TSPLIB

Generated by Doxygen 1.7.6.1

Fri Jan 4 2013 09:57:35

Contents

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Class Documentation

3.1 CommandLineArgumentsInvalid Class Reference

`#include <ValidateConstantsFunctionsAndClasses.hpp>`

Public Member Functions

- CommandLineArgumentsInvalid ()

3.1.1 Detailed Description

Exception class used if the command-line arguments are invalid.

Definition at line 133 of file ValidateConstantsFunctionsAndClasses.hpp.

3.1.2 Constructor & Destructor Documentation

3.1.2.1 CommandLineArgumentsInvalid::CommandLineArgumentsInvalid ( ) [inline]

Constructor for the class CommandLineArgumentsInvalid.

Definition at line 138 of file ValidateConstantsFunctionsAndClasses.hpp.

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

- src/ValidateConstantsFunctionsAndClasses.hpp

3.2 Graph Class Reference

`#include <ValidateGraph.hpp>`

Public Member Functions

- Graph (const std::vector< std::vector< double > > &adjacencyMatrix)
- Graph (const Graph &graph)
- Graph & operator= (const Graph &graph)
- bool getIsUndirected () const
- std::vector< std::vector < double > >::size_type getN () const
- double getEdgeCost (const std::vector< std::vector< double > >::size_type i, const std::vector< double >::size_type j) const

Protected Member Functions

- void setAdjacencyMatrixElement (const std::vector< std::vector< double > >::size_type i, const std::vector< double >::size_type j, const double value)
- double getAdjacencyMatrixElement (const std::vector< std::vector< double > >::size_type i, const std::vector< double >::size_type j) const

Private Attributes

- bool isUndirected
- std::vector< std::vector < double > >::size_type n
- std::vector< std::vector < double > > adjacencyMatrix

Friends

- std::ostream & operator<< (std::ostream &os, Graph &graph)

3.2.1 Detailed Description

Saves one weighted complete graph. This class provides no checks of ranges.

Definition at line 35 of file ValidateGraph.hpp.

3.2.2 Constructor & Destructor Documentation

3.2.2.1 Graph::Graph (const std::vector< std::vector< double > > & *adjacencyMatrix*)

Constructor for the class Graph.

Parameters

adjacency-Matrix	Adjacency matrix of a weighted undirected complete graph. The elements in the matrix are the costs. The matrix can be either only a lower triangular matrix for undirected graphs or it can be a square matrix for the directed graph. In both cases on the main diagonale there must be 0.

Definition at line 17 of file ValidateGraph.cpp.

3.2.2.2 Graph::Graph (const Graph & *graph*)

Copy constructor for the class Graph.

Parameters

graph	Instance to be copied.

Definition at line 48 of file ValidateGraph.cpp.

References getAdjacencyMatrixElement(), getIsUndirected(), and getN().

3.2.3 Member Function Documentation

3.2.3.1 double Graph::getAdjacencyMatrixElement (const std::vector< std::vector< double > >::size_type *i*, const std::vector< double >::size_type *j*) const [inline, protected]

Returns one element in the adjacency matrix. The parameters are not checked.

Parameters

i	Row.
j	Column.

Returns

Value of the element on the i-th row and j-th column.

Definition at line 80 of file ValidateGraph.hpp.

References adjacencyMatrix.

Referenced by getEdgeCost(), Graph(), and operator=().

3.2.3.2 double Graph::getEdgeCost (const std::vector< std::vector< double > >::size_type *i*, const std::vector< double >::size_type *j*) const [inline]

Returns the cost of the edge from i to j. The parameters are not checked.

Parameters

i	Start vertex.
j	End vertex.

Returns

Weight of the edge from i to j.

Definition at line 144 of file ValidateGraph.hpp.

References getAdjacencyMatrixElement(), and isUndirected.

Referenced by operator<<().

3.2.3.3 bool Graph::getIsUndirected ( ) const [inline]

Returns the indicator indicating if the graph is undirected.

Returns

Indicator indicating if the graph is undirected.

- true The Graph is undirected.
- false The Graph is directed.

Definition at line 126 of file ValidateGraph.hpp.

References isUndirected.

Referenced by Graph(), main(), operator<<(), and operator=().

3.2.3.4 std::vector<std::vector<double> >::size_type Graph::getN ( ) const [inline]

Returns the number of vertices in the graph of the instance.

Returns

Number of vertices in the graph of the instance.

Definition at line 134 of file ValidateGraph.hpp.

References n.

Referenced by Graph(), main(), operator<<(), and operator=().

3.2.3.5 Graph & Graph::operator= (const Graph & *graph*)

Implements the operator "=".

Parameters

graph	Right side of the operator.

Returns

Left side of the operator.

Definition at line 79 of file ValidateGraph.cpp.

References getAdjacencyMatrixElement(), getIsUndirected(), and getN().

3.2.3.6 void Graph::setAdjacencyMatrixElement (const std::vector< std::vector< double > >::size_type *i*, const std::vector< double >::size_type *j*, const double *value*) [inline, protected]

Sets one element in the adjacency matrix. The parameters are not checked.

Parameters

i	Row.
j	Column.
value	New value of the element on the i-th row and j-th column.

Definition at line 67 of file ValidateGraph.hpp.

References adjacencyMatrix.

3.2.4 Friends And Related Function Documentation

3.2.4.1 std::ostream& operator<< (std::ostream & *os*, Graph & *graph*) [friend]

Implements the operator "<<".

Parameters

os	Ostream.
graph	Graph.

Returns

Ostream with the information about the graph instance.

Definition at line 117 of file ValidateGraph.cpp.

3.2.5 Member Data Documentation

3.2.5.1 std::vector<std::vector<double> > Graph::adjacencyMatrix [private]

Adjacency matrix of a weighted undirected complete graph. The elements in the matrix are the costs. The elements in the matrix are the costs. The matrix can be either only a lower triangular matrix for undirected graphs or it can be a square matrix for the directed graph.

Definition at line 58 of file ValidateGraph.hpp.

Referenced by getAdjacencyMatrixElement(), and setAdjacencyMatrixElement().

3.2.5.2 bool Graph::isUndirected [private]

Indicator indicating if the graph is undirected.

- true The Graph is undirected.
- false The Graph is directed.

Definition at line 44 of file ValidateGraph.hpp.

Referenced by getEdgeCost(), and getIsUndirected().

3.2.5.3 std::vector<std::vector<double> >::size_type Graph::n [private]

Number of vertices in the graph of the instance.

Definition at line 49 of file ValidateGraph.hpp.

Referenced by getN().

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

- src/ValidateGraph.hpp
- src/ValidateGraph.cpp

3.3 GraphNotValid Class Reference

`#include <ValidateGraph.hpp>`

Public Member Functions

- GraphNotValid ()

3.3.1 Detailed Description

Exception class used if the graph is not valid.

Definition at line 22 of file ValidateGraph.hpp.

3.3.2 Constructor & Destructor Documentation

3.3.2.1 GraphNotValid::GraphNotValid ( ) [inline]

Constructor for the class GraphNotValid.

Definition at line 27 of file ValidateGraph.hpp.

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

- src/ValidateGraph.hpp

3.4 InputFileFormatNotSupported Class Reference

`#include <TransformConstantsClassesAndFunctions.hpp>`

Public Member Functions

- InputFileFormatNotSupported ()

3.4.1 Detailed Description

Exception class used if the input format of the tsp file is not supported.

Definition at line 316 of file TransformConstantsClassesAndFunctions.hpp.

3.4.2 Constructor & Destructor Documentation

3.4.2.1 InputFileFormatNotSupported::InputFileFormatNotSupported ( ) [inline]

Constructor for the class FormatNotSupported.

Definition at line 321 of file TransformConstantsClassesAndFunctions.hpp.

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

- src/TransformConstantsClassesAndFunctions.hpp

3.5 Instance Class Reference

`#include <ValidateInstance.hpp>`

Public Member Functions

- Instance (const std::string &name, const std::string &source, const std::string &description, const std::streamsize doublePrecision, const std::streamsize ignoredDigits, const Graph &graph)

- Instance (const Instance &instance)
- ~Instance ()
- Instance & operator= (const Instance &instance)
- std::string getName () const
- std::string getSource () const
- std::string getDescription () const
- std::streamsize getDoublePrecision () const
- std::streamsize getIgnoredDigits () const
- double getDoubleZero () const
- Graph * getGraph () const

Private Member Functions

- void init (const std::string &name, const std::string &source, const std::string &description, const std::streamsize doublePrecision, const std::streamsize ignoredDigits, const Graph &graph)

Private Attributes

- std::string name
- std::string source
- std::string description
- std::streamsize doublePrecision
- std::streamsize ignoredDigits
- double doubleZero
- Graph * graph

Friends

- std::ostream & operator<< (std::ostream &os, Instance &instance)

3.5.1 Detailed Description

Saves one instance of the orienteering problem with the graph in the form of a weighted complete graph. This class provides no checks of ranges.

Definition at line 29 of file ValidateInstance.hpp.

3.5.2 Constructor & Destructor Documentation

3.5.2.1 Instance::Instance (const std::string & *name*, const std::string & *source*, const std::string & *description*, const std::streamsize *doublePrecision*, const std::streamsize *ignoredDigits*, const Graph & *graph*)

Constructor for the class Instance.

Parameters

name	Name of the instance.
source	Source of the instance.
description	Description of the instance.
doublePrecision	Precision of doubles.
ignoredDigits	Number of ignored digits of double types. (The deviation of double values can be at most 1e-(DoublePrecision - IgnoredDigits).)
graph	The underlying weighted complete graph.

Definition at line 37 of file ValidateInstance.cpp.

3.5.2.2 Instance::Instance (const Instance & *instance*)

Copy constructor for the class Instance.

Parameters

instance	Instance to be copied.

Definition at line 56 of file ValidateInstance.cpp.

References getDescription(), getDoublePrecision(), getGraph(), getIgnoredDigits(), getName(), and getSource().

3.5.2.3 Instance::~Instance ( )

Destructor for the class Instance.

Definition at line 75 of file ValidateInstance.cpp.

3.5.3 Member Function Documentation

3.5.3.1 std::string Instance::getDescription ( ) const [inline]

Returns the description of the instance.

Returns

Description of the instance.

Definition at line 152 of file ValidateInstance.hpp.

References description.

Referenced by Instance(), main(), operator<<(), and operator=().

3.5.3.2 std::streamsize Instance::getDoublePrecision ( ) const [inline]

Returns the precision of doubles.

Returns

Precision of doubles.

Definition at line 160 of file ValidateInstance.hpp.

References doublePrecision.

Referenced by Instance(), operator<<(), and operator=().

3.5.3.3 double Instance::getDoubleZero ( ) const [inline]

Returns the maximum of an absolute value considered as zero.

Returns

Maximum of an absolute value considered as zero.

Definition at line 180 of file ValidateInstance.hpp.

References doubleZero.

Referenced by operator<<().

3.5.3.4 Graph* Instance::getGraph ( ) const [inline]

Returns the underlying weighted undirected complete graph.

Returns

Underlying weighted undirected complete graph.

Definition at line 188 of file ValidateInstance.hpp.

References graph.

Referenced by Instance(), main(), operator<<(), and operator=().

3.5.3.5 std::streamsize Instance::getIgnoredDigits ( ) const [inline]

Returns the number of ignored digits of double types. (The deviation of double values can be at most 1e-(DoublePrecision - IgnoredDigits).)

Returns

Number of ignored digits of double types. (The deviation of double values can be at most 1e-(DoublePrecision - IgnoredDigits).)

Definition at line 172 of file ValidateInstance.hpp.

References ignoredDigits.

Referenced by Instance(), operator<<(), and operator=().

3.5.3.6 std::string Instance::getName ( ) const [inline]

Returns the name of the instance.

Returns

Name of the instance.

Definition at line 136 of file ValidateInstance.hpp.

References name.

Referenced by Instance(), main(), operator<<(), and operator=().

3.5.3.7 std::string Instance::getSource ( ) const [inline]

Returns the source of the instance.

Returns

Source of the instance.

Definition at line 144 of file ValidateInstance.hpp.

References source.

Referenced by Instance(), main(), operator<<(), and operator=().

3.5.3.8 void Instance::init (const std::string & *name,* const std::string & *source,* const std::string & *description,* const std::streamsize *doublePrecision,* const std::streamsize *ignoredDigits,* const Graph & *graph*) [private]

Initializes the class variables

Parameters

name	Name of the instance.
source	Source of the instance.
description	Description of the instance.
doublePrecision	Precision of doubles.
ignoredDigits	Number of ignored digits of double types. (The deviation of double values can be at most 1e-(DoublePrecision - IgnoredDigits).)
graph	The underlying weighted complete graph.

