente file:

- include/Node.h

3.2 Template per la classe Queue< T >

Membri pubblici

- Queue ()
- bool isEmpty ()
- Node< T > * getHead ()
- Node< T > * getTail ()
- void insertTail (T data)
- void printQ ()
- void dequeue ()
- ~Queue ()

Friend

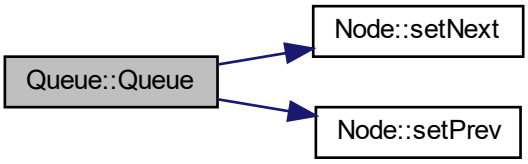
- ostream & operator<< (ostream &out, Queue< T > &q)

3.2.1 Documentazione dei costruttori e dei distruttori

3.2.1.1 Queue() template<class T >

Queue< T >::Queue ( ) [inline]

Costruttore di defaul senza parametri in ingresso che setta head e tail come due nodi vuoti, dove il successivo della Head punta alla Tail e il precedente della Tail punta alla Head Questo è il grafo delle chiamate per questa funzione:



3.2.1.2 ~Queue() template<class T >

Queue< T >::~Queue ( ) [inline]

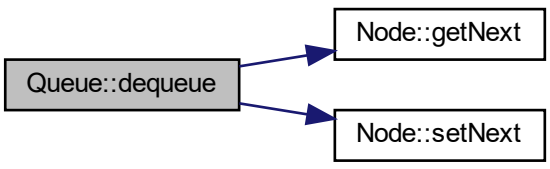
Decostruttore dove elimino head, tail e length

3.2.2 Documentazione delle funzioni membro

3.2.2.1 dequeue() template<class T >

void Queue< T >::dequeue ( ) [inline]

Metodo che si occupa della dequeue impostando il successivo di head come il successivo del nodo da rimuovere dalla coda e il precedente del successivo del nodo da riumere come head Questo è il grafo delle chiamate per questa funzione:



3.2.2.2 getHead() template<class T >

Node<T>* Queue< T >::getHead ( ) [inline]

Metodo che ritorna il puntatore alla Head

3.2.2.3 getTail() template<class T >

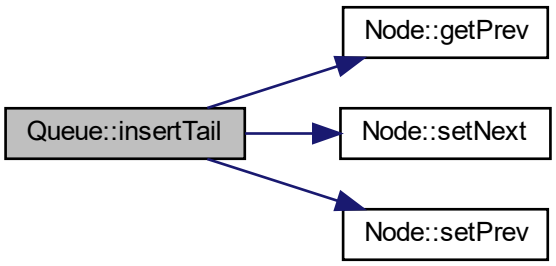
Node<T>* Queue< T >::getTail ( ) [inline]

Metodo che ritorna il puntatore alla Tail

3.2.2.4 insertTail() template<class T >

void Queue< T >::insertTail (
T *data*) [inline]

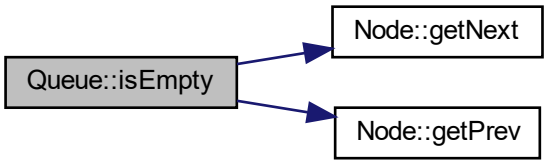
Metodo che inserisce i nodi in coda(Queue) il metodo si suddivide in due casi gestiti dalla condizione di isEmpty() nel caso la Queue sia vuota allora inserisco il nodo e aggiorno i link nel caso in cui non sia vuota allora inserisco il nodo in coda Questo è il grafo delle chiamate per questa funzione:



3.2.2.5 isEmpty() template<class T >

bool Queue< T >::isEmpty ( ) [inline]

Metodo che ritorna 1 se la Queue ota cio successivo della Head il e il precedente della tail ad in caso contrario ritorna 0 Questo è il grafo delle chiamate per questa funzione:



3.2.2.6 printQ() template<class T >

```
void Queue< T >::printQ ( )  [inline]
```

Metodo che gestisce la stampa dei nodi la stampa parte dal primo elemento fino a quando non si raggiunge la Tail che non viene stampata Questo è il grafo delle chiamate per questa funzione:

3.2.3 Documentazione dei friend e delle funzioni collegate

3.2.3.1 operator<< template<class T >

```
ostream& operator<< (
            ostream & out,
            Queue< T > & q )  [friend]
```

Overriding dell <<operator per stampare i dati dei nodi tramite cout della coda

La documentazione per questa classe è stata generata a partire dal seguente file:

- include/Queue.h

Indice analitico