

TDA-imagen

Generated by Doxygen 1.9.2

1 Class Index	1
1.1 Class List	1
2 File Index	3
2.1 File List	3
3 Class Documentation	5
3.1 MyListaPila Class Reference	5
3.1.1 Detailed Description	5
3.1.2 Member Function Documentation	5
3.1.2.1 borrar()	5
3.1.2.2 insertar()	6
4 File Documentation	7
4.1 src/ejercicio16.cpp File Reference	7
4.1.1 Detailed Description	7
4.1.2 Function Documentation	8
4.1.2.1 main()	8
4.2 src/ejercicio19.cpp File Reference	8
4.2.1 Detailed Description	8
4.2.2 Function Documentation	9
4.2.2.1 imprimir()	9
4.2.2.2 introduce()	10
4.2.2.3 main()	10
4.2.2.4 transformarpila()	10
4.3 src/ejercicio20.cpp File Reference	10
4.3.1 Detailed Description	11
4.3.2 Function Documentation	11
4.3.2.1 imprimir()	11
4.3.2.2 introduce()	12
4.3.2.3 main()	12
4.3.2.4 rotacion()	12
4.4 src/ejercicio21.cpp File Reference	12
4.4.1 Detailed Description	13
4.4.2 Function Documentation	13
4.4.2.1 esmatrizbiordenada()	13
4.4.2.2 imprimir()	14
4.4.2.3 introduce()	14
4.4.2.4 main()	14
4.5 src/ejercicio24.cpp File Reference	14
4.5.1 Detailed Description	15
4.5.2 Function Documentation	15
4.5.2.1 imprimir()	15

4.5.2.2 introduce()	15
4.5.2.3 main()	16
4.5.2.4 posintercalar()	16

Index	17
--------------	-----------

Chapter 1

Class Index

1.1 Class List

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

MyListaPila	MyPilaLista Una clase con metodos de tipo lista implementada con pilas	5
-----------------------------	--	-------------------

Chapter 2

File Index

2.1 File List

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

src/ejercicio16.cpp		
Ejercicio 16 de la relacion del reto 3	7
src/ejercicio19.cpp		
Ejercicio 19 de la relacion del reto 3	8
src/ejercicio20.cpp		
Ejercicio 20 de la relacion del reto 3	10
src/ejercicio21.cpp		
Ejercicio 20 de la relacion del reto 3	12
src/ejercicio24.cpp		
Ejercicio 20 de la relacion del reto 3	14

Chapter 3

Class Documentation

3.1 MyListaPila Class Reference

MyPilaLista Una clase con metodos de tipo lista implementada con pilas.

Public Member Functions

- void **introduce** ()
Pide los datos al usuario para crear la lista a su antojo.
- void **insertar** (const int posicion, int a_insertar)
Inserta al estilo de las listas pero usando stacks No podemos pasar un objeto iterator ya que las pilas no pueden usarlo.
- void **borrar** (const int posicion)
Borra al estilo de las listas pero usando stacks.
- void **imprimir** ()
Imprime la lista a partir de stacks sin destruirla en el proceso.

3.1.1 Detailed Description

MyPilaLista Una clase con metodos de tipo lista implementada con pilas.

3.1.2 Member Function Documentation

3.1.2.1 borrar()

```
void MyListaPila::borrar (  
    const int posicion )    [inline]
```

Borra al estilo de las listas pero usando stacks.

Parameters

<i>posicion</i>	: int Posicion del elemento a borrar
-----------------	--------------------------------------

3.1.2.2 insertar()

```
void MyListaPila::insertar (
    const int posicion,
    int a_insertar ) [inline]
```

Inserta al estilo de las listas pero usando stacks No podemos pasar un objeto iterator ya que las pilas no pueden usarlo.

Parameters

<i>posicion</i>	: int Posicion a insertar
<i>a_insertar</i>	: int Dato entero a insertar

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

- [src/ejercicio16.cpp](#)

Chapter 4

File Documentation

4.1 src/ejercicio16.cpp File Reference

Ejercicio 16 de la relacion del reto 3.

```
#include <iostream>
#include <stack>
```

Classes

- class [MyListaPila](#)

MyPilaLista Una clase con metodos de tipo lista implementada con pilas.

Functions

- void **enunciado** ()

El enunciado del problema a resolver, para tenerlo a la vista sin tener que abrir el documento.

- int [main](#) ()

Programa principal donde realizar las pruebas y probar la clase.

4.1.1 Detailed Description

Ejercicio 16 de la relacion del reto 3.

Author

López Ramírez, Yeray

Date

21 de Noviembre de 2021

4.1.2 Function Documentation

4.1.2.1 main()

```
int main ( )
```

Programa principal donde realizar las pruebas y probar la clase.

