Primeiro Trabalho - INE5408

Generated by Doxygen 1.8.13

Contents

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Namespace Documentation

4.1 structures Namespace Reference

Classes

- class ElementToSee
- class LinkedQueue

  Fila Encadeada: FIFO: First-In & First-Out.
- class LinkedStack

  Classe de Pilha Encadeada.

4.1.1 Detailed Description

Classe para armazenar a linha e a coluna de um elemento da matriz.

Utilizada para o processo de rotulação de imagens binárias do trabalho, guardará a linha e a coluna da respectiva matriz que está sendo rotulada.

Chapter 5

Class Documentation

5.1 structures::ElementToSee Class Reference

Public Member Functions

- ElementToSee ()
- int getLine () const
- void setLine (int line_)
- int getColumn () const
- void setColumn (int column_)

5.1.1 Detailed Description

Classe para guardar linha e coluna

5.1.2 Constructor & Destructor Documentation

5.1.2.1 ElementToSee()

`structures::ElementToSee::ElementToSee ( )`

Construtor

5.1.3 Member Function Documentation

5.1.3.1 getColumn()

```
int structures::ElementToSee::getColumn ( ) const
```

Getter da coluna

5.1.3.2 getLine()

```
int structures::ElementToSee::getLine ( ) const
```

Getter da linha

5.1.3.3 setColumn()

```
void structures::ElementToSee::setColumn (
            int column_ )
```

Setter da coluna

5.1.3.4 setLine()

```
void structures::ElementToSee::setLine (
            int line_ )
```

Setter da linha

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

- main.cpp

5.2 structures::LinkedQueue< T > Class Template Reference

Fila Encadeada: FIFO: First-In & First-Out.

`#include <linked_queue.h>`

Public Member Functions

- LinkedQueue ()
  Construtor.
- ~LinkedQueue ()
  Destrutor.
- void clear ()
  Limpa Fila.
- void enqueue (const T &data)
  Colocar um elemento no final da fila.
- T dequeue ()
  Tirar o primeiro elemento da fila.
- T & front () const
  Pegar o dado do primeiro elemento.
- T & back () const
  Tirar o dado do ultimo elemento.
- bool empty () const
  Verificar se esta vazia.
- std::size_t size () const
  Retorna o tamano da fila.

5.2.1 Detailed Description

template<typename T>
class structures::LinkedQueue< T >

Fila Encadeada: FIFO: First-In & First-Out.

5.2.2 Constructor & Destructor Documentation

5.2.2.1 LinkedQueue()

```
template<typename T >
structures::LinkedQueue< T >::LinkedQueue ( )
```

Construtor.

5.2.2.2 ~LinkedQueue()

```
template<typename T >
structures::LinkedQueue< T >::~LinkedQueue ( )
```

Destrutor.

5.2.3 Member Function Documentation

5.2.3.1 back()

```
template<typename T >
T & structures::LinkedQueue< T >::back ( ) const
```

Tirar o dado do ultimo elemento.

5.2.3.2 clear()

```
template<typename T >
void structures::LinkedQueue< T >::clear ( )
```

Limpa Fila.

5.2.3.3 dequeue()

```
template<typename T >
T structures::LinkedQueue< T >::dequeue ( )
```

Tirar o primeiro elemento da fila.

5.2.3.4 empty()

```
template<typename T >
bool structures::LinkedQueue< T >::empty ( ) const
```

Verificar se esta vazia.

5.2.3.5 enqueue()

```
template<typename T >
void structures::LinkedQueue< T >::enqueue (
            const T & data )
```

Colocar um elemento no final da fila.

5.2.3.6 front()

```
template<typename T >
T & structures::LinkedQueue< T >::front ( ) const
```

Pegar o dado do primeiro elemento.

5.2.3.7 size()

```
template<typename T >
std::size_t structures::LinkedQueue< T >::size ( ) const
```

Retorna o tamano da fila.

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

- linked_queue.h

5.3 structures::LinkedStack< T > Class Template Reference

Classe de Pilha Encadeada.

