Algotester lib

1 Class Index

1.1 Class List

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

2 Class Documentation

2.1 AlgotesterGenerator Class Reference

Public Member Functions

- **AlgotesterGenerator** (unsigned long long seed=0)
- **AlgotesterGenerator** (int argc, char ∗∗argv)
- unsigned long long randUInt ()
  Generate uniform random unsigned long long.
- unsigned long long randUInt (unsigned long long n)
  Generate uniform random unsigned long long in range [0, n)
- unsigned long long randUInt (unsigned long long l, unsigned long long r)
  Generate uniform random unsigned long long in range [l, r].
- long long randInt ()
  Generate uniform random long long.
- long long randInt (long long n)
  Generate uniform random long long in range [0, n)
- long long randInt (long long l, long long r)
  Generate uniform random long long in range [l, r].
- long double randDouble ()
  Generate random long double in range [0, 1].
- long double randDouble (long double n)
  Generate random long double in range [0, n].
- long double randDouble (long double l, long double r)
  Generate random long double in range [l, r].
- template<class T >
  void shuffle (std::vector< T > &A)

Random shuffle vector A.

- template<class RandomAccessIterator >
  void shuffle (RandomAccessIterator first, RandomAccessIterator last)

  Random shuffle the elements in the range [first,last)
- template<class T >
  T getArg (int index)

  Get argument.

2.1.1 Member Function Documentation

2.1.1.1 getArg() `template<class T >`

```
T AlgotesterGenerator::getArg (
            int index )  [inline]
```

Get argument.

Template Parameters

T	Type of the argument.

Parameters

index	Argument index(1-based).

2.1.1.2 randUInt() `unsigned long long AlgotesterGenerator::randUInt (`

```
            unsigned long long n )  [inline]
```

Generate uniform random unsigned long long in range [0, n)

Note

To make random number uniform several tries could be made. Expected value of number of tries is less than 2.

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

- algotester_generator.h

2.2 AlgotesterReader Class Reference

Public Member Functions

- **AlgotesterReader** (istream *in=&cin, bool strict=true)
- **AlgotesterReader** (string filename, bool strict=true)
- char readChar (string values="", bool skipWhitespace=false, string name="")

  Read single char and check that its value is among possible values.
- void readSpace ()

  Read space character.
- void readEndl ()

  Read endline character.
- void readEof ()

  Read end of file character.
- bool isEof (bool skipWhitespace=false)

  Check if it is end of file.
- long long readInt (string name="")

  Read long long.
- long long readInt (long long l, long long r, string name="")

  Read long long and check if it's in given range.
- vector< long long > readVectorInt (int n, string name="")

  Read vector of long long with given length.
- vector< long long > readVectorInt (int n, long long l, long long r, string name="")

  Read vector of long long with given length and check that each element is in given range.
- unsigned long long readUInt (string name="")

  Read unsigned long long.
- unsigned long long readUInt (unsigned long long l, unsigned long long r, string name="")

  Read unsigned long long and check if it's in given range.
- string readToken (string name="", string regex=".*")

  Read token till the next whitespace and check if it matches given regex.
- string readToken (int minLength, int maxLength, string name="", string regex=".*")

  Read token till the next whitespace and check if it matches given regex.
- string readLine ()

  Read line till the next newline character.
- long double readDouble (string name="")

  Read long double.
- long double readDouble (long double l, long double r, string name="")

  Read long double and check if it's in given range.
- long double readDouble (int minDecimals, int maxDecimals, bool allowExpFormat, string name="")

  Read long double and check its characteristics.
- long double readDouble (long double l, long double r, int minDecimals, int maxDecimals, bool allowExp↩ Format, string name="")

  Read long double and check its characteristics and range.

2.2.1 Member Function Documentation

2.2.1.1 isEof() `bool AlgotesterReader::isEof ( bool skipWhitespace = false ) [inline]`

Check if it is end of file.

Parameters

skipWhitespace	If true, whitespaces are skipped, i.e. result of this function is true if only whitespaces are left.

Returns

If the reader reached end of file.

Note

`skipWhitespace=true` could not be used if `strict=true`

2.2.1.2 readChar()

```
char AlgotesterReader::readChar (
        string values = "",
        bool skipWhitespace = false,
        string name = "" )  [inline]
```

Read single char and check that its value is among possible values.