Definition at line 21 of file ValidateInstance.cpp.

3.5.3.9 Instance & Instance::operator= (const Instance & *instance*)

Implements the operator "=".

Parameters

instance	Right side of the operator.

Returns

Left side of the operator.

Definition at line 81 of file ValidateInstance.cpp.

References getDescription(), getDoublePrecision(), getGraph(), getIgnoredDigits(), getName(), and getSource().

3.5.4 Friends And Related Function Documentation

3.5.4.1 std::ostream& operator<< (std::ostream & *os,* Instance & *instance*) [friend]

Implements the operator "<<".

Parameters

os	Ostream.
instance	Instance.

Returns

Ostream with the information about the instance instance.

Definition at line 107 of file ValidateInstance.cpp.

3.5.5 Member Data Documentation

3.5.5.1 std::string Instance::description [private]

Description of the instance.

Definition at line 44 of file ValidateInstance.hpp.

Referenced by getDescription().

3.5.5.2 std::streamsize Instance::doublePrecision [private]

Precision of doubles.

Definition at line 49 of file ValidateInstance.hpp.

Referenced by getDoublePrecision().

3.5.5.3 double Instance::doubleZero [private]

Maximum of an absolute value considered as zero.

Definition at line 60 of file ValidateInstance.hpp.

Referenced by getDoubleZero().

3.5.5.4 Graph* Instance::graph [private]

The underlying weighted complete graph.

Definition at line 65 of file ValidateInstance.hpp.

Referenced by getGraph().

3.5.5.5 std::streamsize Instance::ignoredDigits [private]

Number of ignored digits of double types. (The deviation of double values can be at most 1e-(DoublePrecision - IgnoredDigits).)

Definition at line 55 of file ValidateInstance.hpp.

Referenced by getIgnoredDigits().

3.5.5.6 std::string Instance::name [private]

Name of the instance.

Definition at line 34 of file ValidateInstance.hpp.

Referenced by getName().

3.5.5.7 std::string Instance::source [private]

Source of the instance.

Definition at line 39 of file ValidateInstance.hpp.

Referenced by getSource().

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

- src/ValidateInstance.hpp
- src/ValidateInstance.cpp

3.6 Point Struct Reference

Public Attributes

- double x
- double y

3.6.1 Detailed Description

Saves one point in the two-dimensional plain.

Definition at line 118 of file TransformConstantsClassesAndFunctions.cpp.

3.6.2 Member Data Documentation

3.6.2.1 double Point::x

Definition at line 119 of file TransformConstantsClassesAndFunctions.cpp.

Referenced by readInputFileTSPLIB().

3.6.2.2 double Point::y

Definition at line 120 of file TransformConstantsClassesAndFunctions.cpp.

Referenced by readInputFileTSPLIB().

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

- src/TransformConstantsClassesAndFunctions.cpp

3.7 SAX2ContentHandler Class Reference

```
#include <ValidateSAX2ContentHandler.hpp>
```

Public Member Functions

- SAX2ContentHandler ()
- ~SAX2ContentHandler ()
- bool getFailed () const
- std::string getName () const
- std::string getSource () const
- std::string getDescription () const
- std::streamsize getDoublePrecision () const
- std::streamsize getIgnoredDigits () const

- double getDoubleZero () const
- std::vector< std::vector < double > > getAdjacencyMatrix ()
- void setDocumentLocator (const Locator *const locator)
- void resetDocument ()
- void startDocument ()
- void endDocument ()
- void startPrefixMapping (const XMLCh *const prefix, const XMLCh *const uri)
- void endPrefixMapping (const XMLCh *const prefix)
- void skippedEntity (const XMLCh *const name)
- void startElement (const XMLCh *const uri, const XMLCh *const localname, const XMLCh *const qname, const Attributes &attributes)
- void endElement (const XMLCh *const uri, const XMLCh *const localname, const XMLCh *const qname)
- void characters (const XMLCh *const chars, const XMLSize_t length)
- void ignorableWhitespace (const XMLCh *const chars, const XMLSize_t length)
- void processingInstruction (const XMLCh *const target, const XMLCh *const data)

Private Member Functions

- SAX2ContentHandler (const SAX2ContentHandler &sAX2ContentHandler)
- SAX2ContentHandler & operator= (const SAX2ContentHandler &sAX2Content-Handler)
- void init ()

Private Attributes

- bool failed
- bool isUndirected
- std::string name
- std::string source
- std::string description
- std::streamsize doublePrecision
- std::streamsize ignoredDigits
- double doubleZero
- bool firstVertex
- std::vector< std::vector < double > >::size_type numberOfParsedVertices
- std::set< std::vector< double > ::size_type > edgesDefinedInTheFirstVertex
- std::vector< double > costsOfEdgesDefinedInTheFirstVertex
- std::vector< std::vector < double > >::size_type n
- std::vector< std::vector< bool > > parsedEntries
- std::vector< std::vector < double > > adjacencyMatrix
- std::string buffer
- double cost

3.7.1 Detailed Description

Definition at line 25 of file ValidateSAX2ContentHandler.hpp.

3.7.2 Constructor & Destructor Documentation

3.7.2.1 SAX2ContentHandler::SAX2ContentHandler (const SAX2ContentHandler & *sAX2ContentHandler*) [private]

Not implemented copy constructor for the class SAX2ContentHandler.

Parameters

sAX2-Content-Handler	Instance to be copied.

3.7.2.2 SAX2ContentHandler::SAX2ContentHandler ( )

Constructor for the class SAXContentHandler.

Definition at line 117 of file ValidateSAX2ContentHandler.cpp.

References init().

3.7.2.3 SAX2ContentHandler::~SAX2ContentHandler ( )

Destructor for the class SAXContentHandler.

Definition at line 121 of file ValidateSAX2ContentHandler.cpp.

3.7.3 Member Function Documentation

3.7.3.1 void SAX2ContentHandler::characters (const XMLCh *const *chars*, const XMLSize_t *length*)

Called if character data occurs.

Parameters

chars	Character data.
length	Length of the Character data.

Definition at line 390 of file ValidateSAX2ContentHandler.cpp.

References buffer, stringForm, and trim().

3.7.3.2 void SAX2ContentHandler::endDocument ( )

Called if the document ends.

Definition at line 131 of file ValidateSAX2ContentHandler.cpp.

References adjacencyMatrix, doubleZero, failed, isUndirected, n, and parsedEntries.

3.7.3.3 void SAX2ContentHandler::endElement (const XMLCh ∗const *uri,* const XMLCh ∗const *localname,* const XMLCh ∗const *qname*)

Called if an element ends.

Parameters

uri	URI of the associated namespace for this element.
localname	Local part of the element name.
qname	QName of this element.

Definition at line 226 of file ValidateSAX2ContentHandler.cpp.

References adjacencyMatrix, buffer, cost, costsOfEdgesDefinedInTheFirstVertex, description, doublePrecision, doubleZero, edgesDefinedInTheFirstVertex, failed, firstVertex, ignoredDigits, n, name, numberOfParsedVertices, parsedEntries, source, trim(), unicodeForm, XML_DESCRIPTION, XML_DOUBLE_PRECISION, XML_EDGE, XML_GRAPH, XML_IGNORED_DIGITS, XML_NAME, XML_SOURCE, and XML_VERTEX.

3.7.3.4 void SAX2ContentHandler::endPrefixMapping (const XMLCh ∗const *prefix*)

Receive notification of the end of an namespace prefix mapping.

Parameters

prefix	Namespace prefix used.

Definition at line 186 of file ValidateSAX2ContentHandler.cpp.

3.7.3.5 std::vector<std::vector<double> > SAX2ContentHandler::getAdjacencyMatrix ( ) [inline]

Returns the adjacency matrix of a weighted undirected complete graph. The elements in the matrix are the costs.

Returns

Adjacency matrix of a weighted undirected complete graph. The elements in the matrix are the costs.

Definition at line 220 of file ValidateSAX2ContentHandler.hpp.

Referenced by instanceIn().

3.7.3.6 std::string SAX2ContentHandler::getDescription ( ) const [inline]

Returns the description.

Returns

Description.

Definition at line 182 of file ValidateSAX2ContentHandler.hpp.

Referenced by instanceIn().

3.7.3.7 std::streamsize SAX2ContentHandler::getDoublePrecision ( ) const [inline]

Returns the precision of doubles.

Returns

Precision of doubles.

Definition at line 190 of file ValidateSAX2ContentHandler.hpp.

Referenced by instanceIn().

3.7.3.8 double SAX2ContentHandler::getDoubleZero ( ) const [inline]

Returns the maximum of an absolute value considered as zero.

Returns

Maximum of an absolute value considered as zero.

Definition at line 210 of file ValidateSAX2ContentHandler.hpp.

3.7.3.9 bool SAX2ContentHandler::getFailed ( ) const [inline]

Returns true if the last validation failed

Returns

- true if the last validation failed
- false if the last validation failed

Definition at line 158 of file ValidateSAX2ContentHandler.hpp.

Referenced by instanceIn().

3.7.3.10 std::streamsize SAX2ContentHandler::getIgnoredDigits ( ) const [inline]

Returns the number of ignored digits of double types. (The deviation of double values can be at most 1e-(DoublePrecision - IgnoredDigits).)

Returns

Number of ignored digits of double types. (The deviation of double values can be at most 1e-(DoublePrecision - IgnoredDigits).)

Definition at line 202 of file ValidateSAX2ContentHandler.hpp.

Referenced by instanceIn().

3.7.3.11 std::string SAX2ContentHandler::getName ( ) const [inline]

Returns the name.

Returns

Name.

Definition at line 166 of file ValidateSAX2ContentHandler.hpp.

Referenced by instanceIn().

3.7.3.12 std::string SAX2ContentHandler::getSource ( ) const [inline]

Returns the source.

Returns

Source.

Definition at line 174 of file ValidateSAX2ContentHandler.hpp.

Referenced by instanceIn().

3.7.3.13 void SAX2ContentHandler::ignorableWhitespace (const XMLCh *const *chars*, const XMLSize_t *length*)

Called if whitespaces occurs. The method does nothing.

Parameters

chars	Whitespace data.
length	Length of the Character data.

Definition at line 414 of file ValidateSAX2ContentHandler.cpp.

3.7.3.14 void SAX2ContentHandler::init ( ) [private]

Initates the class variables.

Definition at line 97 of file ValidateSAX2ContentHandler.cpp.

Referenced by resetDocument(), SAX2ContentHandler(), and startDocument().

3.7.3.15 SAX2ContentHandler& SAX2ContentHandler::operator= (const SAX2ContentHandler & *sAX2ContentHandler*) [private]

Not implemented operator "=".

Parameters

sAX2-Content-Handler	Right side of the operator.

Returns

Left side of the operator.

3.7.3.16 void SAX2ContentHandler::processingInstruction (const XMLCh ∗const *target,* const XMLCh ∗const *data*)

Called if processing instructions occurs. The method does nothing.

Parameters

target	Processing instruction target.
data	Processing instruction data.

Definition at line 417 of file ValidateSAX2ContentHandler.cpp.

3.7.3.17 void SAX2ContentHandler::resetDocument ( )

Reset the Document object on its reuse. The method does nothing.

Definition at line 420 of file ValidateSAX2ContentHandler.cpp.

References init().

3.7.3.18 void SAX2ContentHandler::setDocumentLocator (const Locator ∗const *locator*)

Receive a Locator object for document events. The method does nothing.

Parameters

locator	Object that can return the location of any SAX document event.

Definition at line 124 of file ValidateSAX2ContentHandler.cpp.

3.7.3.19 void SAX2ContentHandler::skippedEntity (const XMLCh ∗const *name*)

Receive notification of a skipped entity. The method does nothing.

Parameters

name	Name of the skipped entity.

Definition at line 189 of file ValidateSAX2ContentHandler.cpp.

3.7.3.20 void SAX2ContentHandler::startDocument ( )

Called if the document starts.

Definition at line 127 of file ValidateSAX2ContentHandler.cpp.

References init().

3.7.3.21 void SAX2ContentHandler::startElement (const XMLCh ∗const *uri,* const XMLCh ∗const *localname,* const XMLCh ∗const *qname,* const Attributes & *attributes*)

Called if an element starts.

Parameters

uri	URI of the associated namespace for this element.
localname	Local part of the element name
qname	QName of this element.
attributes	Attributes attached to the element, if any.

Definition at line 192 of file ValidateSAX2ContentHandler.cpp.

References buffer, cost, costsOfEdgesDefinedInTheFirstVertex, failed, firstVertex, n, stringForm, trim(), unicodeForm, XML_EDGE, and XML_VERTEX.