`#include <linked_stack.h>`

Public Member Functions

- LinkedStack ()
  Construtor.
- ~LinkedStack ()
  Destrutor.
- void clear ()
  Limpa a pilha.
- void push (const T &data)
  Empilha: Coloca um novo elemento no topo.
- T pop ()
  Desempilha: Tira o último elemento e retorna o dado dele.
- T & top () const
  Retorna o dado do topo.
- bool empty () const
  Testa se a pilha esta vazia.
- std::size_t size () const
  Retorna o tamanho da pilha.

5.3.1 Detailed Description

template<typename T>
class structures::LinkedStack< T >

Classe de Pilha Encadeada.

5.3.2 Constructor & Destructor Documentation

5.3.2.1 LinkedStack()

```
template<typename T >
structures::LinkedStack< T >::LinkedStack ( )
```

Construtor.

5.3.2.2 ~LinkedStack()

```
template<typename T >
structures::LinkedStack< T >::~LinkedStack ( )
```

Destrutor.

5.3.3 Member Function Documentation

5.3.3.1 clear()

```
template<typename T >
void structures::LinkedStack< T >::clear ( )
```

Limpa a pilha.

5.3.3.2 empty()

```
template<typename T >
bool structures::LinkedStack< T >::empty ( ) const
```

Testa se a pilha esta vazia.

5.3.3.3 pop()

```
template<typename T >
T structures::LinkedStack< T >::pop ( )
```

Desempilha: Tira o último elemento e retorna o dado dele.

5.3.3.4 push()

```
template<typename T >
void structures::LinkedStack< T >::push (
            const T & data )
```

Empilha: Coloca um novo elemento no topo.

5.3.3.5 size()

```
template<typename T >
std::size_t structures::LinkedStack< T >::size ( ) const
```

Retorna o tamanho da pilha.

5.3.3.6 top()

```
template<typename T >
T & structures::LinkedStack< T >::top ( ) const
```

Retorna o dado do topo.

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

- linked_stack.h

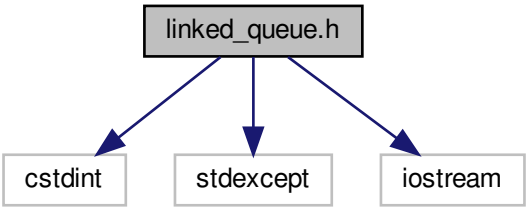
Chapter 6

File Documentation

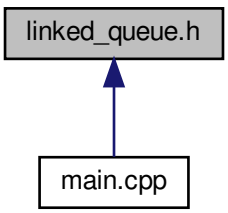
6.1 linked_queue.h File Reference

```
#include <cstdint>
#include <stdexcept>
#include <iostream>
```

Include dependency graph for linked_queue.h:



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



Classes

- class structures::LinkedQueue< T >

  Fila Encadeada: FIFO: First-In & First-Out.

Namespaces

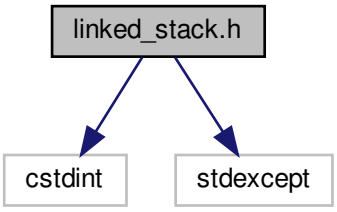
- structures

  Copyright 2018 Maria Eduarda de Melo Hang.

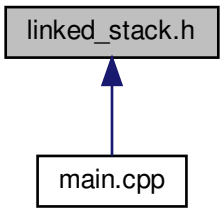
6.2 linked_stack.h File Reference

```
#include <cstdint>
#include <stdexcept>
```

Include dependency graph for linked_stack.h:



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



Classes

- class structures::LinkedStack< T >

  Classe de Pilha Encadeada.

