

My Project

Generated by Doxygen 1.9.8

1 Class Index	1
1.1 Class List	1
2 File Index	3
2.1 File List	3
3 Class Documentation	5
3.1 Multiset Class Reference	5
3.1.1 Detailed Description	6
3.1.2 Constructor & Destructor Documentation	6
3.1.2.1 Multiset() [1/4]	6
3.1.2.2 Multiset() [2/4]	6
3.1.2.3 Multiset() [3/4]	7
3.1.2.4 Multiset() [4/4]	7
3.1.3 Member Function Documentation	7
3.1.3.1 addAll()	7
3.1.3.2 addElement() [1/2]	7
3.1.3.3 addElement() [2/2]	8
3.1.3.4 addSubset()	8
3.1.3.5 addToRes() [1/2]	8
3.1.3.6 addToRes() [2/2]	8
3.1.3.7 boolean()	9
3.1.3.8 countEls()	9
3.1.3.9 countSubs()	9
3.1.3.10 delElement() [1/2]	10
3.1.3.11 delElement() [2/2]	10
3.1.3.12 delSubset()	10
3.1.3.13 findEls()	10
3.1.3.14 findSubs()	11
3.1.3.15 getCardinality()	11
3.1.3.16 isEmpty()	11
3.1.3.17 operator!=()	11
3.1.3.18 operator*()	12
3.1.3.19 operator*=()	12
3.1.3.20 operator+()	12
3.1.3.21 operator+=()	12
3.1.3.22 operator-()	13
3.1.3.23 operator-=()	13
3.1.3.24 operator==()	13
3.1.3.25 operator[]() [1/3]	14
3.1.3.26 operator[]() [2/3]	14
3.1.3.27 operator[]() [3/3]	14
3.1.3.28 parsStrToSet()	15

3.1.3.29 printSet()	15
3.1.3.30 printSetInfo()	15
3.1.3.31 toString()	15
4 File Documentation	17
4.1 Multitest.h	17
Index	19

Chapter 1

Class Index

1.1 Class List

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

Multiset	
Implementation of the Multiset class	5

Chapter 2

File Index

2.1 File List

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

Multitest.h	17
---------------------------------------	----

Chapter 3

Class Documentation

3.1 Multiset Class Reference

Implementation of the [Multiset](#) class.

```
#include <Multitest.h>
```

Public Member Functions

- void [parsStrToSet](#) (const string &elements)
- string [toString](#) () const
- size_t [countEls](#) (const string &element) const
- size_t [countSubs](#) (const [Multiset](#) &subset) const
- bool [findEls](#) (const string &element) const
- bool [findSubs](#) (const [Multiset](#) &subset) const
- void [addToRes](#) (const string &element, size_t amount)
- void [addToRes](#) (const [Multiset](#) &set, size_t amount)
- void [addAll](#) (const [Multiset](#) &set1, const [Multiset](#) &set2)
- [Multiset](#) ()
- [Multiset](#) (const string &elements)
- [Multiset](#) (const char *elements)
- [Multiset](#) (const [Multiset](#) &set)
- bool [isEmpty](#) () const
- [Multiset](#) boolean () const
- size_t [getCardinality](#) () const
- void [addElement](#) (const string &element)
- void [addElement](#) (const char *element)
- void [addSubset](#) (const [Multiset](#) &set)
- void [delElement](#) (const string &element)
- void [delElement](#) (const char *element)
- void [delSubset](#) (const [Multiset](#) &set)
- bool [operator\[\]](#) (const string &element) const
- bool [operator\[\]](#) (const char *element) const
- bool [operator\[\]](#) (const [Multiset](#) &subset) const
- bool [operator==](#) (const [Multiset](#) &Set) const
- bool [operator!=](#) (const [Multiset](#) &Set) const
- [Multiset](#) [operator+](#) (const [Multiset](#) &Set) const

- `Multiset operator+=` (const `Multiset` &Set)
- `Multiset operator*` (const `Multiset` &Set) const
- `Multiset operator*=` (const `Multiset` &Set)
- `Multiset operator-` (const `Multiset` &Set) const
- `Multiset operator-=` (const `Multiset` &Set)
- void `printSetInfo` () const
- void `printSet` () const

Public Attributes

- vector< string > `elements_`
- vector< `Multiset` > `subsets_`
- size_t `cardinality_`

3.1.1 Detailed Description

Implementation of the `Multiset` class.

