

Retroactive Data Structures

1.0

Generated by Doxygen 1.8.13

Contents

1	Namespace Index	1
1.1	Namespace List	1
2	Class Index	3
2.1	Class List	3
3	Namespace Documentation	5
3.1	Retroactivity Namespace Reference	5
3.1.1	Detailed Description	5
4	Class Documentation	7
4.1	Retroactivity::FullPriorityQueue< T > Class Template Reference	7
4.1.1	Detailed Description	7
4.1.2	Member Function Documentation	7
4.1.2.1	empty()	7
4.1.2.2	getPeak()	8
4.1.2.3	insertPop()	8
4.1.2.4	insertPush()	9
4.1.2.5	removePop()	9
4.1.2.6	removePush()	9
4.2	Brute::FullPriorityQueue< T > Class Template Reference	10
4.3	Retroactivity::FullQueue< T > Class Template Reference	10
4.3.1	Detailed Description	10
4.3.2	Member Function Documentation	10
4.3.2.1	DeleteDequeue()	10

4.3.2.2	DeleteEnqueue()	11
4.3.2.3	front()	11
4.3.2.4	getKth()	12
4.3.2.5	InsertDequeue()	12
4.3.2.6	InsertEnqueue()	12
4.4	Brute::FullQueue< T > Class Template Reference	13
4.5	Retroactivity::FullStack< T > Class Template Reference	13
4.5.1	Detailed Description	14
4.5.2	Member Function Documentation	14
4.5.2.1	DeletePop()	14
4.5.2.2	DeletePush()	14
4.5.2.3	getSize()	15
4.5.2.4	InsertPop()	15
4.5.2.5	InsertPush()	15
4.5.2.6	peak()	15
4.6	Brute::FullStack< T > Class Template Reference	16
4.7	IntervalTreeSum< T > Class Template Reference	16
4.8	Retroactivity::PolylogarithmPriorityQueue< T >::Node< T > Class Template Reference	17
4.9	Retroactivity::NonObliviousPriorityQueue< T > Class Template Reference	17
4.9.1	Detailed Description	17
4.9.2	Member Function Documentation	18
4.9.2.1	DeletePop()	18
4.9.2.2	DeletePush()	18
4.9.2.3	fixOperation()	19
4.9.2.4	InsertPop()	19
4.9.2.5	InsertPush()	19
4.9.2.6	Peak()	20
4.10	Retroactivity::NonObliviousQueue< T > Class Template Reference	20
4.10.1	Detailed Description	20
4.10.2	Member Function Documentation	21

4.10.2.1	DeleteDequeue()	21
4.10.2.2	DeleteEnqueue()	21
4.10.2.3	fixDequeueOperation()	22
4.10.2.4	Front()	22
4.10.2.5	InsertDequeue()	22
4.10.2.6	InsertEnqueue()	23
4.10.2.7	showTd()	23
4.10.2.8	showTe()	24
4.11	Retroactivity::NonObliviousStack< T > Class Template Reference	24
4.11.1	Detailed Description	24
4.11.2	Member Function Documentation	24
4.11.2.1	DeletePop()	24
4.11.2.2	DeletePush()	25
4.11.2.3	fixPopOperation()	25
4.11.2.4	InsertPop()	26
4.11.2.5	InsertPush()	26
4.11.2.6	showtpop()	26
4.11.2.7	showtpush()	27
4.12	Retroactivity::FullPriorityQueue< T >::Operation< T > Class Template Reference	27
4.12.1	Detailed Description	27
4.13	Brute::PartialPriorityQueue< T > Class Template Reference	27
4.14	Retroactivity::PartialPriorityQueue< T > Class Template Reference	28
4.14.1	Detailed Description	28
4.14.2	Constructor & Destructor Documentation	28
4.14.2.1	PartialPriorityQueue()	28
4.14.3	Member Function Documentation	28
4.14.3.1	empty()	29
4.14.3.2	getPeak()	29
4.14.3.3	insertPop()	29
4.14.3.4	insertPush()	30

4.14.3.5	removePop()	30
4.14.3.6	removePush()	30
4.15	Brute::PartialQueue< T > Class Template Reference	31
4.16	Retroactivity::PartialQueue< T > Class Template Reference	31
4.16.1	Detailed Description	32
4.16.2	Member Function Documentation	32
4.16.2.1	back()	32
4.16.2.2	DeleteDequeue()	32
4.16.2.3	DeleteEnqueue()	33
4.16.2.4	front()	33
4.16.2.5	InsertDequeue()	33
4.16.2.6	InsertEnqueue()	34
4.17	Brute::PartialStack< T > Class Template Reference	34
4.18	Retroactivity::PartialStack< T > Class Template Reference	34
4.18.1	Detailed Description	35
4.18.2	Member Function Documentation	35
4.18.2.1	DeletePop()	35
4.18.2.2	DeletePush()	35
4.18.2.3	getSize()	36
4.18.2.4	InsertPop()	36
4.18.2.5	InsertPush()	36
4.18.2.6	peak()	37
4.19	Retroactivity::PolylogarithmPriorityQueue< T > Class Template Reference	37
4.19.1	Detailed Description	38
4.19.2	Constructor & Destructor Documentation	38
4.19.2.1	PolylogarithmPriorityQueue()	38
4.19.3	Member Function Documentation	38
4.19.3.1	addPop()	38
4.19.3.2	addPush()	39
4.19.3.3	Fuse()	39

4.19.3.4	getMinimumKey()	40
4.19.3.5	getNodes()	40
4.19.3.6	getPeak()	41
4.19.3.7	getQtdGE()	41
4.19.3.8	getQueryNodeSize()	42
4.19.3.9	getSplitedTrees()	42
4.19.3.10	getSplitKey()	42
4.19.3.11	getView()	43
4.19.3.12	insertPop()	43
4.19.3.13	insertPush()	44
4.19.3.14	query()	44
4.19.3.15	removePop()	45
4.19.3.16	removePush()	45
4.20	Retroactivity::PolylogarithmPriorityQueue< T >::QueryNode< T > Class Template Reference	45

Chapter 1

Namespace Index

1.1 Namespace List

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

Retroactivity	5
---	---

Chapter 2

Class Index

2.1 Class List

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

Retroactivity::FullPriorityQueue< T >	7
Brute::FullPriorityQueue< T >	10
Retroactivity::FullQueue< T >	10
Brute::FullQueue< T >	13
Retroactivity::FullStack< T >	13
Brute::FullStack< T >	16
IntervalTreeSum< T >	16
Retroactivity::PolylogarithmPriorityQueue< T >::Node< T >	17
Retroactivity::NonObliviousPriorityQueue< T >	17
Retroactivity::NonObliviousQueue< T >	20
Retroactivity::NonObliviousStack< T >	24
Retroactivity::FullPriorityQueue< T >::Operation< T >	27
Brute::PartialPriorityQueue< T >	27
Retroactivity::PartialPriorityQueue< T >	28
Brute::PartialQueue< T >	31
Retroactivity::PartialQueue< T >	31
Brute::PartialStack< T >	34
Retroactivity::PartialStack< T >	34
Retroactivity::PolylogarithmPriorityQueue< T >	37
Retroactivity::PolylogarithmPriorityQueue< T >::QueryNode< T >	45

Chapter 3

Namespace Documentation

3.1 Retroactivity Namespace Reference

Classes

- class [FullPriorityQueue](#)
- class [FullQueue](#)
- class [FullStack](#)
- class [NonObliviousPriorityQueue](#)
- class [NonObliviousQueue](#)
- class [NonObliviousStack](#)
- class [PartialPriorityQueue](#)
- class [PartialQueue](#)
- class [PartialStack](#)
- class [PolylogarithmPriorityQueue](#)

Typedefs

- typedef pair< int, int > **ii**

3.1.1 Detailed Description

Implementação retroativa das operações em uma fila de prioridade

Implementação retroativa das operações em uma fila

Implementação retroativa das operações em uma pilha

Chapter 4

Class Documentation

4.1 Retroactivity::FullPriorityQueue< T > Class Template Reference

```
#include <Priority_queue.hpp>
```

Classes

- class [Operation](#)