Namespaces

- structures

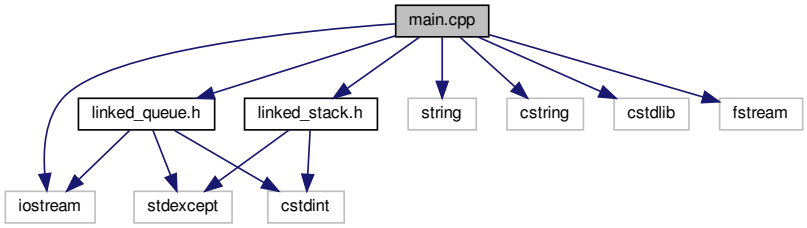
  Copyright 2018 Maria Eduarda de Melo Hang.

6.3 main.cpp File Reference

Primeiro trabalho da disciplina de Estrutura de Dados(INE5408).

```
#include <iostream>
#include <string>
#include <cstring>
#include <cstdlib>
#include <fstream>
#include "linked_stack.h"
#include "linked_queue.h"
```

Include dependency graph for main.cpp:



Classes

- class structures::ElementToSee

Namespaces

- structures

  Copyright 2018 Maria Eduarda de Melo Hang.

Functions

- std::string getXMLText (char xmlfilename[ ])

  Leitura do arquivo xml e armazenamento do respectivo texto.
- int countingWidthTags (std::string XMLtext)

  Contagem das tags "<width>".
- int countingHeightTags (std::string XMLtext)

  Contagem das tags "<height>".
- int countingNameTags (std::string XMLtext)

  Contagem das tags "<name>".

- bool findTheBeginOfXML (std::string XMLtext)

  Verificação da existência da tag "<dataset>".
- bool findTheEndOfXML (std::string XMLtext)

  Verificação da existência da tag "</dataset>".
- int lookUp (int **matrix, int line, int column)

  Verificando o elemento acima do elemento referência.
- int ** createMatrixBase (int lines, int columns)

  Criação de uma matriz nula.
- int lookLeft (int **matrix, int line, int column)

  Verificando o elemento à esquerda do elemento referência.
- std::string cleaningMatrixString (std::string stringMatrix)

  Retirada do caractere de quebra de linha das strings das matrizes.
- int * getMatrixWidths (std::string XMLtext, int countWidths)

  Extração das colunas das matrizes.
- std::string * getXMLMatrix (std::string XMLtext, int counter)

  Extração das matrizes.
- int * getMatrixHeights (std::string XMLtext, int countHeights)

  Extração das linhas das matrizes.
- int lookDown (int **matrix, int line, int column, int lines)

  Verificando o elemento abaixo do elemento referência.
- int lookRight (int **matrix, int line, int column, int columns)

  Verificando o elemento à direita do elemento referência.
- std::string * gettingImageNames (std::string XMLtext, int countNames)

  Extração dos nomes das imagens.
- int ** transformToIntMatrix (std::string stringMatrix, int lines, int columns)

  Transformação das matrizes do arquivo xml em matrizes inteiras.
- void parsingXML (std::string XMLtext, structures::LinkedStack< std::string > stackToTags)

  Verificação dos fechamentos das tags no arquivo xml.
- int searchMatrix (int **matrixXML, int **matrixAuxiliar, int lines, int columns, structures::LinkedQueue< structures::ElementToSee > elementsQueue)

  Rotulamento da Matriz (vizinhança 4).
- int main ()

  Recebe o nome do arquivo como entrada e chama todos os metodos necessarios.

6.3.1 Detailed Description

Primeiro trabalho da disciplina de Estrutura de Dados(INE5408).

Propósito: Verificação do fechamento de "tags" no arquivo xml mediante uma pilha encadeada de strings e rotulação das imagens binárias, presentes no segmento "<data>", com o uso de vizinhança 4, utilizando duas filas encadeadas contendo: matrizes inteiras e ElementToSee (struct definida).

Author

Maria Eduarda de Melo Hang

Date

06 October 2018

6.3.2 Function Documentation

6.3.2.1 cleaningMatrixString()

```
std::string cleaningMatrixString (
            std::string stringMatrix )
```

Retirada do caractere de quebra de linha das strings das matrizes.

Caso o caractere de quebra de linha seja encontrado nessa string, será pego o índice e removido com o uso do método erase até que não reste mais nenhum.