Author

Verkovich E. V.

Version

1.0

Date

October 2023

`Multiset` class contains 2 vector fields for storing multiset elements, a field showing the capacity of the set and its maximum nesting; a constructor with an initialization list, with a parameter, as well as a copy constructor; methods for adding and removing elements; a method for determining the emptiness of the set; a method for forming the booleans of the set; overloaded operators "+", "+=", "*", "*=", "-", "-=", "["; other additional methods.

3.1.2 Constructor & Destructor Documentation

3.1.2.1 `Multiset()` [1/4]

```
Multiset::Multiset ( )
```

constructor with initialization list

3.1.2.2 `Multiset()` [2/4]

```
Multiset::Multiset (
    const string & element )
```

constructor with parameter

Parameters

<i>element</i>	
----------------	--

3.1.2.3 Multiset() [3/4]

```
Multiset::Multiset (  
    const char * element )
```

constructor with parameter

Parameters

<i>element</i>	
----------------	--

3.1.2.4 Multiset() [4/4]

```
Multiset::Multiset (  
    const Multiset & set )
```

copy constructor

Parameters

<i>set</i>	
------------	--

3.1.3 Member Function Documentation**3.1.3.1 addAll()**

```
void Multiset::addAll (  
    const Multiset & set1,  
    const Multiset & set2 )
```

subfunction for the "+" operator

Parameters

<i>set1, set2</i>	
-------------------	--

3.1.3.2 addElement() [1/2]

```
void Multiset::addElement (  
    const char * element )
```

adds an element to set

Parameters

<i>element</i>	
----------------	--

3.1.3.3 addElement() [2/2]

```
void Multiset::addElement (
    const string & element )
```

adds an element to set

Parameters

<i>element</i>	
----------------	--

3.1.3.4 addSubset()

```
void Multiset::addSubset (
    const Multiset & set )
```

adds a subset to set

Parameters

<i>set</i>	
------------	--

3.1.3.5 addToRes() [1/2]

```
void Multiset::addToRes (
    const Multiset & set,
    size_t amount )
```

overloaded function [addToRes\(\)](#)

Parameters

<i>set,amount</i>	
-------------------	--

3.1.3.6 addToRes() [2/2]

```
void Multiset::addToRes (
    const string & element,
    size_t amount )
```

adds string element to vector <string> elements amount times

Parameters

<i>element, amount</i>	
------------------------	--

3.1.3.7 boolean()

```
Multiset Multiset::boolean ( ) const
```

a method for forming the booleans of the set

Returns

result boolean set

3.1.3.8 countEls()

```
size_t Multiset::countEls (
    const string & element ) const
```

counts occurrences of a given element in the set

Parameters

<i>element</i>	an element, the occurrences of which are counted
----------------	--

Returns

result of std::count() function

3.1.3.9 countSubs()

```
size_t Multiset::countSubs (
    const Multiset & subset ) const
```

overloaded function [countEls\(\)](#)

Parameters

<i>subset</i>	
---------------	--

Returns

result of std::count() function

3.1.3.10 delElement() [1/2]

```
void Multiset::delElement (
    const char * element )
```

deletes an element from set

Parameters

<i>element</i>	
----------------	--

3.1.3.11 delElement() [2/2]

```
void Multiset::delElement (
    const string & element )
```

deletes an element from set

Parameters

<i>element</i>	
----------------	--

3.1.3.12 delSubset()

```
void Multiset::delSubset (
    const Multiset & set )
```

deletes a subset from set

Parameters

<i>set</i>	
------------	--

3.1.3.13 findEls()

```
bool Multiset::findEls (
    const string & element ) const
```

determines whether an element in the set

Parameters

<i>element</i>	an element, the occurrences of which are counted
----------------	--