3.7.3.22 void SAX2ContentHandler::startPrefixMapping (const XMLCh ∗const *prefix,* const XMLCh ∗const *uri*)

Receive notification of the start of an namespace prefix mapping. The method does nothing

Parameters

prefix	namespace prefix used.
uri	Namespace URI used.

Definition at line 183 of file ValidateSAX2ContentHandler.cpp.

3.7.4 Member Data Documentation

3.7.4.1 std::vector<std::vector<double> > SAX2ContentHandler::adjacencyMatrix [private]

Adjacency matrix of a weighted undirected complete graph. The elements in the matrix are the costs. The elements in the matrix are the costs. The matrix can be either only a lower triangular matrix for undirected graphs or it can be a square matrix for the directed graph.

Definition at line 109 of file ValidateSAX2ContentHandler.hpp.

Referenced by endDocument(), and endElement().

3.7.4.2 std::string SAX2ContentHandler::buffer [private]

Buffer to parse character data.

Definition at line 114 of file ValidateSAX2ContentHandler.hpp.

Referenced by characters(), endElement(), and startElement().

3.7.4.3 double SAX2ContentHandler::cost [private]

Cost of the current edge.

Definition at line 119 of file ValidateSAX2ContentHandler.hpp.

Referenced by endElement(), and startElement().

3.7.4.4 std::vector<double> SAX2ContentHandler::costsOfEdgesDefinedInTheFirstVertex [private]

Cost of edges incident with the first vertes.

Definition at line 90 of file ValidateSAX2ContentHandler.hpp.

Referenced by endElement(), and startElement().

3.7.4.5 std::string SAX2ContentHandler::description [private]

Description of the instance.

Definition at line 54 of file ValidateSAX2ContentHandler.hpp.

Referenced by endElement().

3.7.4.6 std::streamsize SAX2ContentHandler::doublePrecision [private]

Precision of doubles.

Definition at line 59 of file ValidateSAX2ContentHandler.hpp.

Referenced by endElement().

3.7.4.7 double SAX2ContentHandler::doubleZero [private]

Maximum of an absolute value considered as zero.

Definition at line 70 of file ValidateSAX2ContentHandler.hpp.

Referenced by endDocument(), and endElement().

3.7.4.8 std::set<std::vector<double>::size_type> SAX2ContentHandler::edgesDefinedInTheFirstVertex [private]

Edges defined in the first vertex

Definition at line 85 of file ValidateSAX2ContentHandler.hpp.

Referenced by endElement().

3.7.4.9 bool SAX2ContentHandler::failed [private]

True if the last internal validation fails.

Definition at line 30 of file ValidateSAX2ContentHandler.hpp.

Referenced by endDocument(), endElement(), and startElement().

3.7.4.10 bool SAX2ContentHandler::firstVertex [private]

true during parsing the first vertex

Definition at line 75 of file ValidateSAX2ContentHandler.hpp.

Referenced by endElement(), and startElement().

3.7.4.11 std::streamsize SAX2ContentHandler::ignoredDigits [private]

Number of ignored digits of double types. (The deviation of double values can be at most 1e-(DoublePrecision - IgnoredDigits).)

Definition at line 65 of file ValidateSAX2ContentHandler.hpp.

Referenced by endElement().

3.7.4.12 bool SAX2ContentHandler::isUndirected [private]

Indicator indicating if the graph is undirected.

- true The Graph is undirected.
- false The Graph is directed.

Definition at line 39 of file ValidateSAX2ContentHandler.hpp.

Referenced by endDocument().

3.7.4.13 std::vector<std::vector<double> >::size_type SAX2ContentHandler::n [private]

Number of vertices in the graph of the instance.

Definition at line 95 of file ValidateSAX2ContentHandler.hpp.

Referenced by endDocument(), endElement(), and startElement().

3.7.4.14 std::string SAX2ContentHandler::name [private]

Name of the instance.

Definition at line 44 of file ValidateSAX2ContentHandler.hpp.

Referenced by endElement().

3.7.4.15 std::vector<std::vector<double> >::size_type SAX2ContentHandler::numberOfParsedVertices [private]

Number of parsed vertices.

Definition at line 80 of file ValidateSAX2ContentHandler.hpp.

Referenced by endElement().

3.7.4.16 std::vector<std::vector<bool> > SAX2ContentHandler::parsedEntries [private]

Matrix indicating which vertices were parsed.

Definition at line 100 of file ValidateSAX2ContentHandler.hpp.

Referenced by endDocument(), and endElement().

3.7.4.17 std::string SAX2ContentHandler::source [private]

Source of the instance.

Definition at line 49 of file ValidateSAX2ContentHandler.hpp.

Referenced by endElement().

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

- src/ValidateSAX2ContentHandler.hpp
- src/ValidateSAX2ContentHandler.cpp

3.8 SAX2ErrorHandler Class Reference

`#include <ValidateSAX2ErrorHandler.hpp>`

Public Member Functions

- SAX2ErrorHandler ()
- ~SAX2ErrorHandler ()
- bool getFailed ()
- void warning (const SAXParseException &exc)
- void error (const SAXParseException &exc)
- void fatalError (const SAXParseException &exc)
- void resetErrors ()

Private Member Functions

- SAX2ErrorHandler (const SAX2ErrorHandler &sAXErrorHandler)
- SAX2ErrorHandler & operator= (const SAX2ErrorHandler &sAXErrorHandler)

Private Attributes

- bool failed

3.8.1 Detailed Description

Implements an ErrorHandler

Definition at line 24 of file ValidateSAX2ErrorHandler.hpp.

3.8.2 Constructor & Destructor Documentation

3.8.2.1 SAX2ErrorHandler::SAX2ErrorHandler (const SAX2ErrorHandler & *sAXErrorHandler*) [private]

Not implemented copy constructor for the class SAX2ErrorHandler.

Parameters

sAXError-Handler	Instance to be copied.

3.8.2.2 SAX2ErrorHandler::SAX2ErrorHandler ( )

Constructor for the class SAXErrorHandler.

Definition at line 14 of file ValidateSAX2ErrorHandler.cpp.

References failed.

3.8.2.3 SAX2ErrorHandler::~SAX2ErrorHandler ( )

Destructor for the class SAXErrorHandler.

Definition at line 18 of file ValidateSAX2ErrorHandler.cpp.

3.8.3 Member Function Documentation

3.8.3.1 void SAX2ErrorHandler::error (const SAXParseException & *exc*)

Error handling interface for errors. The method does nothing.

Parameters

exc	Error.

Definition at line 25 of file ValidateSAX2ErrorHandler.cpp.

References failed.

3.8.3.2 void SAX2ErrorHandler::fatalError (const SAXParseException & *exc*)

Error handling interface for fatal errors. The method does nothing

Parameters

exc	Fatal error.

Definition at line 29 of file ValidateSAX2ErrorHandler.cpp.

References failed.

3.8.3.3 bool SAX2ErrorHandler::getFailed ( ) [inline]

Returns true if the last validation failed

Returns

- true if the last validation failed
- false if the last validation failed

Definition at line 63 of file ValidateSAX2ErrorHandler.hpp.

Referenced by instanceIn().

3.8.3.4 SAX2ErrorHandler& SAX2ErrorHandler::operator= (const SAX2ErrorHandler & *sAXErrorHandler*) [private]

Not implemented operator "=".

Parameters

sAXError-Handler	Right side of the operator.

Returns

Left side of the operator.

3.8.3.5 void SAX2ErrorHandler::resetErrors ( )

Reset the Error handler object on its reuse. The method does nothing.

Definition at line 33 of file ValidateSAX2ErrorHandler.cpp.

References failed.

3.8.3.6 void SAX2ErrorHandler::warning (const SAXParseException & *exc*)

Error handling interface for warnings. The method does nothing.

Parameters

exc	Warning.

Definition at line 21 of file ValidateSAX2ErrorHandler.cpp.

References failed.

3.8.4 Member Data Documentation

3.8.4.1 bool SAX2ErrorHandler::failed [private]

True if a validation fails.

Definition at line 29 of file ValidateSAX2ErrorHandler.hpp.

Referenced by error(), fatalError(), resetErrors(), SAX2ErrorHandler(), and warning().

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

- src/ValidateSAX2ErrorHandler.hpp
- src/ValidateSAX2ErrorHandler.cpp

3.9 StringTranscode Class Reference

Public Member Functions

- StringTranscode (const XMLCh ∗toTranscode)
- ∼StringTranscode ()
- const string stringForm () const

Private Attributes

- string stringStringForm

3.9.1 Detailed Description

Trancodes XMLCh data to string.

Definition at line 64 of file ValidateSAX2ContentHandler.cpp.

3.9.2 Constructor & Destructor Documentation

3.9.2.1 StringTranscode::StringTranscode (const XMLCh ∗ *toTranscode*) [inline]

Constructor for the class StringTranscode.

Parameters

toTranscode	String that should be transcoded.

Definition at line 75 of file ValidateSAX2ContentHandler.cpp.

3.9.2.2 StringTranscode::∼StringTranscode ( ) [inline]

Destructor for the class StringTranscode

Definition at line 84 of file ValidateSAX2ContentHandler.cpp.

3.9.3 Member Function Documentation

3.9.3.1 const string StringTranscode::stringForm ( ) const [inline]

Returns the string format of the unicode XMLCh format.

Returns

String format of the unicode XMLCh format.

Definition at line 91 of file ValidateSAX2ContentHandler.cpp.

3.9.4 Member Data Documentation

3.9.4.1 string StringTranscode::stringStringForm [private]

String format of the unicode XMLCh format.

Definition at line 69 of file ValidateSAX2ContentHandler.cpp.

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

- src/ValidateSAX2ContentHandler.cpp

3.10 TransformDOMErrorHandler Class Reference

`#include <TransformDOMErrorHandler.hpp>`

Public Member Functions

- TransformDOMErrorHandler ()
- ~TransformDOMErrorHandler ()
- bool handleError (const DOMError &dOMError)

Private Member Functions

- TransformDOMErrorHandler (const DOMErrorHandler &dOMErrorHandler)
- DOMErrorHandler & operator= (const DOMErrorHandler &dOMErrorHandler)

3.10.1 Detailed Description

Implements a DOMErrorHandler

Definition at line 24 of file TransformDOMErrorHandler.hpp.

3.10.2 Constructor & Destructor Documentation

3.10.2.1 TransformDOMErrorHandler::TransformDOMErrorHandler (const DOMErrorHandler & *dOMErrorHandler*) [private]

Not implemented copy constructor for the class TransformDOMErrorHandler.

Parameters

dOMError-Handler	Instance to be copied.

3.10.2.2 TransformDOMErrorHandler::TransformDOMErrorHandler ( )

Constructor for the class TransformDOMErrorHandler.

Definition at line 17 of file TransformDOMErrorHandler.cpp.

3.10.2.3 TransformDOMErrorHandler::~TransformDOMErrorHandler ( )

Destructor for the class TransformDOMErrorHandler.

Definition at line 20 of file TransformDOMErrorHandler.cpp.

3.10.3 Member Function Documentation

3.10.3.1 bool TransformDOMErrorHandler::handleError (const DOMError & *dOMError*)

Error handling interface. The methode returns always false

Parameters

dOMError	Error.

Returns

- true if it should be continued,
- false if it should not be continued.

Definition at line 23 of file TransformDOMErrorHandler.cpp.

3.10.3.2 DOMErrorHandler& TransformDOMErrorHandler::operator= (const DOMErrorHandler & *dOMErrorHandler*) [private]

Not implemented operator "=".

Parameters

dOMError-Handler	Right side of the operator.

Returns

Left side of the operator.

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

- src/TransformDOMErrorHandler.hpp
- src/TransformDOMErrorHandler.cpp

3.11 TransformInstance Class Reference

`#include <TransformConstantsClassesAndFunctions.hpp>`

Public Member Functions

- TransformInstance (const std::string &type, const std::string &name, const std::string &source, const std::string &description, const std::vector< std::vector< double > >::size_type n)
- TransformInstance (const TransformInstance &transformInstance)
- TransformInstance & operator= (const TransformInstance &transformInstance)
- std::string getType () const
- std::string getName () const
- std::string getSource () const
- std::string getDescription () const
- std::vector< std::vector < double > >::size_type getN () const
- void setAdjacencyMatrixElement (const std::vector< std::vector< double > >::size_type i, const std::vector< double >::size_type j, const double value)
- double getAdjacencyMatrixElement (const std::vector< std::vector< double > >::size_type i, const std::vector< double >::size_type j) const

Private Member Functions

- void init (const std::string &type, const std::string &name, const std::string &source, const std::string &description, const std::vector< std::vector< double > >::size_type n)

Private Attributes

- std::string type
- std::string name
- std::string source
- std::string description
- std::vector< std::vector < double > >::size_type n
- std::vector< std::vector < double > > adjacencyMatrix

3.11.1 Detailed Description

Saves one instance with the graph in the form of an adjacency matrix. If the instance is symmetric the program saves only the lower part of the adjacency matrix. This class provides no checks of validity or ranges.