Public Member Functions

- **FullPriorityQueue** (int _m)
- void [insertPush](#) (int t, T data)
- void [insertPop](#) (int t)
- void [removePush](#) (int t)
- void [removePop](#) (int t)
- bool [empty](#) (int t)
- T [getPeak](#) (int t)

4.1.1 Detailed Description

```
template<typename T>
class Retroactivity::FullPriorityQueue< T >
```

Fila de prioridade totalmente retroativa em tempo $O(\sqrt{m}) * \log(m)$ por operação

4.1.2 Member Function Documentation

4.1.2.1 empty()

```
template<typename T >
bool Retroactivity::FullPriorityQueue< T >::empty (
    int t )
```

Retorna se a fila está vazia no tempo $t \Rightarrow \text{Empty}(t)$

Parameters

<i>t</i>	-> tempo em que se deseja saber se a fila está vazia
----------	--

Returns

-> 'true' se a fila estiver vazia no tempo *t*

4.1.2.2 getPeak()

```
template<typename T >  
T Retroactivity::FullPriorityQueue< T >::getPeak (   
    int t )
```

Retorna o menor elemento da fila de prioridade no tempo *t* => GetPeak(*t*)

Parameters

<i>t</i>	-> tempo em que se deseja consultar o menor elemento da fila
----------	--

Returns

-> o menor elemento da fila de prioridade após a execução de todas as operações até o tempo *t*

Precondition

-> a estrutura deve conter pelo menos um elemento no tempo *t*

4.1.2.3 insertPop()

```
template<typename T >  
void Retroactivity::FullPriorityQueue< T >::insertPop (   
    int t )
```

Insere uma operação de Pop() no tempo *t* => Insert(*t*, Pop())

Parameters

<i>t</i>	-> tempo em que a operação Pop() foi realizada
----------	--

Precondition

->

4.1.2.4 insertPush()

```
template<typename T >
void Retroactivity::FullPriorityQueue< T >::insertPush (
    int t,
    T data )
```

Insere uma operação de Push(data) no tempo t => Insert(t, Push(data))

Parameters

<i>t</i>	-> tempo em que a operação Push(data) foi realizada
----------	---

4.1.2.5 removePop()

```
template<typename T >
void Retroactivity::FullPriorityQueue< T >::removePop (
    int t )
```

Remove uma operação de Pop() no tempo t => Delete(t, Pop())

Parameters

<i>t</i>	-> tempo em que a operação Pop() foi realizada
----------	--

Precondition

-> deve existir uma operação Delete(t, Pop()) na estrutura

4.1.2.6 removePush()

```
template<typename T >
void Retroactivity::FullPriorityQueue< T >::removePush (
    int t )
```

Remove uma operação de Push(data) no tempo t => Delete(t, Push(data))

Parameters

<i>t</i>	-> tempo em que a operação Push(data) foi realizada
----------	---

Precondition

-> deve existir uma operação Insert(t, Push(data)) na estrutura

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

- /home/junior/Dropbox/ArquivosMestrado/Master/src/Priority_Queue/Priority_queue.hpp
- /home/junior/Dropbox/ArquivosMestrado/Master/src/Priority_Queue/Priority_queue.cpp

4.2 Brute::FullPriorityQueue< T > Class Template Reference

Public Member Functions

- void **insertPush** (int t, T data)
- void **insertPop** (int t)
- void **removePush** (int t)
- void **removePop** (int t)
- T **getPeak** (int t)
- bool **empty** (int t)

Public Attributes

- std::multiset< pair< int, pair< int, T > > > **all**

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

- /home/junior/Dropbox/ArquivosMestrado/Master/src/Priority_Queue/Priority_queue.hpp

4.3 Retroactivity::FullQueue< T > Class Template Reference

```
#include <Queue.hpp>
```

Public Member Functions

- void **InsertEnqueue** (int t, const T &x)
- void **InsertDequeue** (int t)
- void **DeleteEnqueue** (int t)
- void **DeleteDequeue** (int t)
- T **getKth** (int t, int k)
- T **front** (int t)

4.3.1 Detailed Description

```
template<typename T>
class Retroactivity::FullQueue< T >
```

Fila totalmente retroativa

4.3.2 Member Function Documentation

4.3.2.1 DeleteDequeue()

```
template<typename T >
void Retroactivity::FullQueue< T >::DeleteDequeue (
    int t )
```

Remove a operação de Dequeue realizada no tempo t => Delete(t, Dequeue())

Parameters

<i>t</i>	-> tempo em que uma operação Dequeue foi realizada
----------	--

Precondition

-> precisa que uma operação Dequeue no tempo *t* tenha sido realizada

4.3.2.2 DeleteEnqueue()

```
template<typename T >
void Retroactivity::FullQueue< T >::DeleteEnqueue (
    int t )
```

Remove a operação de Enqueue realizada no tempo *t* => Delete(*t*, Enqueue(*x*))

Parameters

<i>t</i>	-> tempo em que uma operação Enqueue foi realizada
----------	--

Precondition

-> precisa que uma operação Enqueue no tempo *t* tenha sido realizada

4.3.2.3 front()

```
template<typename T >
T Retroactivity::FullQueue< T >::front (
    int t )
```

Retorna o proximo elemento a ser removido da fila no tempo *t* => Front(*t*)

Parameters

<i>t</i>	-> tempo em que se deseja consultar o próximo elemento da fila
----------	--

Returns

-> O proximo elemento a ser removido da fila no tempo *t*

Precondition

-> a fila não pode estar vazia no tempo *t*

4.3.2.4 getKth()

```
template<typename T >
T Retroactivity::FullQueue< T >::getKth (
    int t,
    int k )
```

Consulta qual o índice do k-ésimo elemento da fila considerando as operações de Dequeue realizadas

Parameters

<i>t</i>	-> tempo em que a consulta é realizada
<i>k</i>	-> índice do elemento consultado

Returns

-> O k-ésimo elemento da fila no tempo t

Precondition

-> a fila não pode estar vazia no tempo t e k deve ser menor que o número de elemento na fila no tempo t

4.3.2.5 InsertDequeue()

```
template<typename T >
void Retroactivity::FullQueue< T >::InsertDequeue (
    int t )
```

Inserir uma operação Dequeue() no tempo t => Insert(t, Dequeue())

Recebe um inteiro t (tempo de inserção da operação) em que a operação Dequeue será inserida

Parameters

<i>t</i>	-> tempo de inserção da operação Dequeue
----------	--

4.3.2.6 InsertEnqueue()

```
template<typename T >
void Retroactivity::FullQueue< T >::InsertEnqueue (
    int t,
    const T & x )
```

Inserir uma operação Enqueue(x) no tempo t => Insert(t, Enqueue(x))

Recebe um inteiro t (tempo de inserção da operação) em que a operação Enqueue(x) será inserida

Parameters

<i>t</i>	-> tempo de inserção da operação Enqueue(x)
<i>x</i>	-> objeto a ser inserido na operação

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

- /home/junior/Dropbox/ArquivosMestrado/Master/src/Queue/Queue.hpp
- /home/junior/Dropbox/ArquivosMestrado/Master/src/Queue/Queue.cpp

4.4 Brute::FullQueue< T > Class Template Reference

Public Member Functions

- void **InsertEnqueue** (int t, const T &x)
- void **InsertDequeue** (int t)
- void **DeleteEnqueue** (int t)
- void **DeleteDequeue** (int t)
- T **front** (int t)

Public Attributes

- map< int, T > **q**

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

- /home/junior/Dropbox/ArquivosMestrado/Master/src/Queue/Queue.hpp

4.5 Retroactivity::FullStack< T > Class Template Reference

```
#include <Stack.hpp>
```

Public Member Functions

- void **InsertPush** (int t, const T &x)
- void **InsertPop** (int t)
- void **DeletePush** (int t)
- void **DeletePop** (int t)
- T **peak** (int t)
- int **getSize** (int t)