Returns

result of std::find() function

3.1.3.14 findSubs()

```
bool Multiset::findSubs (
    const Multiset & subset ) const
```

overloaded function [findEls\(\)](#)

Parameters

<i>subset</i>	
---------------	--

Returns

result of `std::find()` function

3.1.3.15 getCardinality()

```
size_t Multiset::getCardinality ( ) const
```

`cardinality_` getter

3.1.3.16 isEmpty()

```
bool Multiset::isEmpty ( ) const
```

a method for determining the emptiness of the set

Returns

result for `std::empty()` finction for `elements_` and `subsets_`

3.1.3.17 operator"!="()

```
bool Multiset::operator!= (
    const Multiset & set ) const
```

overloaded operator `"!="`

Parameters

<i>set</i>	
------------	--

Returns

true if two sets are not equal

3.1.3.18 operator*()

```
Multiset Multiset::operator* (
    const Multiset & set ) const
```

overloaded operator "*"

Parameters

<i>set</i>	
------------	--

Returns

intersection of current set and parameter-set

3.1.3.19 operator*=()

```
Multiset Multiset::operator*= (
    const Multiset & set )
```

overloaded operator "*="

Parameters

<i>set</i>	
------------	--

Returns

intersection of current set and parameter-set that assigned to the current set

3.1.3.20 operator+()

```
Multiset Multiset::operator+ (
    const Multiset & set ) const
```

overloaded operator "+"

Parameters

<i>set</i>	
------------	--

Returns

union of current set and parameter-set

3.1.3.21 operator+=()

```
Multiset Multiset::operator+= (
```



```
const Multiset & set )
```

overloaded operator "+="

Parameters

<i>set</i>	
------------	--

Returns

union of current set and parameter-set that assigned to the current set

3.1.3.22 operator-()

```
Multiset Multiset::operator- (
    const Multiset & set ) const
```

overloaded operator "-"

Parameters

<i>set</i>	
------------	--

Returns

difference of current set and parameter-set

3.1.3.23 operator-=()

```
Multiset Multiset::operator-= (
    const Multiset & set )
```

overloaded operator "-="

Parameters

<i>set</i>	
------------	--

Returns

difference of current set and parameter-set that assigned to the current set

3.1.3.24 operator==()

```
bool Multiset::operator== (
    const Multiset & set ) const
```

overloaded operator "=="

Parameters

<i>set</i>	
------------	--

Returns

true if two sets are equal

3.1.3.25 operator[]() [1/3]

```
bool Multiset::operator[] (
    const char * element ) const
```

overloaded operator "[]"

Parameters

<i>element</i>	
----------------	--

Returns

result of [findEls\(\)](#) function

3.1.3.26 operator[]() [2/3]

```
bool Multiset::operator[] (
    const Multiset & subset ) const
```

overloaded operator "[]"

Parameters

<i>subset</i>	
---------------	--

Returns

result of [findEls\(\)](#) function

3.1.3.27 operator[]() [3/3]

```
bool Multiset::operator[] (
    const string & element ) const
```

overloaded operator "[]"

Parameters

<i>element</i>	
----------------	--

Returns

result of [findEls\(\)](#) function

3.1.3.28 parsStrToSet()

```
void Multiset::parsStrToSet (
    const string & setString )
```

parsing a string into a set

Parameters

<i>setString</i>	a string from which a set is formed
------------------	-------------------------------------