Returns

0 si todo funciona bien, otro numero en caso de error

4.2 src/ejercicio19.cpp File Reference

Ejercicio 19 de la relacion del reto 3.

```
#include <cstdlib>
#include <iostream>
#include <stack>
```

Functions

- void **enunciado** ()
El enunciado del problema a resolver, para tenerlo a la vista sin tener que abrir el documento.
- void **introduce** (stack< int > &p)
Pide los datos al usuario para crear la lista a su antojo.
- void **transformarpila** (stack< int > &p)
Elimina los elementos repetidos de la pila pasada como parametro.
- void **imprimir** (const stack< int > &p)
Imprime la pila sin destruir la original.
- int **main** ()
Programa principal donde realizar las pruebas y probar los métodos.

4.2.1 Detailed Description

Ejercicio 19 de la relacion del reto 3.

Author

López Ramírez, Yeray

Date

22 de Noviembre de 2021

4.2.2 Function Documentation

4.2.2.1 imprimir()

```
void imprimir (
    const stack< int > & p )
```

Imprime la pila sin destruir la original.

Parameters

<i>p</i>	: stack La pila a imprimir
----------	----------------------------

4.2.2.2 introduce()

```
void introduce (
    stack< int > & p )
```

Pide los datos al usuario para crear la lista a su antojo.

Parameters

<i>p</i>	: stack La pila a crear/inicializar por el usuario
----------	--

4.2.2.3 main()

```
int main ( )
```

Programa principal donde realizar las pruebas y probar los métodos.

Returns

0 si todo funciona bien, otro numero en caso de error

4.2.2.4 transformarpila()

```
void transformarpila (
    stack< int > & p )
```

Elimina los elementos repetidos de la pila pasada como parametro.

Parameters

<i>p</i>	: stack Pila a limpiar de elementos repetidos
----------	---

4.3 src/ejercicio20.cpp File Reference

Ejercicio 20 de la relacion del reto 3.

```
#include <cstdlib>
#include <iostream>
#include <queue>
```

Functions

- void **enunciado** ()
El enunciado del problema a resolver, para tenerlo a la vista sin tener que abrir el documento.
- void **introduce** (queue< int > &C)
Pide los datos al usuario para crear la cola a su antojo.
- void **rotacion** (queue< int > &C)
Busca un elemento PAR en la cola y mueve todos los elementos a su izquierda al final(a la derecha)
- void **imprimir** (const queue< int > &C)
Imprime la cola sin destruir la original.
- int **main** ()
Programa principal donde realizar las pruebas y probar los métodos.

4.3.1 Detailed Description

Ejercicio 20 de la relacion del reto 3.

Author

López Ramírez, Yeray

Date

23 de Noviembre de 2021

4.3.2 Function Documentation

4.3.2.1 imprimir()

```
void imprimir (
    const queue< int > & C )
```

Imprime la cola sin destruir la original.

Parameters

<i>C</i>	: queue La cola a imprimir
----------	----------------------------

4.3.2.2 introduce()

```
void introduce (
    queue< int > & C )
```

Pide los datos al usuario para crear la cola a su antojo.

Parameters

<i>C</i>	: queue La cola a crear/inicializar por el usuario
----------	--

4.3.2.3 main()

```
int main ( )
```

Programa principal donde realizar las pruebas y probar los métodos.

Returns

0 si todo funciona bien, otro numero en caso de error

4.3.2.4 rotacion()

```
void rotacion (
    queue< int > & C )
```

Busca un elemento PAR en la cola y mueve todos los elementos a su izquierda al final(a la derecha)

Parameters

<i>C</i>	: queue La cola a rotar
----------	-------------------------

4.4 src/ejercicio21.cpp File Reference

Ejercicio 20 de la relacion del reto 3.

```
#include <cstdlib>
#include <iostream>
#include <vector>
```


Functions

- void **enunciado** ()
El enunciado del problema a resolver, para tenerlo a la vista sin tener que abrir el documento.
- void **introduce** (vector< vector< int > > &M)
Pide los datos al usuario para crear la matriz a su antojo.
- bool **esmatrizbiordenada** (const vector< vector< int > > &M)
Comprueba si la matriz pasada es bi-ordenada Una matriz es bi-ordenada si sus elementos están ordenados crecientemente por filas y columnas.
- void **imprimir** (const vector< vector< int > > &M)
Imprime la matriz.
- int **main** ()
Programa principal donde realizar las pruebas y probar los métodos.

4.4.1 Detailed Description

Ejercicio 20 de la relacion del reto 3.