Definition at line 331 of file TransformConstantsClassesAndFunctions.hpp.

3.11.2 Constructor & Destructor Documentation

3.11.2.1 TransformInstance::TransformInstance (const std::string & *type*, const std::string & *name*, const std::string & *source*, const std::string & *description*, const std::vector< std::vector< double > >::size_type *n*)

Constructor for the class TransformInstance.

Parameters

type	Type of the instance.
name	Name of the instance.
source	Source of the instance.
description	Description of the instance.
n	Number of vertices in the graph of the instance.

Definition at line 52 of file TransformConstantsClassesAndFunctions.cpp.

References VALUE_TYPE_TSP.

3.11.2.2 TransformInstance::TransformInstance (const TransformInstance & *transformInstance*)

Copy constructor for the class TransformInstance.

Parameters

transform-Instance	Instance to be copied.

Definition at line 73 of file TransformConstantsClassesAndFunctions.cpp.

References getDescription(), getN(), getName(), getSource(), and getType().

3.11.3 Member Function Documentation

3.11.3.1 double TransformInstance::getAdjacencyMatrixElement (const std::vector< std::vector< double > >::size_type *i*, const std::vector< double >::size_type *j*) const [inline]

Returns one element in the adjacency matrix. The parameters are not checked.

Parameters

i	Row.
j	Column.

Returns

Value of the element on the i-th row and j-th column.

Definition at line 483 of file TransformConstantsClassesAndFunctions.hpp.

References adjacencyMatrix, type, and VALUE_TYPE_TSP.

Referenced by readInputFileTSPLIB(), writeOutputFile(), and writeOutputFileWithoutUsingAParser().

3.11.3.2 std::string TransformInstance::getDescription ( ) const [inline]

Returns the description of the instance.

Returns

Description of the instance.

Definition at line 437 of file TransformConstantsClassesAndFunctions.hpp.

References description.

Referenced by operator=(), TransformInstance(), writeOutputFile(), and writeOutputFileWithoutUsingAParser().

3.11.3.3 std::vector<std::vector<double> >::size_type TransformInstance::getN ( ) const [inline]

Returns the number of vertices in the graph of the instance.

Returns

Number of vertices in the graph of the instance.

Definition at line 445 of file TransformConstantsClassesAndFunctions.hpp.

References n.

Referenced by main(), operator=(), TransformInstance(), writeOutputFile(), and writeOutputFileWithoutUsingAParser().

3.11.3.4 std::string TransformInstance::getName ( ) const [inline]

Returns the name of the instance.

Returns

Name of the instance.

Definition at line 421 of file TransformConstantsClassesAndFunctions.hpp.

References name.

Referenced by operator=(), TransformInstance(), writeOutputFile(), and writeOutputFileWithoutUsingAParser().

3.11.3.5 std::string TransformInstance::getSource ( ) const [inline]

Returns the source of the instance.

Returns

Source of the instance.

Definition at line 429 of file TransformConstantsClassesAndFunctions.hpp.

References source.

Referenced by operator=(), TransformInstance(), and writeOutputFile().

3.11.3.6 std::string TransformInstance::getType ( ) const [inline]

Returns the type of the instance.

Returns

Type of the instance.

Definition at line 413 of file TransformConstantsClassesAndFunctions.hpp.

References type.

Referenced by operator=(), TransformInstance(), writeOutputFile(), and writeOutputFileWithoutUsingAParser().

3.11.3.7 XERCES_CPP_NAMESPACE_USE void TransformInstance::init (const std::string & *type,* const std::string & *name,* const std::string & *source,* const std::string & *description,* const std::vector< std::vector< double > >::size_type *n*) [private]

Initializes the name, the source, the description and n

Parameters

type	Type of the instance.
name	Name of the instance.
source	Source of the instance.
description	Description of the instance.
n	Number of vertices in the graph of the instance.

Definition at line 39 of file TransformConstantsClassesAndFunctions.cpp.

3.11.3.8 TransformInstance & TransformInstance::operator= (const TransformInstance & *transformInstance*)

Implements the operator "=".

Parameters

transform-Instance	Right side of the operator.

Returns

Left side of the operator.

Definition at line 90 of file TransformConstantsClassesAndFunctions.cpp.

References getDescription(), getN(), getName(), getSource(), and getType().

3.11.3.9 void TransformInstance::setAdjacencyMatrixElement (const std::vector< std::vector< double > >::size_type *i,* const std::vector< double >::size_type *j,* const double *value*) [inline]

Sets one element in the adjacency matrix. The parameters are not checked. The new value will be rounded according to the constants DOUBLE_PRECISION and IGNORED_DIGITS.

Parameters

i	Row.
j	Column.
value	New value of the element on the i-th row and j-th column.

Definition at line 457 of file TransformConstantsClassesAndFunctions.hpp.

References adjacencyMatrix, roundToDoublePrecisionAndDoubleFloatField(), type, and VALUE_TYPE_TSP.

Referenced by readInputFileTSPLIB().

3.11.4 Member Data Documentation

3.11.4.1 std::vector<std::vector<double> > TransformInstance::adjacencyMatrix [private]

Adjacency matrix of the instance. If the instance is symmetric the program saves only the lower part of the adjacency matrix.

Definition at line 363 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by getAdjacencyMatrixElement(), and setAdjacencyMatrixElement().

3.11.4.2 std::string TransformInstance::description [private]

Description of the instance.

Definition at line 351 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by getDescription().

3.11.4.3 std::vector<std::vector<double> >::size_type TransformInstance::n [private]

Number of vertices in the graph of the instance.

Definition at line 356 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by getN().

3.11.4.4 std::string TransformInstance::name [private]

Name of the instance.

Definition at line 341 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by getName().

3.11.4.5 std::string TransformInstance::source [private]

Source of the instance.

Definition at line 346 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by getSource().

3.11.4.6 std::string TransformInstance::type [private]

Type of the instance.

Definition at line 336 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by getAdjacencyMatrixElement(), getType(), and setAdjacencyMatrixElement().

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

- src/TransformConstantsClassesAndFunctions.hpp
- src/TransformConstantsClassesAndFunctions.cpp

3.12 ValidationFailed Class Reference

`#include <ValidateConstantsFunctionsAndClasses.hpp>`

Public Member Functions

- ValidationFailed ()

3.12.1 Detailed Description

Exception class used if a validation fails.

Definition at line 157 of file ValidateConstantsFunctionsAndClasses.hpp.

3.12.2 Constructor & Destructor Documentation

3.12.2.1 ValidationFailed::ValidationFailed ( ) [inline]

Constructor for the class ValidationFailed.

Definition at line 162 of file ValidateConstantsFunctionsAndClasses.hpp.

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

- src/ValidateConstantsFunctionsAndClasses.hpp

3.13 ValidationSchemaDoesNotExist Class Reference

`#include <ValidateConstantsFunctionsAndClasses.hpp>`

Public Member Functions

- ValidationSchemaDoesNotExist ()

3.13.1 Detailed Description

Exception class used if a validation schema does not exist.

Definition at line 145 of file ValidateConstantsFunctionsAndClasses.hpp.

3.13.2 Constructor & Destructor Documentation

3.13.2.1 ValidationSchemaDoesNotExist::ValidationSchemaDoesNotExist ( ) [inline]

Constructor for the class ValidationSchemaDoesNotExist.

Definition at line 150 of file ValidateConstantsFunctionsAndClasses.hpp.

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

- src/ValidateConstantsFunctionsAndClasses.hpp

3.14 XMLStringTranscode Class Reference

Public Member Functions

- XMLStringTranscode (const string toTranscode)
- ~XMLStringTranscode ()
- const XMLCh * unicodeForm () const
- XMLStringTranscode (const string toTranscode)
- ~XMLStringTranscode ()
- const XMLCh * unicodeForm () const
- XMLStringTranscode (const string toTranscode)
- ~XMLStringTranscode ()
- const XMLCh * unicodeForm () const

Private Attributes

- XMLCh * stringUnicodeForm

3.14.1 Detailed Description

Trancodes char* data to XMLCh data.

Definition at line 1228 of file TransformConstantsClassesAndFunctions.cpp.

3.14.2 Constructor & Destructor Documentation

3.14.2.1 XMLStringTranscode::XMLStringTranscode (const string *toTranscode*) [inline]

Constructor for the class XMLStringTranscode.

Parameters

toTranscode	String that should be transcoded.

Definition at line 1239 of file TransformConstantsClassesAndFunctions.cpp.

3.14.2.2 XMLStringTranscode::~XMLStringTranscode ( ) [inline]

Destructor for the class XMLString.

Definition at line 1247 of file TransformConstantsClassesAndFunctions.cpp.

3.14.2.3 XMLStringTranscode::XMLStringTranscode (const string *toTranscode*) [inline]

Constructor for the class XMLStringTranscode.

Parameters

toTranscode	String that should be transcoded.

Definition at line 99 of file ValidateIO.cpp.

3.14.2.4 XMLStringTranscode::~XMLStringTranscode ( ) [inline]

Destructor for the class XMLString.

Definition at line 107 of file ValidateIO.cpp.

3.14.2.5 XMLStringTranscode::XMLStringTranscode (const string *toTranscode*) [inline]

Constructor for the class XMLStringTranscode.

Parameters

toTranscode	String that should be transcoded.

Definition at line 39 of file ValidateSAX2ContentHandler.cpp.

3.14.2.6 XMLStringTranscode::~XMLStringTranscode ( ) [inline]

Destructor for the class XMLString.

Definition at line 47 of file ValidateSAX2ContentHandler.cpp.

3.14.3 Member Function Documentation

3.14.3.1 const XMLCh* XMLStringTranscode::unicodeForm ( ) const [inline]

Returns the unicode XMLCh format of the string.

Returns

Unicode XMLCh format of the string.

Definition at line 55 of file ValidateSAX2ContentHandler.cpp.

3.14.3.2 const XMLCh* XMLStringTranscode::unicodeForm ( ) const [inline]

Returns the unicode XMLCh format of the string.

Returns

Unicode XMLCh format of the string.

Definition at line 115 of file ValidateIO.cpp.

3.14.3.3 const XMLCh* XMLStringTranscode::unicodeForm ( ) const [inline]

Returns the unicode XMLCh format of the string.

Returns

Unicode XMLCh format of the string.

Definition at line 1255 of file TransformConstantsClassesAndFunctions.cpp.

3.14.4 Member Data Documentation

3.14.4.1 XMLCh * XMLStringTranscode::stringUnicodeForm [private]

Unicode XMLCh format of the string.

Definition at line 1233 of file TransformConstantsClassesAndFunctions.cpp.

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

- src/TransformConstantsClassesAndFunctions.cpp
- src/ValidateIO.cpp
- src/ValidateSAX2ContentHandler.cpp

Chapter 4

File Documentation

4.1 src/TransformConstantsClassesAndFunctions.cpp File - Reference

Defines the constants, types and some functions necessary for the transform programs.

```
#include <iostream> #include <stdexcept> #include <limits> ×
#include <algorithm> #include <fstream> #include <sstream> ×
#include <iomanip>  #include <string>  #include <vector>
#include <xercesc/util/PlatformUtils.hpp> #include <xercesc/util/-
XMLString.hpp>  #include <xercesc/dom/DOM.hpp>  #include
<xercesc/util/OutOfMemoryException.hpp> #include <xercesc/framework/-
LocalFileFormatTarget.hpp>  #include "TransformDOMError-
Handler.hpp" #include "TransformConstantsClassesAndFunctions.-
hpp"
```

Classes

- struct Point
- class XMLStringTranscode

Defines

- #define unicodeForm(str) XMLStringTranscode(str).unicodeForm()

Functions

- TransformInstance * readInputFileTSPLIB (const string &inputFileName)
- void writeOutputFile (const string &outputFileName, const TransformInstance *transformInstance)

- void writeOutputFileWithoutUsingAParser (const std::string &outputFileName, const TransformInstance *transformInstance)

4.1.1 Detailed Description

Defines the constants, types and some functions necessary for the transform programs. Defines the constants, classes and some functions necessary for the transform programs.

Author

Ulrich Pferschy and Rostislav Stanek (Institut fuer Statistik und Operations - Research, Universitaet Graz)

Definition in file TransformConstantsClassesAndFunctions.cpp.

4.1.2 Define Documentation

4.1.2.1 #define unicodeForm(*str*) XMLStringTranscode(str).unicodeForm()

Definition at line 1259 of file TransformConstantsClassesAndFunctions.cpp.