3.1.3.29 printSet()

```
void Multiset::printSet ( ) const
```

Console output of a set

3.1.3.30 printSetInfo()

```
void Multiset::printSetInfo ( ) const
```

Console output of a set, its capacity and whether it is empty

3.1.3.31 toString()

```
string Multiset::toString ( ) const
```

forms a string from the class object

Returns

a string

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

- Multitest.h
- mmultisset.cpp

Chapter 4

File Documentation

4.1 Multitest.h

```
00001 #pragma once
00002 #include <iostream>
00003 #include <string>
00004 #include <vector>
00005 #include <algorithm>
00006 using namespace std;
00019 class Multiset
00020 {
00021 public:
00022     vector<string>elements_;
00023     vector<Multiset>subsets_;
00024     size_t cardinality_;
00025
00026     void parsStrToSet(const string& elements);
00027     string toString() const;
00028     size_t countEls(const string& element) const;
00029     size_t countSubs(const Multiset& subset) const;
00030     bool findEls(const string& element) const;
00031     bool findSubs(const Multiset& subset) const;
00032
00033     void addToRes(const string& element, size_t amount);
00034     void addToRes(const Multiset& set, size_t amount);
00035     void addAll(const Multiset& set1, const Multiset& set2);
00036 public:
00037     Multiset();
00038     Multiset(const string& elements);
00039     Multiset(const char* elements);
00040     Multiset(const Multiset& set);
00041
00042     bool isEmpty() const;
00043     Multiset boolean() const;
00044     size_t getCardinality() const;
00045
00046     void addElement(const string& element);
00047     void addElement(const char* element);
00048     void addSubset(const Multiset& set);
00049     void delElement(const string& element);
00050     void delElement(const char* element);
00051     void delSubset(const Multiset& set);
00052
00053     bool operator[](const string& element) const;
00054     bool operator[](const char* element) const;
00055     bool operator[](const Multiset& subset) const;
00056     bool operator==(const Multiset& Set) const;
00057     bool operator!=(const Multiset& Set) const;
00058     Multiset operator+(const Multiset& Set) const;
00059     Multiset operator+= (const Multiset& Set);
00060     Multiset operator*(const Multiset& Set) const;
00061     Multiset operator*= (const Multiset& Set);
00062     Multiset operator-(const Multiset& Set) const;
00063     Multiset operator-= (const Multiset& Set);
00064
00065     void printSetInfo() const;
00066     void printSet() const;
00067 };
```


Index

- addAll
 - Multiset, [7](#)
- addElement
 - Multiset, [7](#), [8](#)
- addSubset
 - Multiset, [8](#)
- addToRes
 - Multiset, [8](#)
- boolean
 - Multiset, [9](#)
- countEls
 - Multiset, [9](#)
- countSubs
 - Multiset, [9](#)
- delElement
 - Multiset, [9](#), [10](#)
- delSubset
 - Multiset, [10](#)
- findEls
 - Multiset, [10](#)
- findSubs
 - Multiset, [10](#)
- getCardinality
 - Multiset, [11](#)
- isEmpty
 - Multiset, [11](#)
- Multiset, [5](#)
 - addAll, [7](#)
 - addElement, [7](#), [8](#)
 - addSubset, [8](#)
 - addToRes, [8](#)
 - boolean, [9](#)
 - countEls, [9](#)
 - countSubs, [9](#)
 - delElement, [9](#), [10](#)
 - delSubset, [10](#)
 - findEls, [10](#)
 - findSubs, [10](#)
 - getCardinality, [11](#)
 - isEmpty, [11](#)
 - Multiset, [6](#), [7](#)
 - operator!=, [11](#)
 - operator+, [12](#)
 - operator+=", [12](#)
 - operator-, [13](#)
 - operator=, [13](#)
 - operator==, [13](#)
 - operator[], [14](#)
 - operator*, [11](#)
 - operator*=", [12](#)
 - parsStrToSet, [15](#)
 - printSet, [15](#)
 - printSetInfo, [15](#)
 - toString, [15](#)
- operator!=
 - Multiset, [11](#)
- operator+
 - Multiset, [12](#)
- operator+="
 - Multiset, [12](#)
- operator-
 - Multiset, [13](#)
- operator=
 - Multiset, [13](#)
- operator==
 - Multiset, [13](#)
- operator[]
 - Multiset, [14](#)
- operator*
 - Multiset, [11](#)
- operator*="
 - Multiset, [12](#)
- parsStrToSet
 - Multiset, [15](#)
- printSet
 - Multiset, [15](#)
- printSetInfo
 - Multiset, [15](#)
- toString
 - Multiset, [15](#)