Parameters

stringMatrix	A string que contém uma das matrizes do arquivo xml.

Returns

Uma string sem o caractere de quebra de linha presente.

6.3.2.2 countingHeightTags()

```
int countingHeightTags (
            std::string XMLtext )
```

Contagem das tags "<height>".

Ao encontrar a tag "<height>", o contador será incrementado.

Parameters

XMLtext	A string que contém o texto arquivo xml.

Returns

Um inteiro representando a quantidade de tags "<height>" encontradas.

6.3.2.3 countingNameTags()

```
int countingNameTags (
            std::string XMLtext )
```

Contagem das tags "<name>".

Ao encontrar a tag "<name>", o contador será incrementado.

Parameters

XMLtext	A string que contém o texto arquivo xml.

Returns

Um inteiro representando a quantidade de tags "<name>" encontradas.

6.3.2.4 countingWidthTags()

```
int countingWidthTags (
            std::string XMLtext )
```

Contagem das tags "<width>".

Ao encontrar a tag "<width>", o contador será incrementado.

Parameters

XMLtext	A string que contém o texto arquivo xml.

Returns

Um inteiro representando a quantidade de tags "<width>" encontradas.

6.3.2.5 createMatrixBase()

```
int ** createMatrixBase (
            int lines,
            int columns )
```

Criação de uma matriz nula.

Cria uma matriz nula para ser usada na rotulação, tendo a mesma quantidade de linhas e colunas que matriz do xml.

Parameters

lines	Quantiade de linhas.
columns	Quantidade de colunas.

Returns

Uma matriz de inteiros nula.

6.3.2.6 findTheBeginOfXML()

```
bool findTheBeginOfXML (
            std::string XMLtext )
```

Verificação da existência da tag "<dataset>".

Esse método, com a utilização do método find, procurará pela tag "<dataset>", caso seja encontrada, será atribuído o índice onde se encontra e retornará verdadeiro. Caso contrário, retornará falso.

Parameters

XMLtext	A string que contém o texto arquivo xml.

Returns

Retornará verdadeiro se a tag "<dataset>" foi encontrada no arquivo.

6.3.2.7 findTheEndOfXML()

```
bool findTheEndOfXML (
            std::string XMLtext )
```

Verificação da existência da tag "</dataset>".

Esse método, com a utilização do método find, procurará pela tag "</dataset>", caso seja encontrada, será atribuído o índice onde se encontra e retornará verdadeiro. Caso contrário, retornará falso.

Parameters

XMLtext	A string que contém o texto arquivo xml.

Returns

Retornará verdadeiro se a tag "</dataset>" foi encontrada no arquivo.

6.3.2.8 getMatrixHeights()

```
int * getMatrixHeights (
            std::string XMLtext,
            int countHeights )
```

Extração das linhas das matrizes.

Da mesma forma que o método parsingXML, utiliza-se a função find da biblioteca de strings, nesse caso as strings padrão serão "<height>", para obter o índice do primeiro caractere, e "</height>", para encontrar o índice do último caractere dessa tag. Posteriormente, será pego a substring presente no intervalo [begin, position - begin], sendo extraído apenas o número entre as duas tags. Por fim, essa substring será convertida em inteiro e guardada no array de inteiros heights. O begin será incrementado para buscar os novos índices.

Parameters

XMLtext	A string que contém o texto arquivo xml.
countHeights	Quantidade de valores que devem ser coletados da string(número de matrizes presentes no arquivo).

Returns

Um array de inteiros contendo todos os valores de linhas de cada matriz contida no arquivo xml.

6.3.2.9 getMatrixWidths()

```
int * getMatrixWidths (
            std::string XMLtext,
            int countWidths )
```

Extração das colunas das matrizes.