Referenced by writeOutputFile().

4.1.3 Function Documentation

4.1.3.1 TransformInstance* readInputFileTSPLIB (const string & *inputFileName*)

Definition at line 123 of file TransformConstantsClassesAndFunctions.cpp.

References DISPLAY_DATA_SECTION, EDGE_WEIGHT_SECTION, TransformInstance::getAdjacencyMatrixElement(), INPUT_FILE_FILENAME_EXTENSION_ATSP, INPUT_FILE_FILENAME_EXTENSION_TSP, RRR, TransformInstance::setAdjacencyMatrixElement(), TAG_COMMENT, TAG_DIMENSION, TAG_DISPLAY_DATA_TYPE, TAG_EDGE_WEIGHT_FORMAT, TAG_EDGE_WEIGHT_TYPE, TAG_EOF, TAG_NAME, TAG_NODE_COORD_SECTION, TAG_TYPE, TRANSFORM_DOUBLE_ZERO, trim(), VALUE_DISPLAY_DATA_TYPE_COORD_DISPLAY, VALUE_DISPLAY_DATA_TYPE_TWOD_DISPLAY, VALUE_EDGE_WEIGHT_FORMAT_FULL_MATRIX, VALUE_EDGE_WEIGHT_FORMAT_FUNCTION, VALUE_EDGE_WEIGHT_FORMAT_LOWER_DIAG_ROW, VALUE_EDGE_WEIGHT_FORMAT_UPPER_DIAG_ROW, VALUE_EDGE_WEIGHT_FORMAT_UPPER_ROW, VALUE_EDGE_WEIGHT_TYPE_ATT, VALUE_EDGE_WEIGHT_TYPE_CEIL_2D, VALUE_EDGE_WEIGHT_TYPE_EUC_2D, VALUE_EDGE_WEIGHT_TYPE_EXPLICIT, VALUE_EDGE_WEIGHT_TYPE_GEO, VALUE_TYPE_ATSP, VALUE_TYPE_TSP, Point::x, XML_VALUE_SOURCE_TSPLIB, and Point::y.

Referenced by main().

4.1.3.2 void writeOutputFile (const string & *outputFileName,* const TransformInstance * *transformInstance*)

Definition at line 1261 of file TransformConstantsClassesAndFunctions.cpp.

References DOUBLE_FLOATFIELD, TransformInstance::getAdjacencyMatrixElement(), TransformInstance::getDescription(), TransformInstance::getN(), TransformInstance::getName(), TransformInstance::getSource(), TransformInstance::getType(), TRANSFORM_DOUBLE_PRECISION, TRANSFORM_IGNORED_DIGITS, unicodeForm, VALUE_TYPE_ATSP, XML_DESCRIPTION, XML_DOCUMENT_NODE, XML_DOUBLE_PRECISION, XML_EDGE, XML_EDGE_ATTRIBUTE_COST, XML_ENCODING, XML_FORMAT_PRETTY_PRINT, XML_GRAPH, XML_IGNORED_DIGITS, XML_NAME, XML_SOURCE, and XML_VERTEX.

Referenced by main().

4.1.3.3 void writeOutputFileWithoutUsingAParser (const std::string & *outputFileName,* const TransformInstance * *transformInstance*)

Writes one instance of the class TransformInstance to an output file without using a parser. This method is quicker than the previous one but it provides no parser guarantee that the output file is a valid xml file. The parameters are not checked.

Parameters

outputFileName	Name of the output file.
transformInstance	Instance of the class TransformInstance.

Definition at line 1525 of file TransformConstantsClassesAndFunctions.cpp.

References DOUBLE_FLOATFIELD, TransformInstance::getAdjacencyMatrixElement(), TransformInstance::getDescription(), TransformInstance::getN(), TransformInstance::getName(), TransformInstance::getType(), TRANSFORM_DOUBLE_PRECISION, TRANSFORM_IGNORED_DIGITS, and VALUE_TYPE_ATSP.

Referenced by main().

4.2 src/TransformConstantsClassesAndFunctions.hpp File Reference

Defines the constants, types and some functions necessary for the transform programs.

```
#include <cmath>  #include <string>  #include <vector> x
#include <iomanip> #include <xercesc/util/PlatformUtils.-
hpp>
```

Classes

- class InputFileFormatNotSupported
- class TransformInstance

Functions

- void trimLeft (std::string &s, const std::string &t=" \t\r\n")
- void trimRight (std::string &s, const std::string &t=" \t\r\n")
- void trim (std::string &s, const std::string &t=" \t\r\n")
- double roundToDoublePrecisionAndDoubleFloatField (double d)
- TransformInstance * readInputFileTSPLIB (const std::string &inputFileName)
- void writeOutputFile (const std::string &outputFileName, const TransformInstance *transformInstance)
- void writeOutputFileWithoutUsingAParser (const std::string &outputFileName, const TransformInstance *transformInstance)

Variables

- const std::string INPUT_FILE_FILENAME_EXTENSION_TSP = ".tsp"
- const std::string INPUT_FILE_FILENAME_EXTENSION_ATSP = ".atsp"
- const std::string TAG_NAME = "NAME:"
- const std::string TAG_TYPE = "TYPE:"
- const std::string TAG_COMMENT = "COMMENT:"
- const std::string TAG_DIMENSION = "DIMENSION:"
- const std::string TAG_EDGE_WEIGHT_TYPE = "EDGE_WEIGHT_TYPE:"
- const std::string TAG_EDGE_WEIGHT_FORMAT = "EDGE_WEIGHT_FORMAT:"
- const std::string TAG_DISPLAY_DATA_TYPE = "DISPLAY_DATA_TYPE:"
- const std::string TAG_NODE_COORD_SECTION = "NODE_COORD_SECTION"
- const std::string EDGE_WEIGHT_SECTION = "EDGE_WEIGHT_SECTION"
- const std::string DISPLAY_DATA_SECTION = "DISPLAY_DATA_SECTION"
- const std::string TAG_EOF = "EOF"
- const std::string VALUE_TYPE_TSP = "TSP"
- const std::string VALUE_TYPE_ATSP = "ATSP"
- const double RRR = 6378.388
- const std::string VALUE_EDGE_WEIGHT_TYPE_GEO = "GEO"
- const std::string VALUE_EDGE_WEIGHT_TYPE_EUC_2D = "EUC_2D"
- const std::string VALUE_EDGE_WEIGHT_TYPE_CEIL_2D = "CEIL_2D"
- const std::string VALUE_EDGE_WEIGHT_TYPE_ATT = "ATT"
- const std::string VALUE_EDGE_WEIGHT_TYPE_EXPLICIT = "EXPLICIT"
- const std::string VALUE_EDGE_WEIGHT_FORMAT_FUNCTION = "FUNCTION"
- const std::string VALUE_EDGE_WEIGHT_FORMAT_FULL_MATRIX = "FULL_MATRIX"

- const std::string VALUE_EDGE_WEIGHT_FORMAT_LOWER_DIAG_ROW = "-LOWER_DIAG_ROW"
- const std::string VALUE_EDGE_WEIGHT_FORMAT_UPPER_DIAG_ROW = "-UPPER_DIAG_ROW"
- const std::string VALUE_EDGE_WEIGHT_FORMAT_UPPER_ROW = "UPPER-_ROW"
- const std::string VALUE_DISPLAY_DATA_TYPE_COORD_DISPLAY = "COOR-D_DISPLAY"
- const std::string VALUE_DISPLAY_DATA_TYPE_TWOD_DISPLAY = "TWOD_-DISPLAY"
- const std::string OUTPUT_FILE_FILENAME_EXTENSION = ".xml"
- const bool XML_FORMAT_PRETTY_PRINT = true
- const std::string XML_DOCUMENT_NODE = "travellingSalesmanProblem-Instance"
- const std::string XML_NAME = "name"
- const std::string XML_SOURCE = "source"
- const std::string XML_DESCRIPTION = "description"
- const std::string XML_DOUBLE_PRECISION = "doublePrecision"
- const std::string XML_IGNORED_DIGITS = "ignoredDigits"
- const std::string XML_GRAPH = "graph"
- const std::string XML_VERTEX = "vertex"
- const std::string XML_EDGE = "edge"
- const std::string XML_EDGE_ATTRIBUTE_COST = "cost"
- const std::string XML_ENCODING = "UTF-8"
- const std::string XML_VALUE_SOURCE_TSPLIB = "TSPLIB"
- const std::ios::fmtflags DOUBLE_FLOATFIELD = std::ios::scientific
- const std::streamsize TRANSFORM_DOUBLE_PRECISION = std::numeric_-limits<double>::digits10
- const std::streamsize TRANSFORM_IGNORED_DIGITS = 5
- const double TRANSFORM_DOUBLE_ZERO

4.2.1 Detailed Description

Defines the constants, types and some functions necessary for the transform programs. Defines the constants, classes and some functions necessary for the transform programs.

Author

Ulrich Pferschy and Rostislav Stanek (Institut fuer Statistik und Operations - Research, Universitaet Graz)

Definition in file TransformConstantsClassesAndFunctions.hpp.

4.2.2 Function Documentation

4.2.2.1 TransformInstance* readInputFileTSPLIB (const std::string & *inputFileName*)

Reads the input file and creates an instance of the class TransformInstance. The parameters are not checked. Every keyword must be on a new line and is allowed to be used only once. Note that the symbol ":" has to follow the keywords immediately. The order possibilities of the input keywords are: 1) "NAME:", "TYPE:" (= "TSP"), "COMMENT:", "DIMENSION:", "EDGE_WEIGHT_TYPE:" (= "GEO"), ("EDGE_WEIGHT_FORMAT:" (= "FUNCTION"),) DISPLAY_DATA_TYPE:" (= "COORD_DISPLAY"), "NODE_COORD_SECTION", coordinates, "EOF", 2) "NAME:", "TYPE:", "COMMENT:", "DIMENSION:", "EDGE_WEIGHT_TYPE:" (= "EUC_2D" / "CEIL_2D"), "NODE_COORD_SECTION", coordinates, "EOF", 3) "NAME:", "TYPE:", "COMMENT:", "DIMENSION:", "EDGE_WEIGHT_TYPE:" (= "ATT"), "NODE_COORD_SECTION", coordinates, "EOF", 4) "NAME:", "TYPE:" (= "TSP"), "COMMENT:", "DIMENSION:", "EDGE_WEIGHT_TYPE:" (= "EXPLICIT"), "EDGE_WEIGHT_FORMAT:" (= "FULL_MATRIX"), "EDGE_WEIGHT_SECTION", matrix entries, "EOF", 5) "NAME:", "TYPE:" (= "TSP"), "COMMENT:", "DIMENSION:", "EDGE_WEIGHT_TYPE:" (= "EXPLICIT"), "EDGE_WEIGHT_FORMAT:" (= "FULL_MATRIX"), "TAG_DISPLAY_DATA_TYPE" (= "TWOD_DISPLAY"), "EDGE_WEIGHT_SECTION", matrix entries, DISPLAY_DATA_SECTION, coordinates, "EOF", 6) "NAME:", "TYPE:" (= "TSP"), "COMMENT:", "DIMENSION:", "EDGE_WEIGHT_TYPE:" (= "EXPLICIT"), "EDGE_WEIGHT_FORMAT:" (= "LOWER_DIAG_ROW"), "EDGE_WEIGHT_SECTION", matrix entries, "EOF", 7) "NAME:", "TYPE:" (= "TSP"), "COMMENT:", "DIMENSION:", "EDGE_WEIGHT_TYPE:" (= "EXPLICIT"), "EDGE_WEIGHT_FORMAT:" (= "LOWER_DIAG_ROW"), "TAG_DISPLAY_DATA_TYPE" (= "TWOD_DISPLAY"), "EDGE_WEIGHT_SECTION", matrix entries, DISPLAY_DATA_SECTION, coordinates, "EOF", 8) "NAME:", "TYPE:" (= "TSP"), "COMMENT:", "DIMENSION:", "EDGE_WEIGHT_TYPE:" (= "EXPLICIT"), "EDGE_WEIGHT_FORMAT:" (= "UPPER_ROW"), "EDGE_WEIGHT_SECTION", matrix entries, "EOF", 9) "NAME:", "TYPE:" (= "TSP"), "COMMENT:", "DIMENSION:", "EDGE_WEIGHT_TYPE:" (= "EXPLICIT"), "EDGE_WEIGHT_FORMAT:" (= "UPPER_ROW"), "TAG_DISPLAY_DATA_TYPE" (= "TWOD_DISPLAY"), "EDGE_WEIGHT_SECTION", matrix entries, DISPLAY_DATA_SECTION, coordinates, "EOF", 10) "NAME:", "TYPE:" (= "ATSP"), "COMMENT:", "DIMENSION:", "EDGE_WEIGHT_TYPE:" (= "EXPLICIT"), "EDGE_WEIGHT_FORMAT:" (= "FULL_MATRIX"), "NODE_COORD_SECTION", matrix entries, "EOF".