4.5.1 Detailed Description

```
template<typename T>  
class Retroactivity::FullStack< T >
```

Pilha totalmente retroativa

4.5.2 Member Function Documentation

4.5.2.1 DeletePop()

```
template<typename T >  
void Retroactivity::FullStack< T >::DeletePop (  
    int t )
```

Remove a operação Pop realizada no tempo t => Delete(t, Pop())

Parameters

<i>t</i>	-> tempo em que uma operação Pop foi realizada
----------	--

Precondition

-> deve existir uma operação Pop no tempo t

4.5.2.2 DeletePush()

```
template<typename T >  
void Retroactivity::FullStack< T >::DeletePush (  
    int t )
```

Remove a operação Push realizada no tempo t => Delete(t, Push(x))

Parameters

<i>t</i>	-> tempo em que uma operação Push foi realizada
----------	---

Precondition

-> deve existir uma operação Push no tempo t

4.5.2.3 getSize()

```
template<typename T >
int Retroactivity::FullStack< T >::getSize (
    int t )
```

Obtem o tamanho da pilha no tempo atual => getSize(t)

t -> tempo que é desejado o tamanho da pilha

Returns

-> retorna o número de elementos na pilha no tempo t

4.5.2.4 InsertPop()

```
template<typename T >
void Retroactivity::FullStack< T >::InsertPop (
    int t )
```

Adiciona a realização da operação Pop no tempo t => Insert(t, Pop())

Parameters

<i>t</i>	-> tempo de realização da operação Pop()
----------	--

4.5.2.5 InsertPush()

```
template<typename T >
void Retroactivity::FullStack< T >::InsertPush (
    int t,
    const T & x )
```

Insere o objeto x no tempo t na pilha => Insert(t, Push(x))

Parameters

<i>t</i>	-> tempo de inserção do objeto
<i>x</i>	-> objeto a ser inserido no topo da pilha no tempo t

4.5.2.6 peak()

```
template<typename T >
```

```
T Retroactivity::FullStack< T >::peak (
    int t )
```

Retorna o topo da pilha no tempo t => Peak(t)

t -> tempo que é desejado o elemento ao topo da pilha

Returns

-> retorna objeto no topo da pilha no tempo t

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

- /home/junior/Dropbox/ArquivosMestrado/Master/src/Stack/Stack.hpp
- /home/junior/Dropbox/ArquivosMestrado/Master/src/Stack/Stack.cpp

4.6 Brute::FullStack< T > Class Template Reference

Public Member Functions

- void **InsertPush** (int t, const T &x)
- void **InsertPop** (int t)
- void **DeletePush** (int t)
- void **DeletePop** (int t)
- bool **empty** (int t)
- T **peak** (int t)

Public Attributes

- set< pair< int, T > > **v**
- vector< T > **aux**

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

- /home/junior/Dropbox/ArquivosMestrado/Master/src/Stack/Stack.hpp

4.7 IntervalTreeSum< T > Class Template Reference

Public Member Functions

- int **createNode** ()
- void **propagate** (int no, int l, int r)
- void **update** (int no, int l, int r, int t, int inc)
- Node **query** (int no, int l, int r, int i, int j)
- void **showTree** (int no, int l, int r)
- void **update** (int t, T inc)
- Node **query** (int l, int r)
- void **showTree** ()
- int **getLastBridge** (int t)
- int **getNextBridge** (int t)

Public Attributes

- `std::vector< Node > tr`
- `std::vector< int > lz`
- `std::vector< int > L`
- `std::vector< int > R`
- `int n`

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

- `/home/junior/Dropbox/ArquivosMestrado/Master/src/Priority_Queue/IntervalTree.cpp`

4.8 Retroactivity::PolylogarithmPriorityQueue< T >::Node< T > Class Template Reference

Public Attributes

- [Retroactivity::PartialPriorityQueue< T >](#) `pq`
- `PersistentTreap< T, int >` `qnow`
- `PersistentTreap< T, int >` `qdel`
- `int qnowV`
- `int qdelV`

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

- `/home/junior/Dropbox/ArquivosMestrado/Master/src/Priority_Queue/Priority_queue.cpp`

4.9 Retroactivity::NonObliviousPriorityQueue< T > Class Template Reference

```
#include <Priority_queue.hpp>
```

Public Member Functions

- `int InsertPush (int t, T x)`
- `int InsertPop (int t)`
- `int DeletePush (int t)`
- `int DeletePop (int t)`
- `T Peak (int t)`
- `void fixOperation (int t, Operation type)`

4.9.1 Detailed Description

```
template<typename T>
class Retroactivity::NonObliviousPriorityQueue< T >
```

Fila de prioridade com retroatividade não-consistente

4.9.2 Member Function Documentation

4.9.2.1 DeletePop()

```
template<typename T >
int Retroactivity::NonObliviousPriorityQueue< T >::DeletePop (
    int t )
```

Remove a operação de Pop realizada no tempo $t \Rightarrow \text{Delete}(t, \text{Pop}())$

Parameters

t	-> tempo em que uma operação Pop foi realizada
-----	--

Precondition

-> precisa que uma operação Pop no tempo t tenha sido realizada

Returns

-> um inteiro com o tempo da primeira operação inconsistente

4.9.2.2 DeletePush()

```
template<typename T >
int Retroactivity::NonObliviousPriorityQueue< T >::DeletePush (
    int t )
```

Remove a operação de Push realizada no tempo $t \Rightarrow \text{Delete}(t, \text{Push}(x))$

Parameters

t	-> tempo em que uma operação Push(x) foi realizada
-----	--

Precondition

-> precisa que uma operação Push no tempo t tenha sido realizada

Returns

-> um inteiro com o tempo da primeira operação inconsistente

4.9.2.3 fixOperation()

```
template<typename T >
void Retroactivity::NonObliviousPriorityQueue< T >::fixOperation (
    int t,
    Operation type )
```

Corrige a estrutura após uma inconsistencia no tempo t

Parameters

<i>t</i>	-> tempo em que ocorreu uma inconsistencia na estrutura
----------	---

4.9.2.4 InsertPop()

```
template<typename T >
int Retroactivity::NonObliviousPriorityQueue< T >::InsertPop (
    int t )
```

Insere uma operação de Pop() no tempo t => Insert(t, Pop())

Parameters

<i>t</i>	-> tempo em que a operação Pop() foi realizada
----------	--

Returns

-> primeira operação inconsistente apos a execução da operação

4.9.2.5 InsertPush()

```
template<typename T >
int Retroactivity::NonObliviousPriorityQueue< T >::InsertPush (
    int t,
    T x )
```

Insere uma operação de Push(data) no tempo t => Insert(t, Push(data))

Parameters

<i>t</i>	-> tempo em que a operação Push(data) foi realizada
----------	---

Returns

-> primeira operação inconsistente apos a execução da operação

4.9.2.6 Peak()

```
template<typename T >
T Retroactivity::NonObliviousPriorityQueue< T >::Peak (
    int t )
```

Obtem o elemento minimo na estrutura no tempo t

Parameters

<i>t</i>	-> o tempo que deseja-se consultar o menor elemento na estrutura
----------	--

Returns

-> o menor elemento na estrutura no tempo t

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

- /home/junior/Dropbox/ArquivosMestrado/Master/src/Priority_Queue/Priority_queue.hpp
- /home/junior/Dropbox/ArquivosMestrado/Master/src/Priority_Queue/Priority_queue.cpp

4.10 Retroactivity::NonObliviousQueue< T > Class Template Reference

```
#include <Queue.hpp>
```

Public Member Functions

- int [InsertEnqueue](#) (int t, T x)
- int [InsertDequeue](#) (int t)
- int [DeleteEnqueue](#) (int t)
- int [DeleteDequeue](#) (int t)
- BST::Treap< int, std::pair< T, int > >::iterator [Front](#) (int t)
- void [fixDequeueOperation](#) (int t)
- void [showTe](#) ()
- void [showTd](#) ()