Author

López Ramírez, Yeray

Date

23 de Noviembre de 2021

4.4.2 Function Documentation

4.4.2.1 esmatrizbiordenada()

```
bool esmatrizbiordenada (
    const vector< vector< int > > & M )
```

Comprueba si la matriz pasada es bi-ordenada Una matriz es bi-ordenada si sus elementos están ordenados crecientemente por filas y columnas.

Parameters

<i>M</i>	: vector<vector<int>> La matriz a comprobar
----------	---

Returns

true si es bi-ordenada, false si no lo es

4.4.2.2 imprimir()

```
void imprimir (
    const vector< vector< int > > & M )
```

Imprime la matriz.

Parameters

<i>M</i>	: vector<vector<int>> La matriz a imprimir
----------	--

4.4.2.3 introduce()

```
void introduce (
    vector< vector< int > > & M )
```

Pide los datos al usuario para crear la matriz a su antojo.

Parameters

<i>M</i>	: queue La matriz a crear/inicializar por el usuario
----------	--

4.4.2.4 main()

```
int main ( )
```

Programa principal donde realizar las pruebas y probar los métodos.

Returns

0 si todo funciona bien, otro numero en caso de error

4.5 src/ejercicio24.cpp File Reference

Ejercicio 20 de la relacion del reto 3.

```
#include <cstdlib>
#include <iostream>
#include <list>
#include <assert.h>
```

Functions

- void **enunciado** ()
El enunciado del problema a resolver, para tenerlo a la vista sin tener que abrir el documento.
- void **introduce** (list< int > &L1, list< int > &L2)
Pide los datos al usuario para crear las listas a su antojo.
- void **posintercalar** (list< int > &L1, list< int > &L2)
Intercala las listas segun el enunciado.
- void **imprimir** (const list< int > &L1, const list< int > &L2)
Imprime las listas sin destruir las originales.
- int **main** ()
Programa principal donde realizar las pruebas y probar los métodos.

4.5.1 Detailed Description

Ejercicio 20 de la relacion del reto 3.

Author

López Ramírez, Yeray

Date

23 de Noviembre de 2021

4.5.2 Function Documentation

4.5.2.1 imprimir()

```
void imprimir (  
    const list< int > & L1,  
    const list< int > & L2 )
```

Imprime las listas sin destruir las originales.

Parameters

<i>L1</i>	: list
<i>L2</i>	: list

4.5.2.2 introduce()

```
void introduce (
```

```
list< int > & L1,  
list< int > & L2 )
```

Pide los datos al usuario para crear las listas a su antojo.

Parameters

<i>L1</i>	: list
<i>L2</i>	: lsit

4.5.2.3 main()

```
int main ( )
```

Programa principal donde realizar las pruebas y probar los métodos.

Returns

0 si todo funciona bien, otro numero en caso de error

4.5.2.4 posintercalar()

```
void posintercalar (  
    list< int > & L1,  
    list< int > & L2 )
```

Intercala las listas segun el enunciado.

Parameters

<i>L1</i>	: list Lista a vaciar en el intercalado
<i>L2</i>	: lsit Lista a rellenar intercaladamente con L1

Index

- borrar
 - MyListaPila, 5
- ejercicio16.cpp
 - main, 8
- ejercicio19.cpp
 - imprimir, 9
 - introduce, 10
 - main, 10
 - transformarpila, 10
- ejercicio20.cpp
 - imprimir, 11
 - introduce, 11
 - main, 12
 - rotacion, 12
- ejercicio21.cpp
 - esmatrizbiordenada, 13
 - imprimir, 13
 - introduce, 14
 - main, 14
- ejercicio24.cpp
 - imprimir, 15
 - introduce, 15
 - main, 16
 - posintercalar, 16
- esmatrizbiordenada
 - ejercicio21.cpp, 13
- imprimir
 - ejercicio19.cpp, 9
 - ejercicio20.cpp, 11
 - ejercicio21.cpp, 13
 - ejercicio24.cpp, 15
- insertar
 - MyListaPila, 6
- introduce
 - ejercicio19.cpp, 10
 - ejercicio20.cpp, 11
 - ejercicio21.cpp, 14
 - ejercicio24.cpp, 15
- main
 - ejercicio16.cpp, 8
 - ejercicio19.cpp, 10
 - ejercicio20.cpp, 12
 - ejercicio21.cpp, 14
 - ejercicio24.cpp, 16
- MyListaPila, 5
 - borrar, 5
 - insertar, 6
- posintercalar
 - ejercicio24.cpp, 16
- rotacion
 - ejercicio20.cpp, 12
- src/ejercicio16.cpp, 7
- src/ejercicio19.cpp, 8
- src/ejercicio20.cpp, 10
- src/ejercicio21.cpp, 12
- src/ejercicio24.cpp, 14
- transformarpila
 - ejercicio19.cpp, 10