Parameters

inputFileName	Name of the input file.

Returns

Instance of the class TransformInstance.

4.2.2.2 double roundToDoublePrecisionAndDoubleFloatField (double *d*) [inline]

Rounds the double to the TRANSFORM_DOUBLE_PRECISION. The floatfield is set to DOUBLE_FLOATFIELD (see IOConstantsBasicFunctionsAndBasicClasses.hpp).

Parameters

d	Number which should be rounded.

Returns

Rounded number d.

Definition at line 299 of file TransformConstantsClassesAndFunctions.hpp.

References DOUBLE_FLOATFIELD, and TRANSFORM_DOUBLE_PRECISION.

Referenced by TransformInstance::setAdjacencyMatrixElement().

4.2.2.3 void trim (std::string & *s*, const std::string & *t* = " \t\r\n") [inline]

Trims the string from left and from right.

Parameters

s	The string.
t	The trimmed characters.

Definition at line 288 of file TransformConstantsClassesAndFunctions.hpp.

References trimLeft(), and trimRight().

Referenced by SAX2ContentHandler::characters(), SAX2ContentHandler::endElement(), readInputFileTSPLIB(), and SAX2ContentHandler::startElement().

4.2.2.4 void trimLeft (std::string & *s*, const std::string & *t* = " \t\r\n") [inline]

Trims the string from left.

Parameters

s	The string.
t	The trimmed characters.

Definition at line 264 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by trim().

4.2.2.5 void trimRight (std::string & *s*, const std::string & *t* = " \t\r\n") [inline]

Trims the string from right.

Parameters

s	The string.
t	The trimmed characters.

Definition at line 273 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by trim().

4.2.2.6 void writeOutputFile (const std::string & *outputFileName*, const TransformInstance * *transformInstance*)

Writes one instance of the class TransformInstance to an output file. The parameters are not checked.

Parameters

outputFile-Name	Name of the output file.
transform-Instance	Instance of the class TransformInstance.

4.2.2.7 void writeOutputFileWithoutUsingAParser (const std::string & *outputFileName*, const TransformInstance * *transformInstance*)

Writes one instance of the class TransformInstance to an output file without using a parser. This method is quicker than the previous one but it provides no parser guarantee that the output file is a valid xml file. The parameters are not checked.

Parameters

outputFile-Name	Name of the output file.
transform-Instance	Instance of the class TransformInstance.

Definition at line 1525 of file TransformConstantsClassesAndFunctions.cpp.

References DOUBLE_FLOATFIELD, TransformInstance::getAdjacencyMatrixElement(), TransformInstance::getDescription(), TransformInstance::getN(), TransformInstance-::getName(), TransformInstance::getType(), TRANSFORM_DOUBLE_PRECISION, TRANSFORM_IGNORED_DIGITS, and VALUE_TYPE_ATSP.

Referenced by main().

4.2.3 Variable Documentation

4.2.3.1 const std::string DISPLAY_DATA_SECTION = "DISPLAY_DATA_SECTION"

Tag for the TSPLIB: "DISPLAY_DATA_SECTION".

Definition at line 82 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.2 const std::ios::fmtflags DOUBLE_FLOATFIELD = std::ios::scientific

Floatfield flag of doubles writed to stringstreams.

Definition at line 239 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by roundToDoublePrecisionAndDoubleFloatField(), writeOutputFile(), and writeOutputFileWithoutUsingAParser().

4.2.3.3 const std::string EDGE_WEIGHT_SECTION = "EDGE_WEIGHT_SECTION"

Tag for the TSPLIB: "EDGE_WEIGHT_SECTION".

Definition at line 77 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.4 const std::string INPUT_FILE_FILENAME_EXTENSION_ATSP = ".atsp"

Filename extension for the asymmetric travelling salesman problem files from the TSPLIB.

Definition at line 32 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by main(), and readInputFileTSPLIB().

4.2.3.5 const std::string INPUT_FILE_FILENAME_EXTENSION_TSP = ".tsp"

Filename extension for the symmetric travelling salesman problem files from the TSPLIB.

Definition at line 27 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by main(), and readInputFileTSPLIB().

4.2.3.6 const std::string OUTPUT_FILE_FILENAME_EXTENSION = ".xml"

Filename extension for xml files.

Definition at line 168 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by main().

4.2.3.7 const double RRR = 6378.388

Idealized sphere radius of the earth.

Definition at line 102 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.8 const std::string TAG_COMMENT = "COMMENT:"

Tag for the TSPLIB: "COMMENT:".

Definition at line 47 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.9 const std::string TAG_DIMENSION = "DIMENSION:"

Tag for the TSPLIB: "DIMENSION:".

Definition at line 52 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.10 const std::string TAG_DISPLAY_DATA_TYPE = "DISPLAY_DATA_TYPE:"

Tag for the TSPLIB: "DISPLAY_DATA_TYPE:".

Definition at line 67 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.11 const std::string TAG_EDGE_WEIGHT_FORMAT = "EDGE_WEIGHT_FORMAT:"

Tag for the TSPLIB: "EDGE_WEIGHT_FORMAT:".

Definition at line 62 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.12 const std::string TAG_EDGE_WEIGHT_TYPE = "EDGE_WEIGHT_TYPE:"

Tag for the TSPLIB: "EDGE_WEIGHT_TYPE:".

Definition at line 57 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.13 const std::string TAG_EOF = "EOF"

Tag for the TSPLIB: "EOF:".

Definition at line 87 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.14 const std::string TAG_NAME = "NAME:"

Tag for the TSPLIB: "NAME":.

Definition at line 37 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.15 const std::string TAG_NODE_COORD_SECTION = "NODE_COORD_SECTION"

Tag for the TSPLIB: "NODE_COORD_SECTION".

Definition at line 72 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.16 const std::string TAG_TYPE = "TYPE:"

Tag for the TSPLIB: "TYPE:".

Definition at line 42 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.17 const std::streamsize TRANSFORM_DOUBLE_PRECISION = std::numeric_limits<double>::digits10

Precision of doubles writed to the .xml-files.

Definition at line 244 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by roundToDoublePrecisionAndDoubleFloatField(), writeOutputFile(), and writeOutputFileWithoutUsingAParser().

4.2.3.18 const double TRANSFORM_DOUBLE_ZERO

Initial value:

```
        pow(10.0, -1.0 * static_cast<double>(TRANSFORM_DOUBLE_PRECISION
- TRANSFORM_IGNORED_DIGITS))
```

Maximum of an absolute value considered as zero used by "Transform"-classes.

Definition at line 255 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.19 const std::streamsize TRANSFORM_IGNORED_DIGITS = 5

Number of ignored digits of double types. (The deviation of double values can be at most 1e-(DoublePrecision - IgnoredDigits).)

Definition at line 250 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by writeOutputFile(), and writeOutputFileWithoutUsingAParser().

4.2.3.20 const std::string VALUE_DISPLAY_DATA_TYPE_COORD_DISPLAY = "COORD_DISPLAY"

Value "COORD_DISPLAY" for the tag "VALUE_DISPLAY_DATA_TYPE:" for the TSPLIB.

Definition at line 157 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.21 const std::string VALUE_DISPLAY_DATA_TYPE_TWOD_DISPLAY = "TWOD_DISPLAY"

Value "COORD_DISPLAY" for the tag "VALUE_DISPLAY_DATA_TYPE:" for the TSPLIB.

Definition at line 162 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.22 const std::string VALUE_EDGE_WEIGHT_FORMAT_FULL_MATRIX = "FULL_MATRIX"

Value "FUNCTION" for the tag "EDGE_WEIGHT_FORMAX:" for the TSPLIB.

Definition at line 137 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.23 const std::string VALUE_EDGE_WEIGHT_FORMAT_FUNCTION = "FUNCTION"

Value "FUNCTION" for the tag "EDGE_WEIGHT_FORMAT:" for the TSPLIB.

Definition at line 132 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.24 const std::string VALUE_EDGE_WEIGHT_FORMAT_LOWER_DIAG_ROW = "LOWER_DIAG_ROW"

Value "FUNCTION" for the tag "EDGE_WEIGHT_FORMAX:" for the TSPLIB.

Definition at line 142 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.25 const std::string VALUE_EDGE_WEIGHT_FORMAT_UPPER_DIAG_ROW = "UPPER_DIAG_ROW"

Value "FUNCTION" for the tag "EDGE_WEIGHT_FORMAX:" for the TSPLIB.

Definition at line 147 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.26 const std::string VALUE_EDGE_WEIGHT_FORMAT_UPPER_ROW = "UPPER_ROW"

Value "FUNCTION" for the tag "EDGE_WEIGHT_FORMAX:" for the TSPLIB.

Definition at line 152 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.27 const std::string VALUE_EDGE_WEIGHT_TYPE_ATT = "ATT"

Value "GEO" for the tag "EDGE_WEIGHT_TYPE:" for the TSPLIB.

Definition at line 122 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.28 const std::string VALUE_EDGE_WEIGHT_TYPE_CEIL_2D = "CEIL_2D"

Value "CEIL_2D" for the tag "EDGE_WEIGHT_TYPE:" for the TSPLIB.

Definition at line 117 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.29 const std::string VALUE_EDGE_WEIGHT_TYPE_EUC_2D = "EUC_2D"

Value "EUC_2D" for the tag "EDGE_WEIGHT_TYPE:" for the TSPLIB.

Definition at line 112 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.30 const std::string VALUE_EDGE_WEIGHT_TYPE_EXPLICIT = "EXPLICIT"

Value "EXPLICIT" for the tag "EDGE_WEIGHT_TYPE:" for the TSPLIB.

Definition at line 127 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.31 const std::string VALUE_EDGE_WEIGHT_TYPE_GEO = "GEO"

Value "GEO" for the tag "EDGE_WEIGHT_TYPE:" for the TSPLIB.

Definition at line 107 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.32 const std::string VALUE_TYPE_ATSP = "ATSP"

Value for the tag "TYPE:" for the asymmetric travelling salesman problem.

Definition at line 97 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB(), writeOutputFile(), and writeOutputFileWithoutUsingAParser().

4.2.3.33 const std::string VALUE_TYPE_TSP = "TSP"

Value for the tag "TYPE:" for the symmetric travelling salesman problem.

Definition at line 92 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by TransformInstance::getAdjacencyMatrixElement(), readInputFileTSPLIB(), TransformInstance::setAdjacencyMatrixElement(), and TransformInstance::TransformInstance().

4.2.3.34 const std::string XML_DESCRIPTION = "description"

Xml tag: Description of the instance.

Definition at line 193 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by SAX2ContentHandler::endElement(), and writeOutputFile().

4.2.3.35 const std::string XML_DOCUMENT_NODE = "travellingSalesmanProblemInstance"

Xml tag: Name of the document node.

Definition at line 178 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by writeOutputFile().

4.2.3.36 const std::string XML_DOUBLE_PRECISION = "doublePrecision"

Precision of doubles.

Definition at line 198 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by SAX2ContentHandler::endElement(), and writeOutputFile().

4.2.3.37 const std::string XML_EDGE = "edge"

Xml tag: One edge

Definition at line 219 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by SAX2ContentHandler::endElement(), SAX2ContentHandler::startElement(), and writeOutputFile().

4.2.3.38 const std::string XML_EDGE_ATTRIBUTE_COST = "cost"

Xml tag - attribute: Cost of the edge.

Definition at line 224 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by writeOutputFile().

4.2.3.39 const std::string XML_ENCODING = "UTF-8"

Encoding of the xml file

Definition at line 229 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by writeOutputFile().

4.2.3.40 const bool XML_FORMAT_PRETTY_PRINT = true

The output will be pretty printed

Definition at line 173 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by writeOutputFile().

4.2.3.41 const std::string XML_GRAPH = "graph"

Xml tag: Underlying graph.

Definition at line 209 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by SAX2ContentHandler::endElement(), and writeOutputFile().

4.2.3.42 const std::string XML_IGNORED_DIGITS = "ignoredDigits"

Number of ignored digits of double types. (The deviation of double values can be at most 1e-(DoublePrecision - IgnoredDigits).)

Definition at line 204 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by SAX2ContentHandler::endElement(), and writeOutputFile().

4.2.3.43 const std::string XML_NAME = "name"

Xml tag: Name of the instance.

Definition at line 183 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by SAX2ContentHandler::endElement(), and writeOutputFile().

4.2.3.44 const std::string XML_SOURCE = "source"

Xml tag: Source of the instance.