4.10.1 Detailed Description

```
template<typename T>
class Retroactivity::NonObliviousQueue< T >
```

Fila com retroatividade não consistente

4.10.2 Member Function Documentation

4.10.2.1 DeleteDequeue()

```
template<typename T >
int Retroactivity::NonObliviousQueue< T >::DeleteDequeue (
    int t )
```

Remove a operação de Dequeue realizada no tempo $t \Rightarrow \text{Delete}(t, \text{Dequeue}())$

Parameters

t	-> tempo em que uma operação Dequeue foi realizada
-----	--

Precondition

-> precisa que uma operação Dequeue no tempo t tenha sido realizada

Returns

-> um inteiro com o tempo da primeira operação Dequeue inconsistente

4.10.2.2 DeleteEnqueue()

```
template<typename T >
int Retroactivity::NonObliviousQueue< T >::DeleteEnqueue (
    int t )
```

Remove a operação de Enqueue realizada no tempo $t \Rightarrow \text{Delete}(t, \text{Enqueue}(x))$

Parameters

t	-> tempo em que uma operação Enqueue foi realizada
-----	--

Precondition

-> precisa que uma operação Enqueue no tempo t tenha sido realizada

Returns

-> um inteiro com o tempo da primeira operação Dequeue inconsistente

4.10.2.3 fixDequeueOperation()

```
template<typename T >
void Retroactivity::NonObliviousQueue< T >::fixDequeueOperation (
    int t )
```

Corrige a estrutura após uma inconsistencia no tempo t

Parameters

<i>t</i>	-> tempo em que ocorreu uma inconsistencia na estrutura
----------	---

4.10.2.4 Front()

```
template<typename T >
BST::Treap< int, std::pair< T, int > >::iterator Retroactivity::NonObliviousQueue< T >↔
::Front (
    int t )
```

Remove a operação de Dequeue realizada no tempo t => Front(t)

Parameters

<i>t</i>	-> tempo em que deseja-se saber o elemento na frente da fila
----------	--

Precondition

-> precisa que exista um elemento na fila

Returns

-> o iterator do elemento na posição frontal da fila

4.10.2.5 InsertDequeue()

```
template<typename T >
int Retroactivity::NonObliviousQueue< T >::InsertDequeue (
    int t )
```

Insere uma operação Dequeue() no tempo t => Insert(t, Dequeue())

Recebe um inteiro t (tempo de inserção da operação) em que a operação Dequeue será inserida

Parameters

<i>t</i>	-> tempo de inserção da operação Dequeue
----------	--

Returns

-> um inteiro com o tempo da primeira operação Dequeue inconsistente

4.10.2.6 InsertEnqueue()

```
template<typename T >
int Retroactivity::NonObliviousQueue< T >::InsertEnqueue (
    int t,
    T x )
```

Insere uma operação Enqueue(x) no tempo t => Insert(t, Enqueue(x))

Recebe um inteiro t (tempo de inserção da operação) em que a operação Enqueue(x) será inserida

Parameters

<i>t</i>	-> tempo de inserção da operação Enqueue(x)
<i>x</i>	-> objeto a ser inserido na operação

Returns

-> um inteiro com o tempo da primeira operação Dequeue inconsistente

4.10.2.7 showTd()

```
template<typename T >
void Retroactivity::NonObliviousQueue< T >::showTd ( )
```

Função auxiliar para mostrar os elementos do conjunto das remocoes

Returns

-> os elementos do conjunto Td no formato "tempoDelecao, (elementoDeletado)"

4.10.2.8 showTe()

```
template<typename T >
void Retroactivity::NonObliviousQueue< T >::showTe ( )
```

Função auxiliar para mostrar os elementos do conjunto das inserções

Returns

-> os elementos do conjunto Te no formato "tempoInsercao, (elemento, tempoRemocao)"

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

- /home/junior/Dropbox/ArquivosMestrado/Master/src/Queue/Queue.hpp
- /home/junior/Dropbox/ArquivosMestrado/Master/src/Queue/Queue.cpp

4.11 Retroactivity::NonObliviousStack< T > Class Template Reference

```
#include <Stack.hpp>
```

Public Member Functions

- int [InsertPush](#) (int t, T x)
- int [InsertPop](#) (int t)
- int [DeletePush](#) (int t)
- int [DeletePop](#) (int t)
- void [fixPopOperation](#) (int t)
- void [showtpush](#) ()
- void [showtpop](#) ()

4.11.1 Detailed Description

```
template<typename T>
class Retroactivity::NonObliviousStack< T >
```

Pilha com retroatividade não consistente

4.11.2 Member Function Documentation

4.11.2.1 DeletePop()

```
template<typename T >
int Retroactivity::NonObliviousStack< T >::DeletePop (
    int t )
```

Remove a operação Pop realizada no tempo t => Delete(t, Pop())

Parameters

<i>t</i>	-> tempo em que uma operação Pop foi realizada
----------	--

Precondition

-> deve existir uma operação Pop no tempo t

Returns

-> a próxima operação Pop inconsistente

4.11.2.2 DeletePush()

```
template<typename T >
int Retroactivity::NonObliviousStack< T >::DeletePush (
    int t )
```

Remove a operação Push realizada no tempo t => Delete(t, Push(x))

Parameters

<i>t</i>	-> tempo em que uma operação Push foi realizada
----------	---

Precondition

-> deve existir uma operação Push no tempo t

Returns

-> a próxima operação Pop inconsistente

4.11.2.3 fixPopOperation()

```
template<typename T >
void Retroactivity::NonObliviousStack< T >::fixPopOperation (
    int t )
```

Corrige a estrutura após uma inconsistencia no tempo t

Parameters

<i>t</i>	-> tempo em que ocorreu uma inconsistência na estrutura
----------	---

4.11.2.4 InsertPop()

```
template<typename T >
int Retroactivity::NonObliviousStack< T >::InsertPop (
    int t )
```

Adiciona a realização da operação Pop no tempo t => Insert(t, Pop())

Parameters

<i>t</i>	-> tempo de realização da operação Pop()
----------	--

Returns

-> a próxima operação Pop inconsistente

4.11.2.5 InsertPush()

```
template<typename T >
int Retroactivity::NonObliviousStack< T >::InsertPush (
    int t,
    T x )
```

Insere o objeto x no tempo t na pilha => Insert(t, Push(x))

Parameters

<i>t</i>	-> tempo de inserção do objeto
<i>x</i>	-> objeto a ser inserido no topo da pilha no tempo t

Returns

-> a próxima operação Pop inconsistente

4.11.2.6 showtpop()

```
template<typename T >
void Retroactivity::NonObliviousStack< T >::showtpop ( )
```

Função auxiliar para mostrar os elementos do conjunto tpop

Returns

-> os elementos do conjunto tpop no formato "tempoDelecao, (elementoDeletado)"

4.11.2.7 showtpush()

```
template<typename T >
void Retroactivity::NonObliviousStack< T >::showtpush ( )
```

Função auxiliar para mostrar os elementos do conjunto tpush

Returns

-> os elementos do conjunto tpush no formato "tempoInsercao, (elemento, tempoRemocao)"

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

- /home/junior/Dropbox/ArquivosMestrado/Master/src/Stack/Stack.hpp
- /home/junior/Dropbox/ArquivosMestrado/Master/src/Stack/Stack.cpp

4.12 Retroactivity::FullPriorityQueue< T >::Operation< T > Class Template Reference

Public Member Functions

- **Operation** (int _t, int _op, T _data)
- bool **operator**< (Operation o) const

Public Attributes

- int **t**
- int **op**
- T **data**

4.12.1 Detailed Description

```
template<typename T>
template<typename T>
class Retroactivity::FullPriorityQueue< T >::Operation< T >
```

Classe para armazenamento das operações de um bloco

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

- /home/junior/Dropbox/ArquivosMestrado/Master/src/Priority_Queue/Priority_queue.cpp