Da mesma forma que o método parsingXML, utiliza-se a função find da biblioteca de strings, nesse caso as strings padrão serão "<width>", para obter o índice do primeiro caractere, e "</width>", para encontrar o índice do último caractere dessa tag. Posteriormente, será pego a substring presente no intervalo [begin, position - begin], sendo extraído apenas o número entre as duas tags. Por fim, essa substring será convertida em inteiro e guardada no array de inteiros widths. O begin será incrementado para buscar os novos índices.

Parameters

XMLtext	A string que contém o texto arquivo xml.
countWidths	Quantidade de valores que devem ser coletados da string(número de matrizes presentes no arquivo).

Returns

Um array de inteiros contendo todos os valores de colunas de cada matriz contida no arquivo xml.

6.3.2.10 gettingImageNames()

```
std::string * gettingImageNames (
            std::string XMLtext,
            int countNames )
```

Extração dos nomes das imagens.

Da mesma forma que o método parsingXML, utiliza-se a função find da biblioteca de strings, nesse caso as strings padrão serão "<name>", para obter o índice do primeiro caractere, e "</name>", para encontrar o índice do último caractere dessa tag. Posteriormente, será pego a substring presente no intervalo [begin, position - begin], sendo extraído apenas o número entre as duas tags. Por fim, essa substring será guardada no array de strings names. O begin será incrementado para buscar os novos índices.

Parameters

XMLtext	A string que contém o texto arquivo xml.
countNames	Quantidade de nomes que devem ser buscados.

Returns

Um array de strings contendo todos os nomes das imagens.

6.3.2.11 getXMLMatrix()

```
std::string * getXMLMatrix (
            std::string XMLtext,
            int counter )
```

Extração das matrizes.

Da mesma forma que nos métodos de extração de linhas e colunas das matrizes, esse método realizará a busca das matrizes contidas entre as tags com o padrão "<data>" e "</data>", pegando os respectivos índices e extraindo a substring no intervalo [begin, position - begin] (a matriz). O begin será incrementado para buscar os novos índices.

Parameters

XMLtext	A string que contém o texto arquivo xml.
counter	Quantidade de matrizes presentes no arquivo.

Returns

Um array de strings contendo cada matrize do arquivo xml.

6.3.2.12 getXMLText()

```
std::string getXMLText (
            char xmlfilename[] )
```

Leitura do arquivo xml e armazenamento do respectivo texto.

Primeiramente, tenta-se abrir o arquivo xml recebido no terminal, caso não seja possível, o programa sairá retorna 1 (erro). Após essa verificação, o ifstream source lerá caractere por caractere e concatenará com a variável local XMLtext e terminará quando encontrar o EOF.

Parameters

xmlfilename	Nome do arquivo xml que será aberto e lido.

Returns

O texto do arquivo xml em formate de uma string.

6.3.2.13 lookDown()

```
int lookDown (
            int ** matrix,
            int line,
            int column,
            int lines )
```

Verificando o elemento abaixo do elemento referência.

Parameters

matrix	A matriz.
line	A linha do elemento de referência que deseja verificar.
column	A coluna do elemento de referência que deseja verificar.
lines	Quantidade de linhas da matriz para testar o limite do índice.

Returns

Retornará 1 caso o elemento debaixo contenha o valor 1, caso contrário, 0.

6.3.2.14 lookLeft()

```
int lookLeft (
            int ** matrix,
            int line,
            int column )
```

Verificando o elemento à esquerda do elemento referência.

Parameters

matrix	A matriz.
line	A linha do elemento de referência que deseja verificar.
column	A coluna do elemento de referência que deseja verificar.

Returns

Retornará 1 caso o elemento à esquerda contenha o valor 1, caso contrário, 0.

6.3.2.15 lookRight()

```
int lookRight (
            int ** matrix,
            int line,
            int column,
            int columns )
```

Verificando o elemento à direita do elemento referência.

Parameters

matrix	A matriz.
line	A linha do elemento de referência que deseja verificar.
column	A coluna do elemento de referência que deseja verificar.
columns	Quantidade de colunas da matriz para testar o limite do índice.

Returns

Retornará 1 caso o elemento à direita contenha o valor 1, caso contrário, 0.