Definition at line 188 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by SAX2ContentHandler::endElement(), and writeOutputFile().

4.2.3.45 const std::string XML_VALUE_SOURCE_TSPLIB = "TSPLIB"

Value "TSPLIB" for the xml tag "source"

Definition at line 234 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by readInputFileTSPLIB().

4.2.3.46 const std::string XML_VERTEX = "vertex"

Xml tag: One vertex.

Definition at line 214 of file TransformConstantsClassesAndFunctions.hpp.

Referenced by SAX2ContentHandler::endElement(), SAX2ContentHandler::startElement(), and writeOutputFile().

4.3 src/TransformDOMErrorHandler.cpp File Reference

Defines the tag names in the xml structur, filename extensions etc.

```
#include <xercesc/dom/DOMError.hpp> #include "TransformDOMErrorHandler.hpp"
```

4.3.1 Detailed Description

Defines the tag names in the xml structur, filename extensions etc. Defines the Error-Handler for the DOM parser.

Author

Ulrich Pferschy and Rostislav Stanek (Institut fuer Statistik und Operations - Research, Universitaet Graz)

Definition in file TransformDOMErrorHandler.cpp.

4.4 src/TransformDOMErrorHandler.hpp File Reference

Defines the ErrorHandler for the DOM parser.

```
#include <xercesc/dom/DOMErrorHandler.hpp>
```

Classes

- class TransformDOMErrorHandler

4.4.1 Detailed Description

Defines the ErrorHandler for the DOM parser. Defines the ErrorHandler for the DOM parser.

Author

Ulrich Pferschy and Rostislav Stanek (Institut fuer Statistik und Operations - Research, Universitaet Graz)

Definition in file TransformDOMErrorHandler.hpp.

4.5 src/TransformTSPLIB.cpp File Reference

Transforms the instances of the TSPLIB to the xml structure.

```
#include <iostream> #include <stdexcept> #include <string> ×
#include <algorithm> #include <fstream> #include <sstream> ×
#include <xercesc/util/XMLException.hpp> #include <xercesc/dom/-
DOMException.hpp>     #include <xercesc/util/OutOfMemory-
Exception.hpp>    #include "TransformConstantsClassesAnd-
Functions.hpp"
```

Functions

- int main (int argc, char *argv[])

Variables

- XERCES_CPP_NAMESPACE_USE const vector< vector< double > >::size_type N_THRESHOLD = 4000

4.5.1 Detailed Description

Transforms the instances of the TSPLIB to the xml structure. Transforms the instances of the TSPLIB to the xml structure. The instance has to be a symmetric or asymmetric travelling salesman problem instance. Every keyword must be on a new line and is allowed to be used only once. Note that the symbol ":" has to follow the keywords immediately. The order possibilities of the input keywords are: 1) "NAME:", "TYPE:" (= "TSP"), "COMMENT:", "DIMENSION:", "EDGE_WEIGHT_TYPE:" (= "GEO"), ("EDGE_WEIGHT_FORMAT:" (= "FUNCTION"),) DISPLAY_DATA_TYPE:" (= "COORD_DISPLAY"), "NODE_COORD_SECTION", coordinates, "EOF", 2) "NAME:", "TYPE:", "COMMENT:", "DIMENSION:", "EDGE_WEIGHT_TYPE:" (= "EUC_2D" / "CEIL_2D"), "NODE_COORD_SECTION", coordinates, "EOF", 3) "NAME:", "TYPE:", "COMMENT:", "DIMENSION:", "EDGE_WEIGHT_TYPE:" (= "ATT"), "NODE_COORD_SECTION", coordinates, "EOF", 4) "NAME:", "TYPE:" (= "TSP"), "COMMENT:", "DIMENSION:", "EDGE_WEIGHT_TYPE:" (= "EXPLICIT"), "EDGE_WEIGHT_FORMAT:" (= "FULL_MATRIX"), "EDGE_WEIGHT_SECTION", matrix entries, "EOF", 5) "NAME:", "TYPE:" (= "TSP"), "COMMENT:", "DIMENSION:", "EDGE_WEIGHT_TYPE:" (= "EXPLICIT"), "EDGE_WEIGHT_FORMAT:" (= "FULL_MATRIX"), "TAG_DISPLAY_DATA_TYPE" (= "TWOD_DISPLAY"), "EDGE_WEIGHT_SECTION", matrix entries, DISPLAY_DATA_SECTION, coordinates, "EOF", 6) "NAME:", "TYPE:" (= "TSP"), "COMMENT:", "DIMENSION:", "EDGE_WEIGHT_TYPE:" (= "EXPLICIT"), "EDGE_WEIGHT_FORMAT:" (= "LOWER_DIAG_ROW"), "EDGE_WEIGHT_SECTION", matrix entries, "EOF", 7) "NAME:", "TYPE:" (= "TSP"), "COMMENT:", "DIMENSION:", "EDGE_WEIGHT_TYPE:" (= "EXPLICIT"), "EDGE_WEIGHT_FORMAT:" (= "LOWER_DIAG_ROW"), "TAG_DISPLAY_DATA_TYPE" (= "TWOD_DISPLAY"), "EDGE_WEIGHT_SECTION", matrix entries, DISPLAY_DATA_SECTION, coordinates, "EOF", 8) "NAME:", "TYPE:" (= "TSP"), "COMMENT:", "DIMENSION:", "EDGE_WEIGHT_TYPE:" (= "EXPLICIT"), "EDGE_WEIGHT_FORMAT:" (= "UPPER_ROW"), "EDGE_WEIGHT_SECTION", matrix entries, "EOF", 9) "NAME:", "TYPE:" (= "TSP"), "COMMENT:", "DIMENSION:", "EDGE_WEIGHT_TYPE:" (= "EXPLICIT"), "EDGE_WEIGHT_FORMAT:" (= "UPPER_ROW"), "TAG_DISPLAY_DATA_TYPE" (= "TWOD_DISPLAY"), "EDGE_WEIGHT_SECTION", matrix entries, DISPLAY_DATA_SECTION, coordinates, "EOF", 10) "NAME:", "TYPE:" (= "ATSP"), "COMMENT:", "DIMENSION:", "EDGE_WEIGHT_TYPE:" (= "EXPLICIT"), "EDGE_WEIGHT_FORMAT:" (= "FULL_MATRIX"), "NODE_COORD_SECTION", matrix entries, "EOF". The program uses 2 different strategies to create the XML file. The choice which one will be used is determined by the value of the N_THRESHOLD constant.

Author

Ulrich Pferschy and Rostislav Stanek (Institut fuer Statistik und Operations - Research, Universitaet Graz)

Definition in file TransformTSPLIB.cpp.

4.5.2 Function Documentation

4.5.2.1 int main (int *argc*, char * *argv[]*)

The main function.

Parameters

argc	Number of elements in the array argv, in particular 1 or 4.
argv	Name of the running program or name of the running program, the name of the input file and the name of the output file.

Returns

- 0 if succeeded,
- 1 otherwise.

Definition at line 95 of file TransformTSPLIB.cpp.

References TransformInstance::getN(), INPUT_FILE_FILENAME_EXTENSION_ATSP, INPUT_FILE_FILENAME_EXTENSION_TSP, N_THRESHOLD, OUTPUT_FILE_FILENAME_EXTENSION, readInputFileTSPLIB(), writeOutputFile(), and writeOutputFileWithoutUsingAParser().

4.5.3 Variable Documentation

4.5.3.1 XERCES_CPP_NAMESPACE_USE const vector<vector<double> >::size_type N_THRESHOLD = 4000

A threshold value of the size of the graph (i.e. of the number of vertices) which determines if the program uses the safe method to create the xml file or the quicker method which provides no guarantee that the output file is a valid XML file.

Definition at line 81 of file TransformTSPLIB.cpp.

Referenced by main().

4.6 src/Validate.cpp File Reference

Validates an travelling salesman problem instance.

```
#include <iostream> #include <xercesc/util/XMLException.-
hpp>     #include <xercesc/util/OutOfMemoryException.hpp>
#include "ValidateConstantsFunctionsAndClasses.hpp" #include
"ValidateIO.hpp" #include "ValidateInstance.hpp"
```

Functions

- XERCES_CPP_NAMESPACE_USE int main (int argc, char ∗argv[])

4.6.1 Detailed Description

Validates an travelling salesman problem instance. Validates an travelling salesman problem instance.

Author

Ulrich Pferschy and Rostislav Stanek (Institut fuer Statistik und Operations - Research, Universitaet Graz)

Definition in file Validate.cpp.

4.6.2 Function Documentation

4.6.2.1 XERCES_CPP_NAMESPACE_USE int main (int *argc,* char ∗ *argv[]*)

The main function.

Parameters

argc	Number of elements in the array argv, in particular 1 or 2.
argv	Name of the running program or name of the running program, name of the input file.

Returns

- 0 if succeeded,
- 1 otherwise.

Definition at line 37 of file Validate.cpp.

References Instance::getDescription(), Instance::getGraph(), Graph::getIsUndirected(), Graph::getN(), Instance::getName(), Instance::getSource(), instanceIn(), and parseCommandLineArguments().

4.7 src/ValidateConstantsFunctionsAndClasses.hpp File Reference

Defines the constants, some basic functions and some basic classes necessary for the IO.

```
#include <limits> #include <fstream> #include <sstream> ×
#include <iomanip> #include <string>
```

Classes

- class CommandLineArgumentsInvalid
- class ValidationSchemaDoesNotExist
- class ValidationFailed

Functions

- void trimLeft (std::string &s, const std::string &t=" \t\r\n")
- void trimRight (std::string &s, const std::string &t=" \t\r\n")
- void trim (std::string &s, const std::string &t=" \t\r\n")

Variables

- const std::string INPUT_FILE_FILENAME_EXTENSION = ".xml"
- const std::string VALIDATION_SCHEMA = "TSPConfiguration.xsd"
- const std::string XML_ENCODING = "UTF-8"
- const std::ios::fmtflags DOUBLE_FLOATFIELD = std::ios::scientific
- const std::string XML_DOCUMENT_NODE = "travellingSalesmanProblem-Instance"
- const std::string XML_NAME = "name"
- const std::string XML_SOURCE = "source"
- const std::string XML_DESCRIPTION = "description"
- const std::string XML_DOUBLE_PRECISION = "doublePrecision"
- const std::string XML_IGNORED_DIGITS = "ignoredDigits"
- const std::string XML_GRAPH = "graph"
- const std::string XML_VERTEX = "vertex"
- const std::string XML_EDGE = "edge"
- const std::string XML_EDGE_ATTRIBUTE_COST = "cost"

4.7.1 Detailed Description

Defines the constants, some basic functions and some basic classes necessary for the IO. Defines the constants, some basic functions and some basic functions necessary for the IO.

Author

Ulrich Pferschy and Rostislav Stanek (Institut fuer Statistik und Operations - Research, Universitaet Graz)

Definition in file ValidateConstantsFunctionsAndClasses.hpp.

4.7.2 Function Documentation

4.7.2.1 void trim (std::string & *s*, const std::string & *t* = " \t\r\n") [inline]

Trims the string from left and from right.

Parameters

s	The string.
t	The trimmed characters.

Definition at line 124 of file ValidateConstantsFunctionsAndClasses.hpp.

References trimLeft(), and trimRight().

4.7.2.2 void trimLeft (std::string & *s*, const std::string & *t* = " \t\r\n") [inline]

Trims the string from left.

Parameters

s	The string.
t	The trimmed characters.

Definition at line 100 of file ValidateConstantsFunctionsAndClasses.hpp.

4.7.2.3 void trimRight (std::string & *s*, const std::string & *t* = " \t\r\n") [inline]

Trims the string from right.

Parameters

s	The string.
t	The trimmed characters.

Definition at line 109 of file ValidateConstantsFunctionsAndClasses.hpp.

4.7.3 Variable Documentation

4.7.3.1 const std::ios::fmtflags DOUBLE_FLOATFIELD = std::ios::scientific

Floatfield flag of doubles writed to stringstreams.

Definition at line 41 of file ValidateConstantsFunctionsAndClasses.hpp.

4.7.3.2 const std::string INPUT_FILE_FILENAME_EXTENSION = ".xml"

Filename extension for xml files.

Definition at line 26 of file ValidateConstantsFunctionsAndClasses.hpp.

Referenced by parseCommandLineArguments().

4.7.3.3 const std::string VALIDATION_SCHEMA = "TSPConfiguration.xsd"

Validation schema for the xml files.

Definition at line 31 of file ValidateConstantsFunctionsAndClasses.hpp.

Referenced by instanceIn().

4.7.3.4 const std::string XML_DESCRIPTION = "description"

Xml tag: Description of the instance.