4.13 Brute::PartialPriorityQueue< T > Class Template Reference

Public Member Functions

- void **insertPush** (int t, T data)
- void **insertPop** (int t)
- void **removePush** (int t)
- void **removePop** (int t)
- T **getPeak** ()
- bool **empty** ()

Public Attributes

- `std::multiset< pair< int, pair< int, T > > > all`

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

- `/home/junior/Dropbox/ArquivosMestrado/Master/src/Priority_Queue/Priority_queue.hpp`

4.14 Retroactivity::PartialPriorityQueue< T > Class Template Reference

```
#include <Priority_queue.hpp>
```

Public Member Functions

- [PartialPriorityQueue](#) ()
- `vector< pair< int, T > > insertPush` (int t, T data)
- `vector< pair< int, T > > insertPop` (int t)
- `void removePush` (int t)
- `void removePop` (int t)
- `bool empty` ()
- `T getPeak` ()

4.14.1 Detailed Description

```
template<typename T>  
class Retroactivity::PartialPriorityQueue< T >
```

Fila de prioridade parcialmente retroativa em tempo $O(\log(m))$ por operação

4.14.2 Constructor & Destructor Documentation

4.14.2.1 PartialPriorityQueue()

```
template<typename T >  
Retroactivity::PartialPriorityQueue< T >::PartialPriorityQueue ( ) [inline]
```

Construtor padrão para fila de prioridade parcialmente retroativa

4.14.3 Member Function Documentation

4.14.3.1 empty()

```
template<typename T >
bool Retroactivity::PartialPriorityQueue< T >::empty ( )
```

Retorna se a fila está vazia no tempo presente => Empty()

Returns

-> 'true' se a fila estiver vazia

4.14.3.2 getPeak()

```
template<typename T >
T Retroactivity::PartialPriorityQueue< T >::getPeak ( )
```

Retorna o menor elemento da fila de prioridade no tempo atual => GetPeak()

Returns

-> o menor elemento da fila de prioridade após a execução de todas as operações na estrutura.

Precondition

-> a estrutura deve conter pelo menos um elemento

4.14.3.3 insertPop()

```
template<typename T >
vector< pair< int, T > > Retroactivity::PartialPriorityQueue< T >::insertPop (
    int t )
```

Insere uma operação de Pop() no tempo t => Insert(t, Pop())

Parameters

<i>t</i>	-> tempo em que a operação Pop() foi realizada
----------	--

Returns

-> vetor de operações realizadas nas filas qnow e nqnow com a execução da função

Precondition

->

4.14.3.4 insertPush()

```
template<typename T >
vector< pair< int, T > > Retroactivity::PartialPriorityQueue< T >::insertPush (
    int t,
    T data )
```

Insere uma operação de Push(data) no tempo t => Insert(t, Push(data))

Parameters

<i>t</i>	-> tempo em que a operação Push(data) foi realizada
----------	---

Returns

vetor de operações realizadas nas filas qnow e nqnow

4.14.3.5 removePop()

```
template<typename T >
void Retroactivity::PartialPriorityQueue< T >::removePop (
    int t )
```

Remove uma operação de Pop() no tempo t => Delete(t, Pop())

Parameters

<i>t</i>	-> tempo em que a operação Pop() foi realizada
----------	--

Returns

-> vetor de operações realizadas nas filas qnow e nqnow

Precondition

-> deve existir uma operação Delete(t, Pop()) na estrutura

4.14.3.6 removePush()

```
template<typename T >
void Retroactivity::PartialPriorityQueue< T >::removePush (
    int t )
```

Remove uma operação de Push(data) no tempo t => Delete(t, Push(data))

Parameters

<code>t</code>	-> tempo em que a operação Push(data) foi realizada
----------------	---

Returns

-> vetor de operações realizadas nas filas qnow e nqnow

Precondition

-> deve existir uma operação Insert(t, Push(data)) na estrutura

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

- /home/junior/Dropbox/ArquivosMestrado/Master/src/Priority_Queue/Priority_queue.hpp
- /home/junior/Dropbox/ArquivosMestrado/Master/src/Priority_Queue/Priority_queue.cpp

4.15 Brute::PartialQueue< T > Class Template Reference

Public Member Functions

- void **InsertEnqueue** (int t, const T &x)
- void **InsertDequeue** (int t)
- void **DeleteEnqueue** (int t)
- void **DeleteDequeue** (int t)
- T **front** ()

Public Attributes

- map< int, T > **q**

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

- /home/junior/Dropbox/ArquivosMestrado/Master/src/Queue/Queue.hpp

4.16 Retroactivity::PartialQueue< T > Class Template Reference

```
#include <Queue.hpp>
```

Public Member Functions

- void **InsertEnqueue** (int t, const T &x)
- void **InsertDequeue** (int t)
- void **DeleteEnqueue** (int t)
- void **DeleteDequeue** (int t)
- T **front** ()
- T **back** ()

4.16.1 Detailed Description

```
template<typename T>
class Retroactivity::PartialQueue< T >
```

Fila parcialmente retroativa

4.16.2 Member Function Documentation

4.16.2.1 back()

```
template<typename T >
T Retroactivity::PartialQueue< T >::back ( )
```

Retorna o último elemento a ser removido da fila considerando a versão mais atual da estrutura => Back()

Returns

-> O último elemento a ser removido da fila

Precondition

-> a fila não pode estar vazia

4.16.2.2 DeleteDequeue()

```
template<typename T >
void Retroactivity::PartialQueue< T >::DeleteDequeue (
    int t )
```

Remove a operação de Dequeue realizada no tempo t => Delete(t, Dequeue())

Parameters

<i>t</i>	-> tempo em que uma operação Dequeue foi realizada
----------	--

Precondition

-> precisa que uma operação Dequeue no tempo t tenha sido realizada

4.16.2.3 DeleteEnqueue()

```
template<typename T >
void Retroactivity::PartialQueue< T >::DeleteEnqueue (
    int t )
```

Remove a operação de Enqueue realizada no tempo $t \Rightarrow \text{Delete}(t, \text{Enqueue}(x))$

Parameters

t	-> tempo em que uma operação Enqueue foi realizada
-----	--

Precondition

-> precisa que uma operação Enqueue no tempo t tenha sido realizada

4.16.2.4 front()

```
template<typename T >
T Retroactivity::PartialQueue< T >::front ( )
```

Retorna o proximo elemento a ser removido da fila considerando a versão mais atual da estrutura $\Rightarrow \text{Front}()$

Returns

-> O proximo elemento a ser removido da fila

Precondition

-> a fila não pode estar vazia

4.16.2.5 InsertDequeue()

```
template<typename T >
void Retroactivity::PartialQueue< T >::InsertDequeue (
    int t )
```

Insera uma operação Dequeue() no tempo $t \Rightarrow \text{Insert}(t, \text{Dequeue}())$

Recebe um inteiro t (tempo de inserção da operação) em que a operação Dequeue será inserida

Parameters

t	-> tempo de inserção da operação Dequeue
-----	--

4.16.2.6 InsertEnqueue()

```
template<typename T >
void Retroactivity::PartialQueue< T >::InsertEnqueue (
    int t,
    const T & x )
```

Insere uma operação Enqueue(x) no tempo t => Insert(t, Enqueue(x))

Recebe um inteiro t (tempo de inserção da operação) em que a operação Enqueue(x) será inserida

Parameters

<i>t</i>	-> tempo de inserção da operação Enqueue(x)
<i>x</i>	-> objeto a ser inserido na operação

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

- /home/junior/Dropbox/ArquivosMestrado/Master/src/Queue/Queue.hpp
- /home/junior/Dropbox/ArquivosMestrado/Master/src/Queue/Queue.cpp

4.17 Brute::PartialStack< T > Class Template Reference

Public Member Functions

- void **InsertPush** (int t, const T &x)
- void **InsertPop** (int t)
- void **DeletePush** (int t)
- void **DeletePop** (int t)
- T **peak** ()