6.3.2.16 lookUp()

```
int lookUp (
            int ** matrix,
            int line,
            int column )
```

Verificando o elemento acima do elemento referência.

Parameters

matrix	A matriz.
line	A linha do elemento de referência que deseja verificar.
column	A coluna do elemento de referência que deseja verificar.

Returns

Retornará 1 caso o elemento acima contenha o valor 1, caso contrário, 0.

6.3.2.17 main()

```
int main ( )
```

Recebe o nome do arquivo como entrada e chama todos os metodos necessarios.

6.3.2.18 parsingXML()

```
void parsingXML (
            std::string XMLtext,
            structures::LinkedStack< std::string > stackToTags )
```

Verificação dos fechamentos das tags no arquivo xml.

Usando o método find da biblioteca de strings, pegará o índice do primeiro caractere que for igual a "<", depois o outro índice que for igual a ">" e por fim pegará uma substring, presente no intervalo [begin, end - begin + 1] do texto do XML e passará para a string auxiliar. Caso o caractere "/" seja encontrado, ele será retirado com o método erase. Posteriormente, haverá uma verificação no caso da pilha ser vazia, se a condição for verdadeira, essa string auxiliar será automaticamente empilhada. Caso contrário, o método verificará se o topo coincide com a string auxiliar, se for verdade, a pilha realizará um pop, caso contrário um push, colocando a auxiliar como parâmetro. Ao fim, o begin será incrementado para pegar o próximo índice do caractere que conter o "<". Caso uma tag de fechamento foi encontrada e não for igual ao topo, o programa será finalizado e colocará "error" na tela.

Parameters

XMLtext	A string que contém o texto arquivo xml.
stackToTags	Uma pilha para guardar as tags.

6.3.2.19 searchMatrix()

```
int searchMatrix (
            int ** matrixXML,
            int ** matrixAuxiliar,
            int lines,
            int columns,
            structures::LinkedQueue< structures::ElementToSee > elementsQueue )
```

Rotulamento da Matriz (vizinhança 4).

Primeiramente, o método percorrerá a matriz do xml e ao encontrar o valor 1 em um dos elementos, ocorrerá uma verificação para ver se já não foi visitado na matriz de rotulamento (se não foi visitado o valor será 0), caso não tenha sido visitado, será instanciado um elemento da classe ElementToSee para guardar a linha e coluna desse elemento, sendo inserido na fila de rotulamento, e o rótulo será aplicado aplicado a matriz de rotulamento na mesma linha e coluna. Posteriormente, os 4 vizinhos desse elemento serão verificados com os métodos lookUp, lookDown, lookRight e lookLeft e depois se a matriz de rotulamento já não visitou esse vizinho outro elemento da classe ElementToSee será instanciado e inserido na fila de rotulamento e o rótulo atual será atribuído ao vizinho que atendeu às condições mostradas anteriormente. Por fim, o rótulo será incrementado se nenhum outro elemento conexo for encontrado.

Parameters

matrixXML	A matriz de inteiros do arquivo xml.
matrixAuxiliar	A matriz auxiliar de rotulamento.
lines	Quantidade de linhas da matriz.
columns	Quantidade de colunas da matriz.
elementsQueue	Fila para guardar os elementos a serem verificados e seus vizinhos (vizinhança 4).

Returns

O maior rótulo atribuído.

6.3.2.20 transformToIntMatrix()

```
int ** transformToIntMatrix (
            std::string stringMatrix,
            int lines,
            int columns )
```

Transformação das matrizes do arquivo xml em matrizes inteiras.

Para realizar essa transformação, o método pegará caractere por caractere e inserir na matriz de inteiros em cada elemento da respectiva posição.

Parameters

stringMatrix	A string que contém uma matriz do arquivo xml.
lines	Quantidade de linhas da matriz.
columns	Quantidade de colunas da matriz.

Returns

Uma matriz de inteiros contendo os valores (0 ou 1) da matrize do arquivo xml.

Index