Parameters

values	If non-empty, string that contain all possible char values.
skipWhitespace	Whether to skip whitespaces before reading next character.
name	Name of the character.

Returns

Value that was read.

Note

`skipWhitespace=true` could not be used if `strict=true`

2.2.1.3 readDouble() [1/4]

```
long double AlgotesterReader::readDouble (
        int minDecimals,
        int maxDecimals,
        bool allowExpFormat,
        string name = "" )  [inline]
```

Read long double and check its characteristics.

Parameters

minDecimals	Minimal number of decimal digits after the point.
maxDecimals	Maximal number of decimal digits after the point.
allowExpFormat	Whether exponential format is allowed.
name	Name of the variable.

Returns

Value that was read.

2.2.1.4 readDouble() [2/4]

```
long double AlgotesterReader::readDouble (
        long double l,
        long double r,
        int minDecimals,
        int maxDecimals,
        bool allowExpFormat,
        string name = "" )  [inline]
```

Read long double and check its characteristics and range.

Parameters

l	Minimal value.
r	Maximal value.
minDecimals	Minimal number of decimal digits after the point.
maxDecimals	Maximal number of decimal digits after the point.
allowExpFormat	Whether exponential format is allowed.
name	Name of the variable.

Returns

Value that was read.

2.2.1.5 readDouble() [3/4]

```
long double AlgotesterReader::readDouble (
        long double l,
        long double r,
        string name = "" )  [inline]
```

Read long double and check if it's in given range.

Parameters

l	Minimal value.
r	Maximal value.
name	Name of the variable.

Returns

Value that was read.

2.2.1.6 readDouble() [4/4] long double AlgotesterReader::readDouble (string *name* = "") [inline]

Read long double.

Parameters

name	Name of the variable.

Returns

Value that was read.

2.2.1.7 readInt() [1/2] long long AlgotesterReader::readInt (long long *l*, long long *r*, string *name* = "") [inline]

Read long long and check if it's in given range.

Parameters

l	Minimal value of integer.
r	Maximal value of integer.
name	Name of the variable.

Returns

Value that was read.

2.2.1.8 readInt() [2/2] long long AlgotesterReader::readInt (string *name* = "") [inline]

Read long long.

Parameters

name	Name of the variable.

Returns

Value that was read.

2.2.1.9 readLine() `string AlgotesterReader::readLine ( ) [inline]`

Read line till the next newline character.

Returns

Line that was read.

2.2.1.10 readToken() [1/2] `string AlgotesterReader::readToken (`

```
int minLength,
int maxLength,
string name = "",
string regex = ".*" ) [inline]
```

Read token till the next whitespace and check if it matches given regex.

Parameters

name	Name of the variable.
regex	Regular expression that token must match.

Returns

Token that was read.

2.2.1.11 readToken() [2/2] `string AlgotesterReader::readToken (`

```
string name = "",
string regex = ".*" ) [inline]
```

Read token till the next whitespace and check if it matches given regex.

Parameters

name	Name of the variable.
regex	Regular expression that token must match.

Returns

Token that was read.

2.2.1.12 readUInt() [1/2] `unsigned long long AlgotesterReader::readUInt (`

```
string name = "" ) [inline]
```

Read unsigned long long.

Parameters

name	Name of the variable.

Returns

Value that was read.

2.2.1.13 readUInt() [2/2] unsigned long long AlgotesterReader::readUInt (

```
unsigned long long l,
unsigned long long r,
string name = "" )  [inline]
```

Read unsigned long long and check if it's in given range.

Parameters

l	Minimal value.
r	Maximal value.
name	Name of the variable.

Returns

Value that was read.

2.2.1.14 readVectorInt() [1/2] vector< long long > AlgotesterReader::readVectorInt (

```
int n,
long long l,
long long r,
string name = "" )  [inline]
```

Read vector of long long with given length and check that each element is in given range.

Parameters

n	Length of vector.
l	Minimal value of element.
r	Maximal value of element.
name	Name

Returns

Values that were read.

Note

In strict mode elements should be space-separated with endl in the end.

2.2.1.15 readVectorInt() [2/2]

```
vector< long long > AlgotesterReader::readVectorInt (
    int n,
    string name = "" )  [inline]
```

Read vector of long long with given length.

Parameters

n	Length of vector.
name	Name

Returns

Values that were read.

Note

In strict mode elements should be space-separated with endl in the end.

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

- algotester.h