Public Attributes

- set< pair< int, T > > **v**
- vector< T > **aux**

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

- /home/junior/Dropbox/ArquivosMestrado/Master/src/Stack/Stack.hpp

4.18 Retroactivity::PartialStack< T > Class Template Reference

```
#include <Stack.hpp>
```

Public Member Functions

- void [InsertPush](#) (int t, const T &x)
- void [InsertPop](#) (int t)
- void [DeletePush](#) (int t)
- void [DeletePop](#) (int t)
- T [peak](#) ()
- int [getSize](#) ()

4.18.1 Detailed Description

```
template<typename T>
class Retroactivity::PartialStack< T >
```

Pilha parcialmente retroativa

4.18.2 Member Function Documentation

4.18.2.1 DeletePop()

```
template<typename T >
void Retroactivity::PartialStack< T >::DeletePop (
    int t )
```

Remove a operação Pop realizada no tempo t => Delete(t, Pop())

Parameters

<i>t</i>	-> tempo em que uma operação Pop foi realizada
----------	--

Precondition

-> deve existir uma operação Pop no tempo t

4.18.2.2 DeletePush()

```
template<typename T >
void Retroactivity::PartialStack< T >::DeletePush (
    int t )
```

Remove a operação Push realizada no tempo t => Delete(t, Push(x))

Parameters

<i>t</i>	-> tempo em que uma operação Push foi realizada
----------	---

Precondition

-> deve existir uma operação Push no tempo *t*

4.18.2.3 `getSize()`

```
template<typename T >  
int Retroactivity::PartialStack< T >::getSize ( )
```

Obtem o tamanho da pilha no tempo atual => `getSize()`

Returns

-> retorna o número de elementos na pilha no tempo atual

4.18.2.4 `InsertPop()`

```
template<typename T >  
void Retroactivity::PartialStack< T >::InsertPop (   
    int t )
```

Adiciona a realização da operação Pop no tempo *t* => `Insert(t, Pop())`

Parameters

<i>t</i>	-> tempo de realização da operação Pop()
----------	--

4.18.2.5 `InsertPush()`

```
template<typename T >  
void Retroactivity::PartialStack< T >::InsertPush (   
    int t,   
    const T & x )
```

Insere o objeto *x* no tempo *t* na pilha => `Insert(t, Push(x))`

Parameters

<i>t</i>	-> tempo de inserção do objeto
<i>x</i>	-> objeto a ser inserido no topo da pilha no tempo <i>t</i>

4.18.2.6 peak()

```
template<typename T >
T Retroactivity::PartialStack< T >::peak ( )
```

Retorna o topo da pilha no tempo atual => Peak()

Returns

-> retorna objeto no topo da pilha no tempo atual

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

- /home/junior/Dropbox/ArquivosMestrado/Master/src/Stack/Stack.hpp
- /home/junior/Dropbox/ArquivosMestrado/Master/src/Stack/Stack.cpp

4.19 Retroactivity::PolylogarithmPriorityQueue< T > Class Template Reference

```
#include <Priority_queue.hpp>
```

Classes

- class [Node](#)
- class [QueryNode](#)

Public Member Functions

- [PolylogarithmPriorityQueue](#) (int _m)
- int [getQueryNodeSize](#) ([QueryNode](#) q)
- int [getQtdGE](#) ([QueryNode](#) q, T key)
- T [getMinimumKey](#) ([QueryNode](#) f)
- void [addPush](#) (int no, int l, int r, int i, T data)
- void [addPop](#) (int no, int l, int r, int i)
- void [getNodes](#) (int no, int l, int r, int i, int j, vector< int > &s)
- T [getSplitKey](#) (T A, vector< ii > all)
- pair< [QueryNode](#), [QueryNode](#) > [getSplitedTrees](#) (int x, vector< ii > all)
- pair< [QueryNode](#), [QueryNode](#) > [Fuse](#) (pair< [QueryNode](#), [QueryNode](#) > q1, pair< [QueryNode](#), [QueryNode](#) > q2)
- pair< [QueryNode](#), [QueryNode](#) > [query](#) (int no, int l, int r, int i, int j)
- void [insertPush](#) (int t, T data)
- void [insertPop](#) (int t)
- void [removePush](#) (int t)
- void [removePop](#) (int t)
- T [getPeak](#) (int t)
- pair< [QueryNode](#), [QueryNode](#) > [getView](#) (int t)

4.19.1 Detailed Description

```
template<typename T>
class Retroactivity::PolylogarithmPriorityQueue< T >
```

Fila de prioridade totalmente retroativa em tempo polilogaritmico

4.19.2 Constructor & Destructor Documentation

4.19.2.1 PolylogarithmPriorityQueue()

```
template<typename T >
Retroactivity::PolylogarithmPriorityQueue< T >::PolylogarithmPriorityQueue (
    int _m )
```

Função construtora da Fila de prioridade polilogarítmica.

4.19.3 Member Function Documentation

4.19.3.1 addPop()

```
template<typename T >
void Retroactivity::PolylogarithmPriorityQueue< T >::addPop (
    int no,
    int l,
    int r,
    int i )
```

Função que adiciona a operação pop no tempo i => Insert(i, Pop())

Insere na checkpoint tree uma deleção no tempo i

Parameters

<i>no</i>	-> indice do nó atual da recursao para inserção do elemento
<i>l</i>	-> limite inferior do intervalo contido por no
<i>r</i>	-> limite superior do intervalo contido por no
<i>i</i>	-> tempo em que a operação Pop será inserida na estrutura

Precondition

-> não deve existir nenhuma outra operação no tempo i

4.19.3.2 addPush()

```
template<typename T >
void Retroactivity::PolylogarithmPriorityQueue< T >::addPush (
    int no,
    int l,
    int r,
    int i,
    T data )
```

Função que adiciona um push no tempo i com o elemento data => Insert(i, Push(data))

Insere na checkpoint tree o elemento data no tempo i

Parameters

<i>no</i>	-> indice do nó atual da recursao para inserção do elemento
<i>l</i>	-> limite inferior do intervalo contido por no
<i>r</i>	-> limite superior do intervalo contido por no
<i>i</i>	-> tempo em que o elemento data será inserido na estrutura
<i>data</i>	-> elemento a ser inserido no tempo i

Precondition

-> não deve existir nenhuma outra operação no tempo i

4.19.3.3 Fuse()

```
template<typename T >
pair< typename PolylogarithmPriorityQueue< T >::QueryNode, typename PolylogarithmPriorityQueue< T >::QueryNode > Retroactivity::PolylogarithmPriorityQueue< T >::Fuse (
    pair< QueryNode, QueryNode > q1,
    pair< QueryNode, QueryNode > q2 )
```

Função que une dois conjuntos q1(qnow, qdel) e q2(qnow, qdel) de modo que a união dos conjuntos representem uma fila de prioridade retroativa que abrange o intervalo que q1 e q2 abrangiam.

Parameters

<i>q1</i>	-> (qnow, qdel) representando o intervalo [l1, r1]
<i>q2</i>	-> (qnow, qdel) representando o intervalo [l2, r2]

Returns

-> (qnow, qdel) representando a união de q1 e q2 com o intervalo [l1, r2]

Precondition

-> q1 e q2 devem ser intervalos disjuntos contínuos (r1 < l2)

4.19.3.4 getMinimumKey()

```
template<typename T >
T Retroactivity::PolylogarithmPriorityQueue< T >::getMinimumKey (
    QueryNode f )
```

Função para obtenção do menor elemento dentre todos os conjuntos em f

Obtem o valor do menor elemento nas árvore binárias balanceadas em f (em qnow)

Parameters

<i>f</i>	-> conjunto de elementos na qual deseja-se saber o menor elemento
----------	---

Precondition

-> o tipo abstrato T deve conter uma ordenação consistente.