Definition at line 61 of file ValidateConstantsFunctionsAndClasses.hpp.

4.7.3.5 const std::string XML_DOCUMENT_NODE = "travellingSalesmanProblemInstance"

Xml tag: Name of the document node.

Definition at line 46 of file ValidateConstantsFunctionsAndClasses.hpp.

4.7.3.6 const std::string XML_DOUBLE_PRECISION = "doublePrecision"

Precision of doubles.

Definition at line 66 of file ValidateConstantsFunctionsAndClasses.hpp.

4.7.3.7 const std::string XML_EDGE = "edge"

Xml tag: One edge

Definition at line 87 of file ValidateConstantsFunctionsAndClasses.hpp.

4.7.3.8 const std::string XML_EDGE_ATTRIBUTE_COST = "cost"

Xml tag - attribute: Cost of the edge.

Definition at line 92 of file ValidateConstantsFunctionsAndClasses.hpp.

4.7.3.9 const std::string XML_ENCODING = "UTF-8"

Encoding of the xml file

Definition at line 36 of file ValidateConstantsFunctionsAndClasses.hpp.

4.7.3.10 const std::string XML_GRAPH = "graph"

Xml tag: Underlying graph.

Definition at line 77 of file ValidateConstantsFunctionsAndClasses.hpp.

4.7.3.11 const std::string XML_IGNORED_DIGITS = "ignoredDigits"

Number of ignored digits of double types. (The deviation of double values can be at most 1e-(DoublePrecision - IgnoredDigits).)

Definition at line 72 of file ValidateConstantsFunctionsAndClasses.hpp.

4.7.3.12 const std::string XML_NAME = "name"

Xml tag: Name of the instance.

Definition at line 51 of file ValidateConstantsFunctionsAndClasses.hpp.

4.7.3.13 const std::string XML_SOURCE = "source"

Xml tag: Source of the instance.

Definition at line 56 of file ValidateConstantsFunctionsAndClasses.hpp.

4.7.3.14 const std::string XML_VERTEX = "vertex"

Xml tag: One vertex.

Definition at line 82 of file ValidateConstantsFunctionsAndClasses.hpp.

4.8 src/ValidateGraph.cpp File Reference

Defines the class Graph.

```
#include "ValidateGraph.hpp"
```

Functions

- std::ostream & operator<< (std::ostream &os, Graph &graph)

4.8.1 Detailed Description

Defines the class Graph. Defines the class Graph which saves one weighted complete graph. The defined class provides no checks of ranges.

Author

Ulrich Pferschy and Rostislav Stanek (Institut fuer Statistik und Operations - Research, Universitaet Graz)

Definition in file ValidateGraph.cpp.

4.8.2 Function Documentation

4.8.2.1 std::ostream& operator<< (std::ostream & *os,* Graph & *graph*)

Implements the operator "<<".

Parameters

os	Ostream.
graph	Graph.

Returns

Ostream with the information about the graph instance.

Definition at line 117 of file ValidateGraph.cpp.

References Graph::getEdgeCost(), Graph::getIsUndirected(), and Graph::getN().

4.9 src/ValidateGraph.hpp File Reference

Defines the class Graph.

```
#include <vector> #include <sstream>
```

Classes

- class GraphNotValid
- class Graph

4.9.1 Detailed Description

Defines the class Graph. Defines the class Graph which saves one weighted complete graph. The defined class provides no checks of ranges.

Author

Ulrich Pferschy and Rostislav Stanek (Institut fuer Statistik und Operations - Research, Universitaet Graz)

Definition in file ValidateGraph.hpp.

4.10 src/ValidateInstance.cpp File Reference

Defines the class Instance.

```
#include <cmath> #include "ValidateInstance.hpp"
```

Functions

- std::ostream & operator<< (std::ostream &os, Instance &instance)

4.10.1 Detailed Description

Defines the class Instance. Defines the class Instance which saves one instance of the travelling salesman problem with the underlying graph in the form of a weighted complete graph. The defined class provides no checks of ranges.

Author

Ulrich Pferschy and Rostislav Stanek (Institut fuer Statistik und Operations - Research, Universitaet Graz)

Definition in file ValidateInstance.cpp.

4.10.2 Function Documentation

4.10.2.1 std::ostream& operator<< (std::ostream & *os,* Instance & *instance*)

Implements the operator "<<".

Parameters

os	Ostream.
instance	Instance.

Returns

Ostream with the information about the instance instance.

Definition at line 107 of file ValidateInstance.cpp.

References Instance::getDescription(), Instance::getDoublePrecision(), Instance::getDoubleZero(), Instance::getGraph(), Instance::getIgnoredDigits(), Instance::getName(), and Instance::getSource().

4.11 src/ValidateInstance.hpp File Reference

Defines the class Instance.

```
#include <ios>  #include <sstream>  #include <string> ×
#include "ValidateGraph.hpp"
```

Classes

- class Instance

4.11.1 Detailed Description

Defines the class Instance. Defines the class Instance which saves one instance of the travelling salesman problem with the underlying graph in the form of a weighted complete graph. The defined class provides no checks of ranges.

Author

Ulrich Pferschy and Rostislav Stanek (Institut fuer Statistik und Operations - Research, Universitaet Graz)

Definition in file ValidateInstance.hpp.

4.12 src/ValidateIO.cpp File Reference

Defines the functions necessary for the IO.

```
#include <iostream> #include <algorithm> #include <cmath> ×
#include <xercesc/util/PlatformUtils.hpp> #include <xercesc/util/-
XMLString.hpp>  #include <xercesc/sax2/SAX2XMLReader.-
hpp> #include <xercesc/sax2/XMLReaderFactory.hpp> #include
<xercesc/util/OutOfMemoryException.hpp> #include "Validate-
ConstantsFunctionsAndClasses.hpp" #include "ValidateSAX2-
ErrorHandler.hpp"  #include "ValidateSAX2ContentHandler.-
hpp" #include "ValidateIO.hpp"
```

Classes

- class XMLStringTranscode

Defines

- #define unicodeForm(str) XMLStringTranscode(str).unicodeForm()

Functions

- XERCES_CPP_NAMESPACE_USE std::string parseCommandLineArguments (int argc, char ∗argv[])
- Instance ∗ instanceIn (const std::string &inputFileName)

4.12.1 Detailed Description

Defines the functions necessary for the IO. Defines the functions and classes necessary for the IO.

Author

Ulrich Pferschy and Rostislav Stanek (Institut fuer Statistik und Operations - Research, Universitaet Graz)

Definition in file ValidateIO.cpp.

4.12.2 Define Documentation

4.12.2.1 #define unicodeForm(*str*) XMLStringTranscode(str).unicodeForm()

Definition at line 119 of file ValidateIO.cpp.

4.12.3 Function Documentation

4.12.3.1 Instance∗ instanceIn (const std::string & *inputFileName*)

Reads one travelling salesman problem instance from the file inputFileName.

Parameters

inputFile-Name	Name of the input file.

Returns

Instance of the travelling salesman problem.

Definition at line 121 of file ValidateIO.cpp.

References SAX2ContentHandler::getAdjacencyMatrix(), SAX2ContentHandler::getDescription(), SAX2ContentHandler::getDoublePrecision(), SAX2ErrorHandler::getFailed(), SAX2ContentHandler::getFailed(), SAX2ContentHandler::getIgnoredDigits(), SAX2ContentHandler::getName(), SAX2ContentHandler::getSource(), and VALIDATION_SCHEMA.

Referenced by main().

4.12.3.2 XERCES_CPP_NAMESPACE_USE std::string parseCommandLineArguments (int *argc*, char * *argv[]*)

Parses the command-line arguments.

Parameters

argc	Number of arguments.
argv	Arguments.

Returns

Name of the input file.

Definition at line 33 of file ValidateIO.cpp.

References INPUT_FILE_FILENAME_EXTENSION.

Referenced by main().

4.13 src/ValidateIO.hpp File Reference

Defines the functions necessary for the IO.

`#include <string> #include <stdexcept> #include "ValidateInstance.hpp" #include "ValidateGraph.hpp"`

Functions

- std::string parseCommandLineArguments (int argc, char *argv[])
- Instance * instanceIn (const std::string &inputFileName)

4.13.1 Detailed Description

Defines the functions necessary for the IO. Defines the functions and classes necessary for the IO.

Author

Ulrich Pferschy and Rostislav Stanek (Institut fuer Statistik und Operations - Research, Universitaet Graz)

Definition in file ValidateIO.hpp.

4.13.2 Function Documentation

4.13.2.1 Instance* instanceIn (const std::string & *inputFileName*)

Reads one travelling salesman problem instance from the file inputFileName.

Parameters

inputFileName	Name of the input file.

Returns

Instance of the travelling salesman problem.

Definition at line 121 of file ValidateIO.cpp.

References SAX2ContentHandler::getAdjacencyMatrix(), SAX2ContentHandler::getDescription(), SAX2ContentHandler::getDoublePrecision(), SAX2ErrorHandler::getFailed(), SAX2ContentHandler::getFailed(), SAX2ContentHandler::getIgnoredDigits(), SAX2ContentHandler::getName(), SAX2ContentHandler::getSource(), and VALIDATION_SCHEMA.

Referenced by main().

4.13.2.2 std::string parseCommandLineArguments (int *argc*, char * *argv[]*)

Parses the command-line arguments.

Parameters

argc	Number of arguments.
argv	Arguments.

Returns

Name of the input file.

Definition at line 33 of file ValidateIO.cpp.

References INPUT_FILE_FILENAME_EXTENSION.

Referenced by main().

4.14 src/ValidateSAX2ContentHandler.cpp File Reference

Defines the ContentHandler for the SAX2 parser.

```
#include <fstream>  #include <sstream>  #include <cmath>
#include <xercesc/util/XMLString.hpp> #include <xercesc/sax2/-
Attributes.hpp>  #include "ValidateConstantsFunctionsAnd-
Classes.hpp" #include "ValidateSAX2ContentHandler.hpp"
```

Classes

- class XMLStringTranscode
- class StringTranscode

Defines

- #define unicodeForm(str) XMLStringTranscode(str).unicodeForm()
- #define stringForm(str) StringTranscode(str).stringForm()

4.14.1 Detailed Description

Defines the ContentHandler for the SAX2 parser. Defines the ContentHandler for the SAX2 parser.

Author

Ulrich Pferschy and Rostislav Stanek (Institut fuer Statistik und Operations - Research, Universitaet Graz)

Definition in file ValidateSAX2ContentHandler.cpp.

4.14.2 Define Documentation

4.14.2.1 #define stringForm(*str*) StringTranscode(str).stringForm()

Definition at line 95 of file ValidateSAX2ContentHandler.cpp.

Referenced by SAX2ContentHandler::characters(), and SAX2ContentHandler::startElement().

4.14.2.2 #define unicodeForm(*str*) XMLStringTranscode(str).unicodeForm()

Definition at line 59 of file ValidateSAX2ContentHandler.cpp.

Referenced by SAX2ContentHandler::endElement(), and SAX2ContentHandler::startElement().

4.15 src/ValidateSAX2ContentHandler.hpp File Reference

Defines the ContentHandler for the SAX2 parser.

```
#include <xercesc/sax2/ContentHandler.hpp> #include <ios> ×
#include <vector> #include <set>
```

Classes

- class SAX2ContentHandler

4.15.1 Detailed Description

Defines the ContentHandler for the SAX2 parser. Defines the ContentHandler for the SAX2 parser.

Author

Ulrich Pferschy and Rostislav Stanek (Institut fuer Statistik und Operations - Research, Universitaet Graz)

Definition in file ValidateSAX2ContentHandler.hpp.

4.16 src/ValidateSAX2ErrorHandler.cpp File Reference

Defines the ErrorHandler for the SAX2 parser.

```
#include "ValidateSAX2ErrorHandler.hpp"
```

4.16.1 Detailed Description

Defines the ErrorHandler for the SAX2 parser. Defines the ErrorHandler for the SAX2 parser.

Author

Ulrich Pferschy and Rostislav Stanek (Institut fuer Statistik und Operations - Research, Universitaet Graz)

Definition in file ValidateSAX2ErrorHandler.cpp.

4.17 src/ValidateSAX2ErrorHandler.hpp File Reference

Defines the ErrorHandler for the SAX2 parser.

```
#include <xercesc/sax/ErrorHandler.hpp>
```

Classes

- class SAX2ErrorHandler

4.17.1 Detailed Description

Defines the ErrorHandler for the SAX2 parser. Defines the ErrorHandler for the SAX2 parser.

Author

Ulrich Pferschy and Rostislav Stanek (Institut fuer Statistik und Operations - Research, Universitaet Graz)

Definition in file ValidateSAX2ErrorHandler.hpp.