Returns

-> um tipo abstrato T, contendo o menor elemento em todos os qnow's no conjunto f

4.19.3.5 getNodes()

```
template<typename T >
void Retroactivity::PolylogarithmPriorityQueue< T >::getNodes (
    int no,
    int l,
    int r,
    int i,
    int j,
    vector< int > & s )
```

Função que obtém os índices dos nós contidos em um intervalo [i, j]

Parameters

<i>no</i>	-> índice do nó atual da recursao para inserção do elemento
<i>l</i>	-> limite inferior do intervalo contido por no
<i>r</i>	-> limite superior do intervalo contido por no
<i>i</i>	-> limite inferior da consulta realizada
<i>j</i>	-> limite superior da consulta realizada
<i>s</i>	-> vetor com os índices retornados pela função

4.19.3.6 getPeak()

```
template<typename T >
T Retroactivity::PolylogarithmPriorityQueue< T >::getPeak (
    int t )
```

Retorna o menor elemento da fila de prioridade no tempo t => GetPeak(t)

Parameters

<i>t</i>	-> tempo em que se deseja consultar o menor elemento da fila
----------	--

Returns

-> o menor elemento da fila de prioridade após a execução de todas as operações até o tempo t

Precondition

-> a estrutura deve conter pelo menos um elemento no tempo t

4.19.3.7 getQtdGE()

```
template<typename T >
int Retroactivity::PolylogarithmPriorityQueue< T >::getQtdGE (
    QueryNode q,
    T key )
```

Função para obtenção do numero de elementos em q maiores que key

Obtem a quantidade de elementos em q maiores que key

Parameters

<i>q</i>	-> conjunto de árvores binárias balanceadas
<i>key</i>	-> elemento na qual deseja-se consultar o número de elementos maiores

Precondition

-> o tipo abstrato T deve conter uma ordenação consistente.

Returns

-> um inteiro, o número de elementos em q maiores que key

4.19.3.8 getQueryNodeSize()

```
template<typename T >
int Retroactivity::PolylogarithmPriorityQueue< T >::getQueryNodeSize (
    QueryNode q )
```

Função para obtenção do numero de elementos em q

Obtem a quantidade de elementos em q

Parameters

<i>q</i>	-> conjunto de árvores binárias balanceadas
----------	---

Precondition

-> o tipo abstrato T deve conter uma ordenação consistente.

Returns

-> um inteiro, o numero de elementos em q

4.19.3.9 getSplitedTrees()

```
template<typename T >
pair< typename PolylogarithmPriorityQueue< T >::QueryNode, typename PolylogarithmPriorityQueue< T >::QueryNode > Retroactivity::PolylogarithmPriorityQueue< T >::getSplitedTrees (
    int x,
    vector< ii > all )
```

Função que retorna as árvores após a sua divisão por um valor x

Parameters

<i>x</i>	-> valor na qual deseja-se dividir os conjuntos.
<i>all</i>	-> conjunto das árvores (qnow, qdel) existentes.

Returns

-> conjunto das árvores após a divisão pelo valor x.

4.19.3.10 getSplitKey()

```
template<typename T >
T Retroactivity::PolylogarithmPriorityQueue< T >::getSplitKey (
```

```

T A,
vector< ii > all )

```

Função que obtém o valor que os conjuntos qnow e qdel devem ser divididos de modo que existam A elementos.

Parameters

<i>A</i>	-> número de elementos desejado
<i>all</i>	-> conjunto das árvores (qnow, qdel) existentes.

Returns

-> o valor em que deve-se dividir as árvores de modo que os menores A elementos estejam separados do restante da árvore

4.19.3.11 getView()

```

template<typename T >
pair< typename PolylogarithmPriorityQueue< T >::QueryNode, typename PolylogarithmPriorityQueue< T >::QueryNode > Retroactivity::PolylogarithmPriorityQueue< T >::getView (
    int t )

```

Retorna os conjuntos qnow e qdel considerando a fila de prioridade retroativa no tempo t

Parameters

<i>t</i>	-> tempo em que se deseja consultar o menor elemento da fila
----------	--

Returns

-> um par contendo as árvores contendo os conjuntos qnow e qdel respectivamente

4.19.3.12 insertPop()

```

template<typename T >
void Retroactivity::PolylogarithmPriorityQueue< T >::insertPop (
    int t )

```

Insere uma operação de Pop() no tempo t => Insert(t, Pop())

Parameters

<i>t</i>	-> tempo em que a operação Pop() foi realizada
----------	--

Precondition

->

4.19.3.13 insertPush()

```
template<typename T >
void Retroactivity::PolylogarithmPriorityQueue< T >::insertPush (
    int t,
    T data )
```

Inserir uma operação de Push(data) no tempo t => Insert(t, Push(data))

Parameters

<i>t</i>	-> tempo em que a operação Push(data) foi realizada
<i>data</i>	-> elemento inserido

4.19.3.14 query()

```
template<typename T >
pair< typename PolylogarithmPriorityQueue< T >::QueryNode, typename PolylogarithmPriorityQueue< T >::QueryNode > Retroactivity::PolylogarithmPriorityQueue< T >::query (
    int no,
    int l,
    int r,
    int i,
    int j )
```

Função que obtém os conjuntos dos nós contidos em um intervalo [i, j]

Parameters

<i>no</i>	-> índice do nó atual da recursão para inserção do elemento
<i>l</i>	-> limite inferior do intervalo contido por no
<i>r</i>	-> limite superior do intervalo contido por no
<i>i</i>	-> limite inferior da consulta realizada
<i>j</i>	-> limite superior da consulta realizada

Returns

s -> os conjuntos (qnow, qdel) que formam o intervalo [i, j]

4.19.3.15 removePop()

```
template<typename T >
void Retroactivity::PolylogarithmPriorityQueue< T >::removePop (
    int t )
```

Remove uma operação de Pop() no tempo t => Delete(t, Pop())

Parameters

<i>t</i>	-> tempo em que a operação Pop() foi realizada
----------	--

Precondition

-> deve existir uma operação Delete(t, Pop()) na estrutura

4.19.3.16 removePush()

```
template<typename T >
void Retroactivity::PolylogarithmPriorityQueue< T >::removePush (
    int t )
```

Remove uma operação de Push(data) no tempo t => Delete(t, Push(data))

Parameters

<i>t</i>	-> tempo em que a operação Push(data) foi realizada
----------	---

Precondition

-> deve existir uma operação Insert(t, Push(data)) na estrutura

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

- /home/junior/Dropbox/ArquivosMestrado/Master/src/Priority_Queue/Priority_queue.hpp
- /home/junior/Dropbox/ArquivosMestrado/Master/src/Priority_Queue/Priority_queue.cpp

4.20 Retroactivity::PolylogarithmPriorityQueue< T >::QueryNode< T > Class Template Reference

Public Member Functions

- void **add** (ii f)
- void **add** (QueryNode f)

Public Attributes

- `vector< ii > t`

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

- `/home/junior/Dropbox/ArquivosMestrado/Master/src/Priority_Queue/Priority_queue.cpp`

Index

- addPop
 - Retroactivity::PolylogarithmPriorityQueue, [38](#)
- addPush
 - Retroactivity::PolylogarithmPriorityQueue, [38](#)
- back
 - Retroactivity::PartialQueue, [32](#)
- Brute::FullPriorityQueue< T >, [10](#)
- Brute::FullQueue< T >, [13](#)
- Brute::FullStack< T >, [16](#)
- Brute::PartialPriorityQueue< T >, [27](#)
- Brute::PartialQueue< T >, [31](#)
- Brute::PartialStack< T >, [34](#)
- DeleteDequeue
 - Retroactivity::FullQueue, [10](#)
 - Retroactivity::NonObliviousQueue, [21](#)
 - Retroactivity::PartialQueue, [32](#)
- DeleteEnqueue
 - Retroactivity::FullQueue, [11](#)
 - Retroactivity::NonObliviousQueue, [21](#)
 - Retroactivity::PartialQueue, [32](#)
- DeletePop
 - Retroactivity::FullStack, [14](#)
 - Retroactivity::NonObliviousPriorityQueue, [18](#)
 - Retroactivity::NonObliviousStack, [24](#)
 - Retroactivity::PartialStack, [35](#)
- DeletePush
 - Retroactivity::FullStack, [14](#)
 - Retroactivity::NonObliviousPriorityQueue, [18](#)
 - Retroactivity::NonObliviousStack, [25](#)
 - Retroactivity::PartialStack, [35](#)
- empty
 - Retroactivity::FullPriorityQueue, [7](#)
 - Retroactivity::PartialPriorityQueue, [28](#)
- fixDequeueOperation
 - Retroactivity::NonObliviousQueue, [21](#)
- fixOperation
 - Retroactivity::NonObliviousPriorityQueue, [18](#)
- fixPopOperation
 - Retroactivity::NonObliviousStack, [25](#)
- Front
 - Retroactivity::NonObliviousQueue, [22](#)
- front
 - Retroactivity::FullQueue, [11](#)
 - Retroactivity::PartialQueue, [33](#)
- Fuse
 - Retroactivity::PolylogarithmPriorityQueue, [39](#)
- getKth
 - Retroactivity::FullQueue, [11](#)
- getMinimumKey
 - Retroactivity::PolylogarithmPriorityQueue, [39](#)
- getNodes
 - Retroactivity::PolylogarithmPriorityQueue, [40](#)
- getPeak
 - Retroactivity::FullPriorityQueue, [8](#)
 - Retroactivity::PartialPriorityQueue, [29](#)
 - Retroactivity::PolylogarithmPriorityQueue, [40](#)
- getQtdGE
 - Retroactivity::PolylogarithmPriorityQueue, [41](#)
- getQueryNodeSize
 - Retroactivity::PolylogarithmPriorityQueue, [41](#)
- getSize
 - Retroactivity::FullStack, [14](#)
 - Retroactivity::PartialStack, [36](#)
- getSplitKey
 - Retroactivity::PolylogarithmPriorityQueue, [42](#)
- getSplitedTrees
 - Retroactivity::PolylogarithmPriorityQueue, [42](#)
- getView
 - Retroactivity::PolylogarithmPriorityQueue, [43](#)
- InsertDequeue
 - Retroactivity::FullQueue, [12](#)
 - Retroactivity::NonObliviousQueue, [22](#)
 - Retroactivity::PartialQueue, [33](#)
- InsertEnqueue
 - Retroactivity::FullQueue, [12](#)
 - Retroactivity::NonObliviousQueue, [23](#)
 - Retroactivity::PartialQueue, [34](#)
- InsertPop
 - Retroactivity::FullStack, [15](#)
 - Retroactivity::NonObliviousPriorityQueue, [19](#)
 - Retroactivity::NonObliviousStack, [26](#)
 - Retroactivity::PartialStack, [36](#)
- insertPop
 - Retroactivity::FullPriorityQueue, [8](#)
 - Retroactivity::PartialPriorityQueue, [29](#)
 - Retroactivity::PolylogarithmPriorityQueue, [43](#)
- InsertPush
 - Retroactivity::FullStack, [15](#)
 - Retroactivity::NonObliviousPriorityQueue, [19](#)
 - Retroactivity::NonObliviousStack, [26](#)
 - Retroactivity::PartialStack, [36](#)
- insertPush
 - Retroactivity::FullPriorityQueue, [8](#)
 - Retroactivity::PartialPriorityQueue, [29](#)
 - Retroactivity::PolylogarithmPriorityQueue, [44](#)

- IntervalTreeSum< T >, 16
- PartialPriorityQueue
 - Retroactivity::PartialPriorityQueue, 28
- Peak
 - Retroactivity::NonObliviousPriorityQueue, 20
- peak
 - Retroactivity::FullStack, 15
 - Retroactivity::PartialStack, 37
- PolylogarithmPriorityQueue
 - Retroactivity::PolylogarithmPriorityQueue, 38
- query
 - Retroactivity::PolylogarithmPriorityQueue, 44
- removePop
 - Retroactivity::FullPriorityQueue, 9
 - Retroactivity::PartialPriorityQueue, 30
 - Retroactivity::PolylogarithmPriorityQueue, 44
- removePush
 - Retroactivity::FullPriorityQueue, 9
 - Retroactivity::PartialPriorityQueue, 30
 - Retroactivity::PolylogarithmPriorityQueue, 45
- Retroactivity, 5
- Retroactivity::FullPriorityQueue
 - empty, 7
 - getPeak, 8
 - insertPop, 8
 - insertPush, 8
 - removePop, 9
 - removePush, 9
- Retroactivity::FullPriorityQueue< T >, 7
- Retroactivity::FullPriorityQueue< T >::Operation< T >, 27
- Retroactivity::FullQueue
 - DeleteDequeue, 10
 - DeleteEnqueue, 11
 - front, 11
 - getKth, 11
 - InsertDequeue, 12
 - InsertEnqueue, 12
- Retroactivity::FullQueue< T >, 10
- Retroactivity::FullStack
 - DeletePop, 14
 - DeletePush, 14
 - getSize, 14
 - InsertPop, 15
 - InsertPush, 15
 - peak, 15
- Retroactivity::FullStack< T >, 13
- Retroactivity::NonObliviousPriorityQueue
 - DeletePop, 18
 - DeletePush, 18
 - fixOperation, 18
 - InsertPop, 19
 - InsertPush, 19
 - Peak, 20
- Retroactivity::NonObliviousPriorityQueue< T >, 17
- Retroactivity::NonObliviousQueue
 - DeleteDequeue, 21
 - DeleteEnqueue, 21
 - fixDequeueOperation, 21
 - Front, 22
 - InsertDequeue, 22
 - InsertEnqueue, 23
 - showTd, 23
 - showTe, 23
- Retroactivity::NonObliviousQueue< T >, 20
- Retroactivity::NonObliviousStack
 - DeletePop, 24
 - DeletePush, 25
 - fixPopOperation, 25
 - InsertPop, 26
 - InsertPush, 26
 - showtpop, 26
 - showtpush, 26
- Retroactivity::NonObliviousStack< T >, 24
- Retroactivity::PartialPriorityQueue
 - empty, 28
 - getPeak, 29
 - insertPop, 29
 - insertPush, 29
 - PartialPriorityQueue, 28
 - removePop, 30
 - removePush, 30
- Retroactivity::PartialPriorityQueue< T >, 28
- Retroactivity::PartialQueue
 - back, 32
 - DeleteDequeue, 32
 - DeleteEnqueue, 32
 - front, 33
 - InsertDequeue, 33
 - InsertEnqueue, 34
- Retroactivity::PartialQueue< T >, 31
- Retroactivity::PartialStack
 - DeletePop, 35
 - DeletePush, 35
 - getSize, 36
 - InsertPop, 36
 - InsertPush, 36
 - peak, 37
- Retroactivity::PartialStack< T >, 34
- Retroactivity::PolylogarithmPriorityQueue
 - addPop, 38
 - addPush, 38
 - Fuse, 39
 - getMinimumKey, 39
 - getNodes, 40
 - getPeak, 40
 - getQtdGE, 41
 - getQueryNodeSize, 41
 - getSplitKey, 42
 - getSplitedTrees, 42
 - getView, 43
 - insertPop, 43
 - insertPush, 44
 - PolylogarithmPriorityQueue, 38

- query, [44](#)
- removePop, [44](#)
- removePush, [45](#)
- Retroactivity::PolylogarithmPriorityQueue< T >, [37](#)
- Retroactivity::PolylogarithmPriorityQueue< T >::←
Node< T >, [17](#)
- Retroactivity::PolylogarithmPriorityQueue< T >::←
QueryNode< T >, [45](#)
- showTd
 - Retroactivity::NonObliviousQueue, [23](#)
- showTe
 - Retroactivity::NonObliviousQueue, [23](#)
- showtpop
 - Retroactivity::NonObliviousStack, [26](#)
- showtpush
 - Retroactivity::NonObliviousStack, [26](#)